PLUMED

2.1.5

Generated by Doxygen 1.8.7

Mon Jan 18 2016 09:20:38

Contents

Chapter 1

Introduction

PLUMED is a plugin that works with a large number of molecular dynamics codes. It can be used to analyse features of the dynamics on-the-fly or to perform a wide variety of free energy methods. The original PLUMED 1 [1] was highly successful and had over 1000 users. PLUMED 2 [2] constitues an extensive rewrite of the original in a way that makes it more modular and thus easier to implement new methods, more straightforward to add it to MD codes and hopefully simpler to use. This is the user manual - if you want to modify PLUMED or to understand how it works internally, have a look at the developer manual .

To understand the difference between PLUMED 1 and PLUMED 2, and to follow the development of PLUMED 2, you can look at the detailed Change Log.

A short tutorial explaining why it is a good idea to move from PLUMED 1 to PLUMED 2 and how to do it in practice is also available (see Moving from PLUMED 1 to PLUMED 2).

To install PLUMED 2, see this page: Installation

1.1 About this manual

This manual has been compiled from PLUMED version **2.1.5** (git version: **v2.1.5-1-g44a8350**). Manual built on Travis CI for branch v2.1.

Regtest results for this version can be found here.

Since version 2.1 we provide an experimental PDF manual. The PDF version is still not complete and has some known issue (e.g. some links are not working properly and images are not correctly included), and the html documentation should be considered as the official one. The goal of the PDF manual is to allow people to download a full copy on the documentation for offline access and to perform easily full-text searches. Notice that the manual is updated very frequently (sometime more than once per week), so keep your local version of the PDF manual up to date. Since the PDF manual is 200+ pages and is continuously updated, **please do not print it!**

1.2 A quick introduction

To run PLUMED 2 you need to provide one input file. In this file you specify what it is that PLUMED should do during the course of the run. Typically this will involve calculating one or more collective variables, perhaps calculating a function of these CVs and then doing some analysis of values of your collective variables/functions or running some free energy method. A very brief introduction to the syntax used in the PLUMED input file is provided in this 10-minute video .

More information on the input syntax as well as details on the the various trajectory analsyis tools that come with PLUMED are given in:

- Collective variables tells you about the ways that you can calculate functions of the positions of the atoms.

- Analysis tells you about the various forms of analysis you can run on trajectories using PLUMED.
- Bias tells you about the methods that you can use to bias molecular dynamics simulations with PLUMED.

PLUMED can be used in one of two ways. First, it can be incorporated into an MD code and used to analyse or bias a molecular dynamics run on the fly. Notice that some MD code could already include calls to the PLUMED library and be PLUMED-ready in its original distribution. As far as we know, the following MD codes can be used with PLUMED 2 out of the box:

- AmberTools, sander module, since version 15.
- CP2K, since Feb 2015.
- ESPResSo, in a Plumedized version that can be found here.
- PINY-MD, in its plumed branch.
- IPHIGENIE.

Please refer to the documentation of the MD code to know how to use it with the latest PLUMED release. If you maintain another MD code that is PLUMED-ready let us know and we will add it to this list.

Additionally, we provide patching procedures for the following codes:

- amber14
- gromacs-4-5-7
- gromacs-4-6-7
- gromacs-5-0-4
- gromacs-5-1-0
- lammps-6Apr13
- namd-2-8
- namd-2-9
- qespresso-5-0-2

Alternatively, one can use PLUMED as a standalone tool for postprocessing the results from molecular dynamics or enhanced sampling calculations.

Chapter 2

Change Log

Here you can find a history of changes across different PLUMED versions. We mostly add new features without breaking exising ones. However, some of the changes lead to incompatible behavior. In the Change Log we try to give as much visibility as possible to these changes to avoid surprises.

We also log changes that are relevant if you are developing the code. These change lists are however not complete, and if you want to put your hands in the code and maintain your own collective variables we suggest you to follow the development on github.

2.1 Version 2.0

Version 2.0.0 (Sep 27, 2013)

Version 2.0 is a complete rewrite, so there is no way to write a complete set of difference with respect to plumed 1.3. Here is a possibly incomplete summary of the difference:

- The input is simpler, more flexible, and more error proof. Many checks are now performed and in this way common errors are avoided.
- The units are now the same for all MD codes. If you want to use a different unit than the default you set it in the input file.
- The analysis tools are now much more flexible. As an example of this it is now possible to write different collective variables with different frequencies.
- Many complex collective variables are considerably faster than they were in plumed1. In particular, all variables based on RMSD distances.
- Centers of mass can be used as if they were atoms. Hence, unlike plumed 1.3, you can use center of mass positions in ALL collective variables.
- The virial contribution is now computed and passed to the MD code. Plumed can thus now be used to perform biased NPT simulations.
- Variables can be dumped on different files, and are computed only when this is necessary.
- PLUMED is now compiled as a separate library. This simplifies the patching procedure, but might require some extra work to configure PLUMED properly. Since PLUMED can be loaded as a shared library, it is possible to setup everything such that PLUMED and MD codes can be updated independently from each other.

In addition, it is now much easier to contribute new functionality to the code because:

- There is a much simpler interface between plumed and the base MD codes. This makes it much easier to add plumed to a new MD code. Hopefully, in the future, interfaces with MD codes will be maintained by the developers of the MD codes independently from PLUMED developers. This will allow more MD codes to be compatible with PLUMED.
- There is C++ object oriented programming and full compatibility with the C++ standard library
- A modular structure.
- New collective variables and methods can be released independently.
- There is an extensive developer documentation.
- User documentation is provided together inside the implementation files.

Caveats:

- PLUMED 2 input file (plumed.dat) has a syntax which is not compatible with PLUMED 1. Transition should be easy, but cannot be done just using the new version with the old input file.
- PLUMED 2 is written in C++, thus requires a C++ compiler
- PLUMED 2 may not include all the features that were available in PLUMED 1.

A tutorial explaining how to move from PLUMED 1 to PLUMED 2 is available (see Moving from PLUMED 1 to PLUMED 2).

Version 2.0.1 (Nov 14, 2013)

For users:

- Fixed a bug in HISTOGRAM with REWEIGHT_BIAS. Reweighting was only done when also temperature-reweighting was enabled.
- Fixed a bug that was sometime crashing code with domain decomposition and non-dense simulation boxes (e.g. implicit solvent).
- Performance improvements for GYRATION.
- Flush all files every 10000 steps by default, without need to use FLUSH
- Errors when writing input for switchingfunction are now properly recognized.
- Added message when simplemd is used on a non-existing file.
- Fixed `plumed mklib` such that it deletes the target shared library in case of compilation error.
- Several small fixes in documentation and log file.

For developers:

- Added possibility to setup replica exchange from MD codes in fortran (commands "GREX setMPIFIntercomm" and "GREX setMPIFIntracomm").
- cmd("setStopFlag") should now be called after PLUMED initialization.
- Several small fixes in documentation.

Version 2.0.2 (Feb 11, 2014)

For users:

- Fixed bug with METAD with INTERVAL and replica exchange, including bias exchange. Now the bias is correctly computed outside the boundaries. Notice that this is different from what was done in PLUMED 1.3. Also notice that INTERVAL now works correctly with grids and splines.
- Fixed bug with READ and periodic variables.
- Fixed bug with HISTOGRAM (option USE_ALL_DATA was not working properly).
- Gromacs patch updated to 4.6.5.
- Gromacs patch for 4.6 has been modified to allow for better load balancing when using GPUs.
- Added option 'plumed info –long-version' and 'plumed info –git-version'.
- Added full reference (page/number) to published paper in doc and log.
- Fixed a bug in file backups (only affecting Windows version - thanks to T. Giorgino).
- Added possibility to search in the documentation.
- Several small fixes in documentation and log file.

For developers:

- Fixed makefile dependencies in some auxiliary files in src/lib (∗cmake and ∗inc).
- Changed way modules are linked in src/. E.g. src/colvar/tools/ is not anymore a symlink to src/colvar but a real directory. (Notice that this introduces a regression: when using plumed as an external library some include files could not work - this only applies when plumed is installed; also notice that this is fixed in 2.0.3)
- Patch for gromacs 4.6 now also include original code so as to simplify its modification.
- Added option 'plumed patch –save-originals'.
- Fixed regtest regtest/secondarystructure/rt32 to avoid problems with NUMERICAL_DERIVATIVES.
- Removed include graphs in the documentation (too large).
- Several small fixes in documentation.

Version 2.0.3 (June 30, 2014)

For users:

- Now compiles on Blue Gene Q with IBM compilers.
- Fixed bug in CENTER where default WEIGHTS were missing.
- Fixed broken CONTACTMAP with SUM
- Fixed DUMPATOMS with gro file and more than 100k atoms.
- Added CMDIST in CONTACTMAP to emulate plumed1 CMAP.
- Several small fixes in documentation and log file.

For developers:

- Fixed cmd("getBias") to retrieve bias. It was not working with single precision codes and it was not converting units properly.

- Fixed a regression in 2.0.2 concerning include files from installed plumed (see commit 562d5ea9dfc3).
- Small fix in tools/Random.cpp that allows Random objects to be declared as static.
- Small fix in user-doc compilation, so that if plumed is not found the sourceme.sh file is sourced
- Fixed non-ansi syntax in a few points and a non-important memory leakage.
- Split cltools/Driver.cpp to make parallel compilation faster.

Version 2.0.4 (Sep 15, 2014)

For users:

- Fixed a bug in BIASVALUE that could produce wrong acceptance with replica exchange simulations.
- Fixed a few innocuous memory leaks.
- Fixed reader for xyz files, that now correctly detects missing columns. Also a related regtest has been changed.
- Several small fixes in documentation and log file.

For developers:

- Renamed Value.cpp to BiasValue.cpp

Version 2.0.5 (Dec 15, 2014)

Notice that branch 2.0 will not be longer maintained! All users are invited to switch to version 2.1.

For users:

- Fixed a bug in replica exchange with different Hamiltonians (either lamdba-dynamics or plumed XX-hrex branch) possibly occuring when using charge or mass dependent variables.
- Fixed a bug in analysis (e.g. HISTOGRAM) leading to wrong accumulation of statistics when running a replica exchange simulation.
- Fixed a bug in the calculation of derivatives in histograms. This should be harmless since people usually only consider the value in histograms and not the derivatives.
- Fixed an issue in Makefile that could results in problems when patching an MD code with –shared option (pointed out by Abhi Acharya). This fixes a regression introduced in 2.0.2.
- Small fixes in documentation.

For developers:

- Added warning when performing regtests using an instance of plumed from a different directory

2.2 Version 2.1

Version 2.1.0 (Sep 15, 2014)

Version 2.1 contains several improvements with respect to 2.0. Users currently working with 2.0 should have a look at the section "Changes leading to incompatible behavior" below and might need tiny adjustments in their input files. In 2.1 we restored more features of 1.3 that were missing in 2.0, so users still working with 1.3 could opt for an

upgrade. A tutorial explaining how to move from PLUMED 1 to PLUMED 2 is available (see Moving from PLUMED 1 to PLUMED 2).

Below you find a list of all the changes with respect to version 2.0. Notice that version 2.1 includes already all the fixes in branch 2.0 up to 2.0.4.

Changes from version 2.0 which are relevant for users:

- Changes leading to incompatible behavior:
  - COORDINATION now skips pairs of one atom with itself.
  - Labels of quantities calculated by BIASVALUE have changed from *label*.bias.*argname* to *label.argname*_bias, which is more consistent with steered MD
  - Labels of quantities calculated by ABMD have change from *label*.min_*argname* to *label.argname*_min, which is more consistent with steered MD
  - Labels of quantities calculated by PIECEWISE have change from *label.argnumber* to *label.argname*_↩ pfunc, which is more consistent with steered MD
  - For multicolvars components calculated with LESS_THAN and MORE_THAN keywords are now labelled lessthan and morethan. This change is necessary as the underscore character now has a special usage in component names.
  - In CONTACTMAP components are now labelled *label*.contact- n.
  - The command SPHERE has been replaced by UWALLS.
- New configuration system based on autoconf (use ./configure from root directory). Optional packages are detected at compile time and correctly enabled or disabled. An internal version of lapack and blas will be used if these libraries are not installed.
- New actions:
  - SPRINT topological collective variables.
  - CH3SHIFTS collective variable.
  - POSITION collective variable.
  - FIT_TO_TEMPLATE.
  - COMMITTOR analysis.
  - LOCAL_AVERAGE.
  - NLINKS.
  - DIHCOR.
  - NOE.
  - RDC.
  - CLASSICAL_MDS.
  - XDISTANCES.
  - YDISTANCES.
  - ZDISTANCES.
  - DUMPMULTICOLVAR.
  - Crystallization module, including Q3, LOCAL_Q3, Q4, Q6, LOCAL_Q4, LOCAL_Q6, MOLECULES, SIMPLECUBIC, TETRAHEDRAL and FCCUBIC.
  - ENSEMBLE to perform Replica-Averaging on any collective variable.
- New features for existing actions:
  - METAD : WALKERS_MPI flag (multiple walkers in a mpi-based multi-replica framework), ACCELER↩ ATION flag (calculate on the fly the Metadynamics acceleration factor), TAU option (alternative way to set Gaussian height in well-tempered metadynamics), GRID_SPACING (alternative to GRID_BIN to set grid spacing). Notice that now one can also omit GRID_BIN and GRID_SPACING when using fixed size Gaussian, and the grid spacing will be automatically set.

  - DISTANCE : added SCALED_COMPONENTS
  - COORDINATION : if a single group is provided, it avoids permuted atom indexes and runs at twice the speed.
  - DUMPATOMS : PRECISION option to set number of digits in output file.
  - GROUP : NDX_FILE and NDX_GROUP options to import atom lists from ndx (gromacs) files.
  - In many multicolvars, MIN and MAX options can be used.
  - HISTOGRAM : GRID_SPACING (alternative to GRID_BIN to set grid spacing), FREE-ENERGY flags in addition to standard probability density, additional option for KERNEL=DISCRETE to accumulate standard histograms.
  - sum_hills : added options –spacing (alternative to –bin to set grid spacing) and –setmintozero to translate the minimum of the output files to zero.
  - CONTACTMAP : parallelised and added weights.

- New features in MD patches (require repatch):

  - New patch for Gromacs 5.0
  - Gromacs 4.6.X patch updated to 4.6.7
  - Gromacs 4.6.7 supports COMMITTOR analysis; can be now be used to perform energy minimization; now passes temperature to PLUMED (this allows temperature to be omitted in some actions, namely METAD and analysis actions).

  Notice that if you use runtime binding it is not compulsory to repatch, and that all combinations should work correctly (new/old PLUMED with repatched/non-repatched MD code).

- Other new features:

  - driver can now read trajectories in many formats using VMD molfile plugin (requires VMD plugins to be compiled and installed). In case VMD plugins are not installed, the configuration system falls back to an internal version which implements a minimal list of plugins (gromacs and dcd) (kindly provided by T. Giorgino).
  - switchingfunction : added STRETCH flag.
  - Negative strides in atom ranges (e.g. ATOMS=10-1:-3 is expanded to ATOMS=10,7,4,1).
  - COORDINATION and DHENERGY with NLIST now work correctly in replica exchange simulations.
  - Multicolvars with neighbor lists now work correctly in replica exchange simulations.
  - Improved multicolvar neighbor lists.

- Optimizations:

  - Root-mean-square devations with align weights different from displace weights are now considerably faster. This will affect RMSD calculations plus other variables based on RMSD.
  - WHOLEMOLECULES is slighlty faster.
  - COORDINATION is slighlty faster when NN and MM are even and D_0=0.
  - Atom scattering with domain decomposition is slightly faster.
  - Link cells are now exploited in some multicolvars.
  - Derivatives are not calculated unless they are specifically required, because for instance you are adding a bias.

- Documentation:

  - All tutorial material from the recent plumed meeting in Belfast is now in the manual
  - Improvements to documentation, including lists of referenceable quantities outputted by each action
  - Manual has been re-organized following suggestions received at the plumed meeting.
  - An experimental PDF version of the manual is now provided (a link can be found in the documentation homepage).

Changes from version 2.0 which are relevant for developers:

- Added regtests for plumed as a library (e.g. basic/rt-make-0). plumed command has an additional flag (–is-installed) to probe if running from a compilation directory or from a fully installed copy (this is needed for regtests to work properly).
- Improved class Communicator. Many operations can now be done directly on Vectors, Tensors, std::vector and PLMD::Matrix.
- Modified class RMSD.
- Patches for GPL codes (QuantumEspresso and Gromacs) now also include original code so as to simplify their modification.
- Fixed dependencies among actions such that it is now possible (and reliable) to use MPI calls inside Action↩ ::prepare()
- colvar/CoordinationBase.cpp has been changed to make it faster. If you devised a class which inherits from here, consider that CoordinationBase::pairing now needs *squared* distance instead of distance
- It is possible to run "make install" from subdirectories (e.g. from src/colvar)
- There is a small script which disables/enables all optional modules (make mod-light/mod-heavy/mod-reset)
- Added "-q" option to plumed patch
- You can now create new metrics to measure distances from a reference configurations. If you do so such metrics can then be used in paths straightforwardly
- You can now use multicolvars in tandem with manyrestraints in order to add a large numbers of restraints.
- Can now do multicolvar like things in which each colvar is a vector rather than a scalar.
- Updated script that generated header files so that they properly show years. Notice that the script should new be run from within a git repository

This list is likely incompleted, if you are developing in PLUMED you are encouraged to follow changes on github.

Version 2.1.1 (Dec 15, 2014)

This release includes all the fixes available in branch 2.0 until 2.0.5.

For users:

- New patch for AMBER 14 (sander module only). This patch should be compatible with any PLUMED 2 version (including 2.0). It includes most PLUMED features with the notable exception of multi-replica framework.
- Changed definition in arbitrary phase of eigenvectors. This will change the result of some analysis method where the phase does matter (e.g. CLASSICAL_MDS) and make some regression test better reproducible.
- Fixed a portability issue in BG/P where gettimeofday is not implemented. Notice that this fix implies that one should execute again ./configure to have plumed timing working correctly.
- CS2Backbone: fixed a bug that resulted in only a fraction of the chemical shifts being printed with WRITE_CS and parallel simulations (requires to get the last almost updated from SVN)
- NOE: fixed a bug in the replica-averaging
- Fixed a linking issue with ALMOST, where bz2 was always used to link ALMOST to PLUMED even if it is not compulsory to build ALMOST.
- Fixed a wrong include in the GMX5 patch.
- FUNCPATHMSD can now be used together with CONTACTMAP to define pathways in contactmaps space

- Configuration is more verbose, a warning is given if a default option cannot be enabled and an error is given if an option explicitly enabled cannot be enabled.
- Compilation is less verbose (use "make VERBOSE=1" to have old behavior)
- Small fixes in documentation.

For developers:

- Tests are now performed at every single push on travis-ci.org
- Manual is built and pushed to the online server from travis-ci.org (see developer doc)
- Fixes in developer doc.

Version 2.1.2 (Mar 16, 2015)

For users:

- Added two new short tutorials to the manual (Cambridge tutorial and Munster tutorial).
- Fixed a severe bug on DRMSD - cutoff values were ignored by PLUMED. Notice that this bug was introduced in 2.1.0, so that it should not affect the 2.0.x series.
- Fixed a bug affecting LAMMPS patch used with a single processor. Notice that the fix is inside PLUMED, thus it does not necessarily requires repatching.
- Sander patch now works with multiple replica (no replica exchange yet). It also contains some fix from J. Swails.
- GMX5 patch was not working for bias-exchange like cases
- Patching system now checks for the availabity of shared/static/runtime version of plumed before patching
- Configure now check better if compiler flag are accepted by the compiler. This makes configure on bluegene more robust.
- Sourceme.sh now sets proper library path in linux also.

Version 2.1.3 (June 30, 2015)

For users:

- Fixed bug in ENSEMBLE derivatives when more than 1 argument was provided
- Fixed bug in GHOST : virial is now computed correctly.
- Fixed a serious bug in virial communicated from plumed to gromacs, for both gromacs versions 4.6 and 5.↩ 0. See #132. This fix requires gromacs to be repatched and could be very important if you run biased simulations in the NPT ensemble.
- Fixed a bug in the virial computed with FIT_TO_TEMPLATE when the reference pdb had center non located at the origin.
- Fixed a bug in the the forces computed with FIT_TO_TEMPLATE when used in combination with COM, CENTER, or GHOST
- Fixed a bug that could lead plumed to be stuck with domain decomposition in some extreme case (one domain with all atoms, other domains empty).
- Fixed a bug when COMBINE or MATHEVAL are used with PERIODIC keyword. Now when PERIODIC keyword is used the result of the calculation is brought within the periodicity domain. See #139.

- Fixed a bug related to RANDOM_EXCHANGES followed by INCLUDE
- Fixed bug in derivatives of histogram bead with triangular kernels
- Updated gromacs patch 4.5.5 to 4.5.7
- Updated internal molfile plugins to VMD 1.9.2.
- Included crd and crdbox formats to internal molfile.
- Added –natoms to driver . This is required to read coordinate files with VMD plugins when number of atoms is not present (e.g. amber crd files)
- Added the checks in the driver to detect cases where molinfo does not provide box information (e.g. pdb).
- Added support for readdir_r when available, which makes opening files thread safe.
- CFLAGS now include -fPIC by default
- Added a warning when using METAD without grids with a large number of hills.
- Fixes in user documentation.

For developers:

- Allow external VMD plugins to be detected with –has-external-molfile. This is required to enable some regtest with amber files.
- Added –dump-full-virial to driver
- Allow definition of variables where some of the components have derivatives and some haven't (#131).
- Improved travis tests with more debug options.
- Improved some regtest to check out-of-diagonal virial components
- Improved make cppcheck options.
- Fixes in developer documentation.

Version 2.1.4 (Oct 13, 2015)

For users:

- Fixed NAMD patch. Masses and charges were not passed correctly, thus resulting in wrong COM or CENTER with MASS. This fix required repatching NAMD. Notice that this bug was present also in v2.0 but in a different form. More information here (#162), including a workaround that allows masses to be fixed without repatching.
- When installing with PLUMED_LIBSUFFIX an underscore is used as separator instead of a dash. E.g. `make install PLUMED_LIBSUFFIX=2.1` will result in an executable named `plumed_v2.1`. This fix a potential problem (see Installation).
- Fixed erroneously reported message about MPI at the end of ./configure.
- Changed warning message about undocumented components.
- PLUMED now says in the log file if it was compiled from a dirty git repository.
- Fixed a problem leading to rare random crashes when using METAD with WALKERS_MPI and multiple processors per replica.
- Small change in numerical accuracy of lattice reduction. Should be more robust when running with highly optimizing compilers.
- Fixed a bug in normalisation of kernel functions. This affects HISTOGRAM If these actions were used with previous versions of the code care should be taken when analysing the results.
- Fixed a bug in derivatives of kernel functions with non-diagonal covariances. This affects the derivatives output by sum_hills

Version 2.1.5 (Jan 18, 2016)

Notice that branch 2.1 will not be longer maintained! All users are invited to switch to version 2.2.

For users:

- PLUMED now reports an error when using HISTOGRAM with FREE-ENERGY without USE_ALL_DATA. See #175
- Fixed a bug in configure together with –enable-almost. The check for lbz2 library was not working properly.

Chapter 3

Installation

In this page you can learn how to configure, compile, and install PLUMED. For those of you who are impatient, the following might do the job:

```
> ./configure --prefix=/usr/local
> make -j 4
> make doc # this is optional and requires proper doxygen version installed
> make install
```

Notice that `make install` is not strictly necessary as plumed can be used from the compilation directory. This is very useful so as to quickly test the implementation of new features. However, we strongly recommend to perform a full install.

Once the above is completed the `plumed` executable should be in your execution path and you will be able to use PLUMED to analyze existing trajectories or play with the Lennard-Jones code that is included. However, because PLUMED is mostly used to bias on the fly simulations performed with serious molecular dynamics packages, you can find instructions about how to patch your favorite MD code so that it can be combined with PLUMED below. Again, if you are impatient, something like this will do the job:

```
> cd /md/root/dir
> plumed patch -p
```

Then compile your MD code. For some MD codes these instructions are insufficient. It is thus recommended that you read the instructions at the end of this page. Notice that MD codes could in principle be "PLUMED ready" in their official distribution. If your favorite MD code is available "PLUMED ready" you will have to compile PLUMED first, then (optionally) install it, then check the MD codes' manual to discover how to link it.

3.1 Configuring PLUMED

The "./configure" command just generates a Makefile.conf file and a sourceme.sh file. In PLUMED 2.0 these files were pre-prepared and stored in the directory configurations/. The new ones generated by ./configure should be compatible with the old ones. In other words, if you have difficulties with the new procedure, you can use one of these old configuration files. However, it should be easy to enforce a similar setup on autoconf by passing the proper arguments on the command line. We strongly encourage you to configure plumed in this way. If you have problems on your architecture, please report them to the mailing list.

Useful command line options for ./configure can be found by typing

```
> ./configure --help
```

Notice that some functionalities of PLUMED depend on external libraries which are looked for by configure. You can typically avoid looking for a library using the "disable" syntax, e.g.

```
> ./configure --disable-mpi --disable-matheval
```

Notice that when mpi search is enabled (by default) compilers such as "mpic++" and "mpicxx" are searched for first. On the other hand, if mpi search is disabled ("./configure --disable-mpi") non-mpi compilers are searched for.

Notice that only a few of the possible compiler name are searched. Thus, compilers such as "g++-mp-4.8" should be explicitly requested with the CXX option.

You can better control which compiler is used by setting the variables CXX and CC. E.g., to use Intel compilers use the following command:

```
> ./configure CXX=icpc CC=icc
```

Notice that we are using icpc in this example, which is not an mpi compiler as a result mpi will not be enabled. Also consider that this is different with respect to what some other configure script does in that variables such as MPICXX are completely ignored here. In case you work on a machine where CXX is set to a serial compiler and MPICXX to a MPI compiler, to compile with MPI you should use

```
> ./configure CXX="$MPICXX"
```

To tune the compilation options you can use the CXXFLAGS variable:

```
> ./configure CXXFLAGS=-O3
```

If you are implementing new functionality and want to build with debug flags in place so as to do some checking you can use

```
> ./configure --enable-debug
```

This will perform some extra check during execution (possibly slowing down PLUMED) and write full symbol tables in the executable (making the final executable much larger).

The main goal of the automatic configure is to find the libraries. When they are stored in unconventional places it is thus sensible to tell autoconf where to look! To do this there are some environment variable that can be used to instruct the linker which directories it should search for libraries inside. These variables are compiler dependent, but could have been set by the system administrator so that libraries are found without any extra flag. Our suggested procedure is to first try to configure without any additional flags and to then check the log so as to see whether or not the libraries were properly detected.

If a library is not found during configuration, you can try to use options to modify the search path. For example if your matheval libraries is in /opt/local (this is where MacPorts put it) and configure is not able to find it you can try

```
> ./configure LDFLAGS=-L/opt/local/lib CPPFLAGS=-I/opt/local/include
```

Notice that PLUMED will first try to link a routine from say matheval without any additional flag, and then in case of failure will retry adding "-lmatheval" to the LIBS options. If also this does not work, the matheval library will be disabled and some features will not be available. This procedure allows you to use libraries with custom names. So, if your matheval library is called /opt/local/lib/libmymatheval.so you can link it with

```
> ./configure LDFLAGS=-L/opt/local/lib CPPFLAGS=-I/opt/local/include LIBS=-lmymatheval
```

In this example, the linker will directly try to link `/opt/local/lib/libmymatheval.so`. This rule is true for all the libraries, so that you will always be able to link a specific version of a library by specifying it using the LIBS variable.

Warning

On Linux you might have problems using the LDFLAGS option. In particular, if makefile complaints that it cannot link the file 'src/lib/plumed-shared', try to set correctly the runtime path by using

```
> ./configure LDFLAGS="-L/opt/local/lib -Wl,-rpath,/opt/local/lib" \
  CPPFLAGS=-I/opt/local/include LIBS=-lmymatheval
```

Notice that although the file 'src/lib/plumed-shared' is not necessary, being able to produce it means that it will be possible to link PLUMED dynamically with MD codes later.

PLUMED needs blas and lapack. These are treated slighty different from other libraries. The search is done in the usual way (i.e., first look for them without any link flag, then add "-lblas" and "-llapack", respectively). As such if you want to use a specific version of blas or lapack you can make them available to configure by using

```
> ./configure LDFLAGS=-L/path/to/blas/lib LIBS=-lnameoflib
```

If the functions of these libraries are not found, the compiler looks for a version with a final underscore added. Finally, since blas and lapack are compulsory in PLUMED, you can use a internal version of these libraries that comes as part of PLUMED. If all else fails the internal version of BLAS and LAPACK are the ones that will be used by PLUMED. If you wish to disable any search for external libraries (e.g. because the system libraries have problems) this can be done with

```
> ./configure --disable-external-lapack
```

As a final resort, you can also edit the resulting Makefile.conf file. Notable variables in this file include:

- DYNAMIC_LIB : these are the libraries needed to compile the PLUMED library (e.g. -L/path/to/matheval -lmatheval etc). Notice that for the PLUMED shared library to be compiled properly these should be dynamic libraries. Also notice that PLUMED preferentially requires BLAS and LAPACK library; see BLAS and LAPACK for further info. Notice that the variables that you supply with `configure LIBS=something` will end up in this variable. This is a bit misleading but is required to keep the configuration files compatible with PLUMED 2.0.

- LIBS : these are the libraries needed when patching an MD code; typically only "-ldl" (needed to have functions for dynamic loading).

- CPPFLAGS : add here definition needed to enable specific optional functions; e.g. use -D__PLUMED_HAS_MATHEVAL to enable the matheval library

- SOEXT : this gives the extension for shared libraries in your system, typically "so" on unix, "dylib" on mac; If your system does not support dynamic libraries or, for some other reason, you would like only static executables you can just set this variable to a blank ("SOEXT=").

3.1.1 BLAS and LAPACK

We tried to keep PLUMED as independent as possible from external libraries and as such those features that require external libraries (e.g. Almost and Matheval) are optional. However, to have a properly working version of plumed PLUMED you need BLAS and LAPACK libraries. We would strongly recommend you download these libraries and install them separately so as to have the most efficient possible implementations of the functions contained within them. However, if you cannot install blas and lapack, you can use the internal ones. Since version 2.1, PLUMED uses a configure script to detect libraries. In case system LAPACK or BLAS are not found on your system, PLUMED will use the internal replacement.

We have had a number of emails (and have struggled ourselves) with ensuring that PLUMED can link BLAS and LAPACK. The following describes some of the pitfalls that you can fall into and a set of sensible steps by which you can check whether or not you have set up the configuration correctly.

Notice first of all that the DYNAMIC_LIB variable in the Makefile.conf should contain the flag necessary to load the BLAS and LAPACK libraries. Typically this will be -llapack -lblas, in some case followed by -lgfortran. Full path specification with -L may be necessary and on some machines the blas and lapack libraries may not be called -llapack and -lblas. Everything will depend on your system configuration.

Some simple to fix further problems include:

- If the linker complains and suggests recompiling lapack with -fPIC, it means that you have static lapack libraries. Either install dynamic lapack libraries or switch to static compilation of PLUMED by unsetting the SOEXT variable in the configuration file.

- If the linker complains about other missing functions (typically starting with "for_" prefix) then you should also link some Fortran libraries. PLUMED is written in C++ and often C++ linkers do not include Fortran libraries by default. These libraries are required for lapack and blas to work. Please check the documentation of your compiler.

- If the linker complains that dsyevr_ cannot be found, try adding -DF77_NO_UNDERSCORE to CPPFLAGS Notice that "./configure" should automatically try this solution.

3.2 Compiling PLUMED

Once configured, PLUMED can be compiled using the following command:

```
> make -j 4
```

This will compile the entire code and produce a number of files in the 'src/lib' directory, including the executable 'src/lib/plumed'. When shared libraries are enabled, a shared libraries called 'src/lib/libKernel.so' should also be present. Notice that the extension could be '.dylib' on a Mac.

In case you want to run PLUMED *without installing it* (i.e. from the compilation directory), you can use the file 'sourceme.sh' that has been created by the configure script in the main PLUMED directory. This file can be "sourced" (presently only working for bash shell) if you want to use PLUMED *before installing it* (i.e. from the compilation directory). It is a good idea to source it now, so that you can play with the just compiled PLUMED:

```
> source sourceme.sh
```

Now a "plumed" executable should be in your path. Try to type

```
> plumed -h
```

Warning

If you are cross compiling, the plumed executable will not work. As a consequence, you won't be able to run regtests or compile the manual. This is not a problem.

You can also check if PLUMED is correctly compiled by performing our regression tests. Be warned that some of them fail because of the different numerical accuracy on different machines.

```
> cd regtest
> make
```

Notice that regtests are performed using the "plumed" executable that is currenty in the path. You can check the exact version they will use by using the command

```
> which plumed
```

This means that if you do not source "sourceme.sh", the tests will fails. This does not mean that plumed is not working it just means that you haven't told them shell where to find plumed!

Notice that the compiled executable, which now sits in 'src/lib/plumed', relies on other resource files present in the compilation directory. This directory should thus stay in the correct place. One should thus not rename or delete it. In fact the path to the PLUMED root directory is hardcoded in the plumed executable as can be verified using

```
> plumed info --root
```

In case you try to use the plumed executable without the compilation directory in place (e.g. you move away the src/lib/plumed static executable and delete or rename the compilation directory) PLUMED will not work correctly and will give you an error message

```
> plumed help
ERROR: I cannot find /xxx/yyy/patches directory
```

You can force plumed to run anyway by using the option –standalone-executable:

```
> plumed --standalone-executable help
```

Many features will not be available if you run in this way. However, this is currently the only way to use the PLUMED static executable on Windows.

3.3 Installing PLUMED

It is strongly suggested to install PLUMED in a predefined location. This is done using

```
> make install
```

This will allow you to remove the original compilation directory, or to recompile a different PLUMED version in the same place.

To install PLUMED one should first decide the location. The standard way to do it is during the configure step:

```
> ./configure --prefix=$HOME/opt
> make
> make install
```

However, you can also change it after compilation setting the environment variable PLUMED_PREFIX.

```
> ./configure
> make
> export PLUMED_PREFIX=$HOME/opt
> make install
```

If you didn't specify the –prefix option during configure, and you did not set the PLUMED_PREFIX environment variable, PLUMED will be installed in /usr/local. The install command should be executed with root permissions (e.g. "sudo make install") if you want to install PLUMED on a system directory. Notice that upon installation PLUMED currently needs to relink a library. If root user does not have access to compilers, "sudo -E make install" might solve the issue. An almost full copy of the compilation directory will be installed into $PLUMED_PREFIX/lib/plumed/ directory. A link to the proper PLUMED executable will be set up in $PLUMED_PREFIX/bin, PLUMED include files will be copied to $PLUMED_PREFIX/include/plumed and PLUMED libraries will be linked to $PLUMED_PREFIX/lib.

One should then set the environment properly. We suggest to do it using the module framework (http://modules.sourceforge.net). An ad hoc generated module file for PLUMED can be found in $PLUMED_PREFIX/lib/plumed/src/lib/modulefile Just edit it as you wish and put it in your modulefile directory. This will also allow you to install multiple PLUMED versions on your machine and to switch amongst them. If you do not want to use modules, you can still have a look at the modulefile we did so as to know which environment variables should be set for PLUMED to work correctly.

If the environment is properly configured one should be able to do the following things:

- use the "plumed" executable from the command line. This is also possible before installing.
- link against the PLUMED library using the "-lplumed" flag for the linker. This allows one to use PLUMED library in general purpose programs
- use the PLUMED internal functionalities (C++ classes) including header files such as "#include <plumed/tools/Vector.h>". This is useful as it may be expedient to exploit the PLUMED library in general purpose programs

As a final note, if you want to install several PLUMED versions without using modules then you can define the environment variable PLUMED_LIBSUFFIX using:

```
> export PLUMED_PREFIX=$HOME/opt
> export PLUMED_LIBSUFFIX=v2.0
> make install
```

This will install a plumed executable named "plumed_v2.0". All the other files will be renamed similarly, e.g. the PLUMED library will be loaded with "-lplumed_v2.0" and the PLUMED header files will be included with "#include <plumed_v2.0/tools/Vector.h>". This trick is useful if you do not want to set up modules, but we believe that using modules as described above is more flexible.

Warning

Until version 2.1.3, the added suffix was in the form `-suffix` instead of `_suffix`. This could cause clashes with the fact that plumed reserved names with a `-something` appended for scripts used in cross compiling (see below). As a consequence, by doing

```
> make install PLUMED_LIBSUFFIX=v2.0
> make install
```

The second install would remove all the executables installed by the first install.

3.4 Patching your MD code

In case your MD code is not supporting PLUMED already, you should modify it. We provide scripts to adjust some of the most popular MD codes so as to provide PLUMED support. At the present times we support patching the following list of codes:

- amber14
- gromacs-4-5-7
- gromacs-4-6-7
- gromacs-5-0-4
- gromacs-5-1-0
- lammps-6Apr13
- namd-2-8
- namd-2-9
- qespresso-5-0-2

In the section Code specific notes you can find information specific for each MD code.

To patch your MD code, you should have already installed PLUMED properly. This is necessary as you need to have the command "plumed" in your execution path. As described above this executible will be in your paths if plumed was installed or if you have run sourceme.sh

Once you have a compiled and working version of plumed, follow these steps to add it to an MD code

- Configure and compile your MD enginge (look for the instructions in its documentation).
- Test if the MD code is working properly.
- Go to the root directory for the source code of the MD engine.
- Patch with PLUMED using:

  ```
  > plumed patch -p
  ```

  The script will interactively ask which MD engine you are patching.
- Once you have patched recompile the MD code (if dependencies are set up properly in the MD engine, only modified files will be recompiled)

There are different options available when patching. You can check all of them using

```
> plumed patch --help
```

Particularly interesting options include:

- –static (default) just link PLUMED as a collection of object files. This is only suggested if for external reasons you absolutely need a static executable. Notice that with this setting it is often more complicated to configure properly the MD code, since all the libraries that PLUMED depends on should be properly specified. The `./configure` script does its best in this sense, but sometime it cannot solve the problem. Additionally, this patching mode has been reported not to work properly on OSX.

- –shared allows you to link PLUMED as a shared library. As a result when PLUMED is updated, there will be no need to recompile the MD code. This is way better than –static since the libraries that PLUMED depends on should be automatically linked. Notice that if you later remove the directory where PLUMED is installed also the MD code will not run anymore.

- –runtime allows you to choose the location of the PLUMED library at runtime by setting the variable PLUMED_KERNEL. This is probably the most flexible option, and we encourage system administrators to use this option when installing PLUMED on shared facilities. Indeed, using this setting it will be possible to update separately the PLUMED library and the MD code, leaving to the user the possibility to combine different versions at will. We also recommend to use the provided modulefile (see above) to properly set the runtime environment.

Notice that it is not currently possible to link PLUMED as a static library (something like 'libplumed.a'). The reason for this is that PLUMED heavily relies on C++ static constructors that do not behave well in static libraries. For this reason, to produce a static executable with an MD code + PLUMED we link PLUMED as a collection of object files.

If your MD code is not supported, you may want to implement an interface for it. Refer to the `developer manual` .

3.5 Cross compiling

If you are compiling an executable from a different machine, then `plumed` executable will not be available in the compilation environment. This means that you won't be able to perform regtests on the machine nor to compile the manual. You can try to run the regtests on the computing nodes, but this might require some tweak since often machines where people do cross compiling have architectures with limited capabilities on the compute nodes. Also notice that many of the `plumed` options (e.g. patch) are implemented as shell scripts launched from within the `plumed` executable. If the compute nodes have some limitation (e.g. they do not allow to fork new processes) these options will not work. Anyway, the PLUMED library in combination with an MD software should work if both PLUMED and the MD software have been properly compiled.

Also notice that it will not be possible to use the command `plumed patch` on the machine where you are compiling. You should thus use `plumed-patch` instead of `plumed patch` (notice that it should be written as a single word). Try e.g.:

```
> plumed-patch --help
```

This script provides a "shell only" implementation of `plumed patch` that will skip the launch of the `plumed` executable.

3.6 Installing PLUMED with ALMOST

In order to used some of the NMR based collective variables (CS2BACKBONE and CH3SHIFTS) PLUMED needs to be linked with ALMOST. To do this the free package ALMOST v.2.1 MUST be dowloaded via SVN (svn checkout svn://svn.code.sf.net/p/almost/code/ almost-code). ALMOST 2.1 can be found in branches/almost-2.1/ and can be compiled:

Warning

ALMOST needs SQLITE3 and GZIP installed on your computer.
ALMOST cannot be installed in the same folder of the source code, use –prefix to install it in a different folder

```
> ./configure --prefix="wherever you want it" CXXFLAGS="-O3 -fPIC" CFLAGS="-O3 -fPIC"
> make
> make install
```

Sometimes ALMOST can give errors related to the automake tools. To fix them it is often enough to execute

```
> autoreconf -fi
> automake
```

and then repeat the configuration and compilation instructions.

PLUMED will not use the RDCs module of ALMOST so you can ignore the warning about LAPACK.

Once ALMOST is installed, PLUMED 2 can then be configured with ALMOST enabled:

```
> ./configure --enable-almost CPPFLAGS="-I/ALMOST_INSTALL_PATH/include \
  -I/ALMOST_INSTALL_PATH/include/almost" LDFLAGS="-L/ALMOST_INSTALL_PATH/lib"
```

with ALMOST_INSTALL_PATH set to the full path to the ALMOST installation folder.

3.7 Code specific notes

Here you can find instructions that are specific for patching each of the supported MD codes.

- amber14
- gromacs-4.5.7
- gromacs-4.6.7
- gromacs-5.0.4
- gromacs-5.1.0
- lammps-6Apr13
- namd-2.8
- namd-2.9
- qespresso-5.0.2

3.7.1 amber14

PLUMED can be incorporated into amber (sander module) using the standard patching procedure. Patching must be done in the root directory of amber *before* compilation.

To enable PLUMED in a sander simulation one should use add to the cntrl input namelist these two fields:

plumed=1 , plumedfile='plumed.dat'

The first is switching plumed on, the second is specifying the name of the plumed input file.

This patch is compatible with the MPI version of sander and support multisander. However, replica exchange is not supported. Multisander can thus only be used for multiple walkers metadynamics or for ensemble restraints.

Bug Charges passed from amber to plumed are in wrong units and thus lead to wrong results for variables depending on their values. See http://github.com/plumed/plumed2/issues/165 for more details.

For more information on amber you should visit http://ambermd.org

3.7.2 gromacs-4.5.7

PLUMED can be incorporated into gromacs using the standard patching procedure. Patching must be done in the gromacs source directory *after* gromacs has been configured but *before* gromacs is compiled. Gromcas should be configured with ./configure (not cmake).

To enable PLUMED in a gromacs simulation one should use mdrun with an extra -plumed flag. The flag can be used to specify the name of the PLUMED input file, e.g.:

mdrun -plumed plumed.dat

For more information on gromacs you should visit http://www.gromacs.org

3.7.3 gromacs-4.6.7

PLUMED can be incorporated into gromacs using the standard patching procedure. Patching must be done in the gromacs root directory *before* the cmake command is invoked.

To enable PLUMED in a gromacs simulation one should use mdrun with an extra -plumed flag. The flag can be used to specify the name of the PLUMED input file, e.g.:

mdrun -plumed plumed.dat

For more information on gromacs you should visit http://www.gromacs.org

3.7.4 gromacs-5.0.4

PLUMED can be incorporated into gromacs using the standard patching procedure. Patching must be done in the gromacs root directory *before* the cmake command is invoked.

To enable PLUMED in a gromacs simulation one should use mdrun with an extra -plumed flag. The flag can be used to specify the name of the PLUMED input file, e.g.:

gmx mdrun -plumed plumed.dat

For more information on gromacs you should visit http://www.gromacs.org

3.7.5 gromacs-5.1.0

PLUMED can be incorporated into gromacs using the standard patching procedure. Patching must be done in the gromacs root directory *before* the cmake command is invoked.

To enable PLUMED in a gromacs simulation one should use mdrun with an extra -plumed flag. The flag can be used to specify the name of the PLUMED input file, e.g.:

gmx mdrun -plumed plumed.dat

For more information on gromacs you should visit http://www.gromacs.org

3.7.6 lammps-6Apr13

PLUMED can be incorporated into LAMMPS using a simple patching procedure. Patching must be done *before* LAMMPS is configured. After patching, one should enable PLUMED using the command make yes-user-plumed In the same way, before reverting one should disable PLUMED using the command make no-user-plumed

Also notice that command "fix plumed" should be used in lammps input file *after* the relevant input parameters have been set (e.g. after "timestep" command)

See also http://lammps.sandia.gov/doc/Section_commands.html for further info on processing LAMMPS input, as well as this discussion on github: http://github.com/plumed/plumed2/issues/67.

For more information on LAMMPS you should visit http://lammps.sandia.gov/

3.7.7 namd-2.8

Bug NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only

For more information on NAMD you should visit http://www.ks.uiuc.edu/Research/namd/

3.7.8 namd-2.9

Bug NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only

For more information on NAMD you should visit http://www.ks.uiuc.edu/Research/namd/

3.7.9 qespresso-5.0.2

For more information on Quantum Espresso you should visit http://www.quantum-espresso.org

Chapter 4

Getting started

To run PLUMED 2 you need to provide one input file. In this file you specify what it is that PLUMED should do during the course of the run. Typically this will involve calculating one or more collective variables, perhaps calculating a function of these CVs and then doing some analysis of values of your collective variables/functions or running some free energy method. A very brief introduction to the syntax used in the PLUMED input file is provided in this 10-minute video .

Additional material and examples can be also found in the tutorial Belfast tutorial: Analyzing CVs.

Within this input file every line is an instruction for PLUMED to perform some particular action. This could be the calculation of a colvar, an occasional analysis of the trajectory or a biassing of the dynamics. The first word in these lines specify what particular action is to be performed. This is then followed by a number of keywords which provide PLUMED with more details as to how the action is to be performed. These keywords are either single words (in which they tell PLUMED to do the calculation in a particular way - for example NOPBC tells PLUMED to not use the periodic bounadry conditions when calculating a particular colvar) or they can be words followed by an equals sign and a comma separated list *with no spaces* of numbers or characters (so for example ATOMS=1,2,3,4 tells PLUMED to use atom numbers 1,2,3 and 4 in the calculation of a particular colvar). Space separated lists can be used instead of commma separated list if the entire list is enclosed in curly braces (e.g. ATOMS={1 2 3 4}). Please note that you can split commands over multiple lines by using Continuation lines.

The most important of these keywords is the label keyword as it is only by using these labels that we can pass data from one action to another. As an example if you do:

```
DISTANCE ATOMS=1,2
```

(see DISTANCE)

Then PLUMED will do nothing other than read in your input file. In contrast if you do:

```
DISTANCE ATOMS=1,2 LABEL=d1
PRINT ARG=d1 FILE=colvar STRIDE=10
```

(see PRINT)

then PLUMED will print out the value of the distance between atoms 1 and 2 every 10 steps to the file colvar as you have told PLUMED to take the value calculated by the action d1 and to print it. You can use any character string to label your actions as long as it does not begin with the symbol @. Strings beginning with @ are used by within PLUMED to reference special, code-generated groups of atoms and to give labels to any Actions for which the user does not provide a label in the input.

Notice that if a word followed by a column is added at the beginning of the line (e.g. pippo:), PLUMED automatically removes it and adds an equivalent label (LABEL=pippo). Thus, a completely equivalent result can be obtained with the following shortcut:

```
d1: DISTANCE ATOMS=1,2
PRINT ARG=d1 FILE=colvar STRIDE=10
```

Also notice that all the actions can be labeled, and that many actions besides normal collective variables can define one or more value, which can be then referred using the corresponding label.

Actions can be referred also with POSIX regular expressions (see Regular Expressions) if regex library is available on your system and detected at configure time. You can also add Comments to the input or set up your input over multiple files and then create a composite input by Including other files.

4.1 Plumed units

By default the PLUMED inputs and outputs quantities in the following units:

- Energy - kJ/mol
- Length - nanometers
- Time - picoseconds

Unlike PLUMED 1 the units used are independent of the MD engine you are using. If you want to change these units you can do this using the UNITS keyword.

4.2 UNITS

	This is part of the setup module

This command sets the internal units for the code. A new unit can be set by either specifying how to convert from the plumed default unit into that new unit or by using the shortcuts described below. This directive MUST appear at the BEGINNING of the plumed.dat file. The same units must be used througout the plumed.dat file.

Notice that all input/output will then be made using the specified units. That is: all the input parameters, all the output files, etc. The only exceptions are file formats for which there is a specific convention concerning the units. For example, trajectories written in .gro format (with DUMPATOMS) are going to be always in nm.

Options

NATURAL	(default=off) use natural units

LENGTH	the units of lengths. Either specify a conversion factor from the default, nm, or A (for angstroms) or um
ENERGY	the units of energy. Either specify a conversion factor from the default, kj/mol, or use j/mol or kcal/mol
TIME	the units of time. Either specify a conversion factor from the default, ps, or use ns or fs

Examples

```
# this is using nm - kj/mol - fs
UNITS LENGTH=nm TIME=fs
```

If a number, x, is found, the new unit is equal to x (default units)

```
# this is using nm - kj/mol - fs
UNITS LENGTH=nm TIME=0.001
```

Chapter 5

Collective variables

Chemical systems contain an enormous number atoms, which, in most cases makes it simply impossible for us to understand anything by monitoring the atom postions directly. Consquentially, we introduce Collective variables (CVs) that describe the chemical processes we are interested in and monitor these simpler quantities instead. These CVs are used in many of the methods implemented in PLUMED - there values can be monitored using PRINT, Functions of them can be calculated or they can be analyzed or biased using the Analysis and Biasing methods implemented in PLUMED. Before doing any of these things however we first have to tell PLUMED how to calculate them.

The simplest collective variables that are implemented in PLUMED 2 take in a set of atomic positions and output one or multiple scalar CV values. Information on these variables is given on the page entitled CV Documentation while information as to how sets of atoms can be selected can be found in the pages on Groups and Virtual Atoms. Please be aware that PLUMED contains implementations of many other collective variables but that the input for these variables may be less transparent when it is first encourntered. In particular, the page on Distances from reference configurations describes the various ways that you can calculate the distance from a particular reference configuration. So you will find instructions on how to calculate the RMSD distance from the folded state of a protein here. Meanwhile, the page on Functions describes the various functions of collective variables that can be used in the code. This is a very powerful feature of PLUMED as you can use the Functions commands to calculate any function or combination of the simple collective variables listed on the page CV Documentation. Lastly the page on MultiColvar Documentation describes MultiColvars. MultiColvars allow you to use many different colvars and allow us to implement all these collective variables without implementing having an unmanigiably large ammount of code. For some things (e.g. DISTANCES GROUPA=1 GROUPB=2-100 LESS_THAN={RATIONAL R_0=3}) there are more computationally efficient options available in plumed (e.g. COORDINATION). However, MultiColvars are worth investigating as they provide a flexible syntax for many quite-complex CVs.

- Groups and Virtual Atoms
- CV Documentation
- Distances from reference configurations
- Functions
- MultiColvar Documentation

5.1 Groups and Virtual Atoms

5.1.1 Specifying Atoms

The vast majority of the CVs implemented in PLUMED are calculated from a list of atom positions. Within PLUMED atoms are specified using their numerical indices in the molecular dynamics input file.

In PLUMED lists of atoms can be either provided directly inside the definition of each collective variable, or predefined as a GROUP that can be reused multiple times. Lists of atoms can be written as:

- comma separated lists of numbers (GROUP ATOMS=10,11,15,20 LABEL=g1)
- numerical ranges. So GROUP ATOMS=10-20 LABEL=g2 is equivalent to GROUP ATOMS=10,11,12,13,14,15,16,17,18,19,20 LABEL=g2
- numerical ranges with a stride. So GROUP ATOMS=10-100:10 LABEL=g3 is equivalent to GROUP ATOMS=10,20,30,40,50,60,70,80,90,100 LABEL=g3
- atoms ranges with a negative stride. So GROUP ATOMS=100-10:-10 LABEL=g4 is equivalent to GROUP ATOMS=100,90,80,70,60,50,40,30,20,10 LABEL=g4
- all the above methods together. For example GROUP ATOMS=1,2,10-20,40-60:5,100-70:-2 LABEL=g5.

Some collective variable must accept a fixed number of atoms, for example a DISTANCE is calculated using two atoms only, an ANGLE is calcuated using either 3 or 4 atoms and TORSION is calculated using 4 atoms.

Additional material and examples can be also found in the tutorial Belfast tutorial: Analyzing CVs.

5.1.1.1 Molecules

In addition, for certain colvars, pdb files can be read in using the following keywords and used to select ATOMS:

MOLINFO	This command is used to provide information on the molecules that are present in your system.

The information on the molecules in your system can either be provided in the form of a pdb file or as a set of lists of atoms that describe the various chains in your system using MOLINFO. If a pdb file is used plumed the MOLINFO command will endeavor to recognize the various chains and residues that make up the molecules in your system using the chainIDs and resnumbers from the pdb file. You can then use this information in commands where this has been implemented to specify atom lists. One place where this is particularly useful is when using the commands ALPHARMSD, ANTIBETARMSD and PARABETARMSD.

MOLINFO also introduces special groups that can be used in atom selection. These special groups always begin with a @ symbol. The following special groups are currently available in PLUMED:

Symbol	Topology type	Despription
@phi-#	protein	The torsional angle defined by the C, CA, N and C atoms of the protein backbone in the #th residue. See http://en.wikipedia.org/wiki/Ramachandran_plot
@psi-#	protein	The torsional angle defined by the N, C, CA and N atoms of the protein backbone in the #th residue. See http://en.wikipedia.org/wiki/Ramachandran_plot
@omega-#	protein	The torsional angle defined by the CA, N, C and CA atoms of the protein backbone in the #th residue. See http://en.wikipedia.org/wiki/Ramachandran_plot

@chi1-#	protein	The first torsional angle of the sidechain of the #th residue. Be aware that this angle is not defined for GLY or ALA residues. See http://en.wikipedia.org/wiki/Ramachandran_plot

The following example shows how to use MOLINFO with TORSION to calculate the torsion angles phi and psi for the first and fourth residue of the protein:

```
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
t1: TORSION ATOMS=@phi-3
t2: TORSION ATOMS=@psi-4
PRINT ARG=t1,t2 FILE=colvar STRIDE=10
```

5.1.1.2 Broken Molecules and PBC

PLUMED is designed so that for the majority of the CVs implemented the periodic boundary conditions are treated in the same manner as they would be treated in the host code. In some codes this can be problematic when the colvars you are using involve some property of a molecule. These codes allow the atoms in the molecules to become separated by periodic boundaries, a fact which PLUMED could only deal with were the topology passed from the MD code to PLUMED. Making this work would involve a lot laborious programming and goes against our original aim of having a general patch that can be implemented in a wide variety of MD codes. Consequentially, we have implemented a more pragmatic solution to this probem - the user specifies in input any molecules (or parts of molecules) that must be kept in tact throughout the simulation run. In PLUMED 1 this was done using the ALIGN_ATOMS keyword. In PLUMED 2 the same effect can be acchieved using the WHOLEMOLECULES command.

The following input computes the end-to-end distance for a polymer of 100 atoms and keeps it at a value around 5.

```
WHOLEMOLECULES ENTITY0=1-100
e2e: DISTANCE ATOMS=1,100 NOPBC
RESTRAINT ARG=e2e KAPPA=1 AT=5
```

Notice that NOPBC is used to be sure in DISTANCE that if the end-to-end distance is larger than half the simulation box the distance is compute properly. Also notice that, since many MD codes break molecules across cell boundary, it might be necessary to use the WHOLEMOLECULES keyword (also notice that it should be before distance).

Notice that most expressions are invariant with respect to a change in the order of the atoms, but some of them depend on that order. E.g., with WHOLEMOLECULES it could be useful to specify atom lists in a reversed order.

```
# to see the effect, one could dump the atoms as they were before molecule reconstruction:
# DUMPATOMS FILE=dump-broken.xyz ATOMS=1-20
WHOLEMOLECULES STRIDE=1 ENTITY0=1-20
DUMPATOMS FILE=dump.xyz ATOMS=1-20
```

Notice that since PLUMED 2.1 it is also possible to shift coordinates stored within PLUMED so as to align them to a template structure, using the FIT_TO_TEMPLATE keyword.

5.1.2 Virtual Atoms

Sometimes, when calculating a colvar, you may not want to use the positions of a number of atoms directly. Instead you may wish to use the position of a virtual atom whose position is generated based on the positions of a collection of other atoms. For example you might want to use the center of mass of a group of atoms. Plumed has a number of routines for calculating the positions of these virtual atoms from lists of atoms:

CENTER	Calculate the center for a group of atoms, with arbitrary weights.
COM	Calculate the center of mass for a group of atoms.
GHOST	Calculate the absolute position of a ghost atom with fixed coordinatesin the local reference frame formed by three atoms. The computed ghost atom is stored as a virtual atom that can be accessed inan atom list through the the label for the GHOST action that creates it.

To specify to a colvar that you want to use the position of a virtual atom to calculate a colvar rather than one of the atoms in your system you simply use the label for your virtual atom in place of the usual numerical index. Virtual atoms and normal atoms can be mixed together in the input to colvars as shown below:

```
COM ATOMS=1,10 LABEL=com1
DISTANCE ATOMS=11,com1
```

If you don't want to calculate CVs from the virtual atom. That is to say you just want to monitor the position of a virtual atom (or any set of atoms) over the course of your trajectory you can do this using DUMPATOMS.

5.1.3 GROUP

	This is part of the generic module

Define a group of atoms so that a particular list of atoms can be referenced with a single label in definitions of CVs or virtual atoms.

Atoms can be listed as comma separated numbers (i.e. 1,2,3,10,45,7,9,..) , simple positive ranges (i.e. 20-40), ranges with a stride either positive or negative (i.e. 20-40:2 or 80-50:-2) or as combinations of all the former methods (1,2,4,5,10-20,21-40:2,80-50:-2).

Finally, lists can be imported from ndx files (GROMACS format). Use NDX_FILE to set the name of the index file and NDX_GROUP to set the name of the group to be imported (default is first one).

Notice that this command just creates a shortcut, and does not imply any real calculation. It is just convenient to better organize input files. Might be used in combination with the INCLUDE command so as to store long group definitions in a separate file.

The atoms involved can be specified using

ATOMS	the numerical indexes for the set of atoms in the group. For more information on how to specify lists of atoms see Groups and Virtual Atoms

NDX_FILE	the name of index file (gromacs syntax)
NDX_GROUP	the name of the group to be imported (gromacs syntax) - first group found is used by default

Examples

This command create a group of atoms containing atoms 1,4,7,11 and 14 (labeled 'o'), and another containing atoms 2,3,5,6,8,9,12,13 (labeled 'h'):

```
o: GROUP ATOMS=1,4,7,11,14
h: GROUP ATOMS=2,3,5,6,8,9,12,13
# compute the coordination among the two groups
c: COORDINATION GROUPA=o GROUPB=h R_0=0.3
```

```
# same could have been obtained without GROUP, just writing:
# c: COORDINATION GROUPA=1,4,7,11,14 GROUPB=2,3,5,6,8,9,12,13

# print the coordination on file 'colvar'
PRINT ARG=c FILE=colvar
```

(see also COORDINATION and PRINT)

Groups can be conveniently stored in a separate file. E.g. one could create a file named 'groups.dat' which reads

```
o: GROUP ATOMS=1,4,7,11,14
h: GROUP ATOMS=2,3,5,6,8,9,12,13
```

and then include it in the main 'plumed.dat' file

```
INCLUDE FILE=groups.dat
# compute the coordination among the two groups
c: COORDINATION GROUPA=o GROUPB=h R_0=0.3
# print the coordination on file 'colvar'
PRINT ARG=c FILE=colvar
```

(see also INCLUDE, COORDINATION, and PRINT). The groups.dat file could be very long and include lists of thousand atoms without cluttering the main plumed.dat file.

A GROMACS index file can also be imported

```
# import group named 'protein' from file index.ndx
pro: GROUP NDX_FILE=index.ndx NDX_GROUP=protein
# dump all the atoms of the protein on a trajectory file
DUMPATOMS ATOMS=pro FILE=traj.gro
```

(see also DUMPATOMS)

5.1.4 MOLINFO

	This is part of the setup module

This command is used to provide information on the molecules that are present in your system.

The information on the molecules in your system can either be provided in the form of a pdb file or as a set of lists of atoms that describe the various chains in your system. If a pdb file is used plumed the MOLINFO command will endeavor to recognize the various chains and residues that make up the molecules in your system using the chain↩ IDs and resnumbers from the pdb file. You can then use this information in later commands to specify atom lists in terms residues. For example using this command you can find the backbone atoms in your structure automatically.

Please be aware that the pdb parser in plumed is far from perfect. You should thus check the log file and examine what plumed is actually doing whenenver you use the MOLINFO action.

Using MOLINFO with a protein's pdb extend the possibility of atoms selection using the @ special symbol. Current registered keywords are:

```
@phi-#
@psi-#
@omega-#
@chi1-#
```

that select the appropriate atoms that define each dihedral angle for residue #.

Bug At the moment the HA1 atoms in a GLY residues are treated as if they are the CB atoms. This may or may not be true - GLY is problematic for secondary structure residues as it is achiral.

Bug If you use WHOLEMOLECULES RESIDUES=1-10 for a 18 amino acid protein (18 amino acids + 2 terminal groups = 20 residues) the code will fail as it will not be able to interpret termnal residue 1.

The atoms involved can be specified using

CHAIN	(for masochists (mostly Davide Branduardi)) The atoms involved in each of the chains of interest in the structure.. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

STRUCTURE	a file in pdb format containing a reference structure. This is used to defines the atoms in the various residues, chains, etc . For more details on the PDB file format visit http://www.wwpdb.org/docs.html
MOLTYPE	(default=protein) what kind of molecule is contained in the pdb file

Examples

In the following example the MOLINFO command is used to provide the information on which atoms are in the backbone of a protein to the ALPHARMSD CV.

```
MOLINFO STRUCTURE=reference.pdb
ALPHARMSD RESIDUES=all TYPE=DRMSD LESS_THAN={RATIONAL R_0=0.08 NN=8 MM=12} LABEL=a
```

(see also ALPHARMSD)

5.1.5 WHOLEMOLECULES

	This is part of the generic module

This action is used to rebuild molecules that can become split by the periodic boundary conditions.

It is similar to the ALIGN_ATOMS keyword of plumed1, and is needed since some MD dynamics code (e.g. GRO↩ MACS) can break molecules during the calculation.

Running some CVs without this command can cause there to be discontinuities changes in the CV value and artifacts in the calculations. This command can be applied more than once. To see what effect is has use a variable without pbc or use the DUMPATOMS directive to output the atomic positions.

Attention

> This directive modifies the stored position at the precise moment it is executed. This means that only collective variables which are below it in the input script will see the corrected positions. As a general rule, put it at the top of the input file. Also, unless you know exactly what you are doing, leave the default stride (1), so that this action is performed at every MD step.

The way WHOLEMOLECULES modifies each of the listed entities is this:

- First atom of the list is left in place
- Each atom of the list is shifted by a lattice vectors so that it becomes as close as possible to the previous one, iteratively.

In this way, if an entity consists of a list of atoms such that consecutive atoms in the list are always closer than half a box side the entity will become whole. This can be usually achieved selecting consecute atoms (1-100), but it is also possible to skip some atoms, provided consecute chosen atoms are close enough.

The atoms involved can be specified using

ENTITY	the atoms that make up a molecule that you wish to align. To specify multiple molecules use a list of ENTITY keywords: ENTITY0, ENTITY1,... You can use multiple instances of this keyword i.e. ENTITY1, ENTITY2, ENTITY3...

Or alternatively by using

RESIDUES	this command specifies that the backbone atoms in a set of residues all must be aligned. It must be used in tandem with the MOLINFO action and the MOLTYPE keyword. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers

Compulsory keywords

STRIDE	(default=1) the frequency with which molecules are reassembled. Unless you are completely certain about what you are doing leave this set equal to 1!

MOLTYPE	the type of molecule that is under study. This is used to define the backbone atoms

Examples

This command instructs plumed to reconstruct the molecule containing atoms 1-20 at every step of the calculation and dump them on a file.

```
# to see the effect, one could dump the atoms as they were before molecule reconstruction:
# DUMPATOMS FILE=dump-broken.xyz ATOMS=1-20
WHOLEMOLECULES ENTITY0=1-20
DUMPATOMS FILE=dump.xyz ATOMS=1-20
```

(see also DUMPATOMS)

This command instructs plumed to reconstruct two molecules containing atoms 1-20 and 30-40

```
WHOLEMOLECULES ENTITY0=1-20 ENTITY1=30-40
DUMPATOMS FILE=dump.xyz ATOMS=1-20,30-40
```

(see also DUMPATOMS)

This command instructs plumed to reconstruct the chain of backbone atoms in a protein

```
MOLINFO STRUCTURE=helix.pdb
WHOLEMOLECULES RESIDUES=all MOLTYPE=protein
```

(See also MOLINFO)

5.1.6 FIT_TO_TEMPLATE

	This is part of the generic module

This action is used to align a molecule to a template.

This can be used to move the coordinates stored in plumed so as to be aligned with a provided template in pdb format. Pdb should contain also weights for alignment (see the format of pdb files used e.g. for RMSD). Weights for displacement are ignored, since no displacement is computed here. Notice that all atoms (not only those in the template) are aligned. To see what effect try the DUMPATOMS directive to output the atomic positions.

Also notice that PLUMED propagate forces correctly so that you can add a bias on a CV computed after alignment. For many CVs this has no effect, but in some case the alignment can change the result. Examples are:

- POSITION CV since it is affected by a rigid shift of the system.
- DISTANCE CV with COMPONENTS. Since the alignment could involve a rotation (with TYPE=OPTIMAL) the actual components could be different from the original ones.
- CELL components for a similar reason.

In the present implementation only TYPE=SIMPLE is implemented. As a consequence, only POSITION CV can be affected by the fit.

Attention

This directive modifies the stored position at the precise moment it is executed. This means that only collective variables which are below it in the input script will see the corrected positions. As a general rule, put it at the top of the input file. Also, unless you know exactly what you are doing, leave the default stride (1), so that this action is performed at every MD step.

Compulsory keywords

STRIDE	(default=1) the frequency with which molecules are reassembled. Unless you are completely certain about what you are doing leave this set equal to 1!
REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
TYPE	(default=SIMPLE) the manner in which RMSD alignment is performed. Should be OPTIMAL or SIMPLE. Currently only SIMPLE is implemented

Examples

Align the atomic position to a template then print them

```
# to see the effect, one could dump the atoms before alignment
DUMPATOMS FILE=dump-before.xyz ATOMS=1-20
FIT_TO_TEMPLATE STRIDE=1 REFERENCE=ref.pdb TYPE=SIMPLE
DUMPATOMS FILE=dump-after.xyz ATOMS=1-20
```

(see also DUMPATOMS)

5.1.7 CENTER

	This is part of the vatom module

Calculate the center for a group of atoms, with arbitrary weights.

The computed center is stored as a virtual atom that can be accessed in an atom list through the label for the CENTER action that creates it. Notice that the generated virtual atom has charge equal to the sum of the charges and mass equal to the sum of the masses. If used with the MASS flag, then it provides a result identical to COM.

When running with periodic boundary conditions, the user should take care that the atoms in the COM group actually are in the proper periodic image. This is typically achieved using the WHOLEMOLECULES action before COM calculation.

The atoms involved can be specified using

ATOMS	the list of atoms which are involved the virtual atom's definition. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

MASS	(default=off) If set center is mass weighted
WEIGHTS	Center is computed as a weighted average.

Examples

```
# a point which is on the line connecting atoms 1 and 10, so that its distance
# from 10 is twice its distance from 1:
c1: CENTER ATOMS=1,1,10
# this is another way of stating the same:
c1bis: CENTER ATOMS=1,10 WEIGHTS=2,1

# center of mass among these atoms:
c2: CENTER ATOMS=2,3,4,5 MASS

d1: DISTANCE ATOMS=c1,c2

PRINT ARG=d1
```

(See also DISTANCE, COM and PRINT).

5.1.8 COM

	This is part of the vatom module

Calculate the center of mass for a group of atoms.

The computed center of mass is stored as a virtual atom that can be accessed in an atom list through the label for the COM action that creates it.

For arbitrary weights (e.g. geometric center) see CENTER.

When running with periodic boundary conditions, the user should take care that the atoms in the COM group actually are in the proper periodic image. This is typically achieved using the WHOLEMOLECULES action before COM calculation.

The atoms involved can be specified using

ATOMS	the list of atoms which are involved the virtual atom's definition. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Examples

The following input instructs plumed to print the distance between the center of mass for atoms 1,2,3,4,5,6,7 and that for atoms 15,20:

```
COM ATOMS=1-7          LABEL=c1
COM ATOMS=15,20        LABEL=c2
DISTANCE ATOMS=c1,c2   LABEL=d1
PRINT ARG=d1
```

(See also DISTANCE and PRINT).

5.1.9 GHOST

	This is part of the vatom module

Calculate the absolute position of a ghost atom with fixed coordinates in the local reference frame formed by three atoms. The computed ghost atom is stored as a virtual atom that can be accessed in an atom list through the the label for the GHOST action that creates it.

The atoms involved can be specified using

ATOMS	the list of atoms which are involved the virtual atom's definition. For more information on how to specify lists of atoms see Groups and Virtual Atoms
COORDINATES	coordinates of the ghost atom in the local reference frame. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Examples

The following input instructs plumed to print the distance between the ghost atom and the center of mass for atoms 15,20:

```
GHOST ATOMS=1,5,10 COORDINATES=10.0,10.0,10.0 LABEL=c1
COM ATOMS=15,20        LABEL=c2
DISTANCE ATOMS=c1,c2   LABEL=d1
PRINT ARG=d1
```

(See also DISTANCE and PRINT).

5.2 CV Documentation

The following list contains descriptions of a number of the colvars that are currently implemented in PLUMED 2.

ALPHABETA	Measures a distance including pbc between the instantaneous values of a set of torsional angles and set of reference values.

ALPHARMSD	Probe the alpha helical content of a protein structure.
ANGLE	Calculate an angle.
ANTIBETARMSD	Probe the antiparallel beta sheet content of your protein structure.
CELL	Calculate the components of the simulation cell
CH3SHIFTS	This collective variable calculates a scoring function based on the comparison of calculated andexperimental methyl chemical shifts.
CONSTANT	Return a constant quantity.
CONTACTMAP	Calculate the distances between a number of pairs of atoms and transform each distance by a switching function.The transformed distance can be compared with a reference value in order to calculate the squared distancebetween two contact maps. Each distance can also be weighted for a given value. CONTACTMAP can be used togetherwith FUNCPATHMSD to define a path in the contactmap space.
COORDINATION	Calculate coordination numbers.
CS2BACKBONE	This collective variable calculates a scoring function based on the comparison of backcalculated andexperimental backbone chemical shifts for a protein (CA, CB, C', H, HA, N).
DHENERGY	Calculate Debye-Huckel interaction energy among GROUPA and GROUPB.
DIHCOR	Measures the degree of similarity between dihedral angles.
DIPOLE	Calculate the dipole moment for a group of atoms.
DISTANCE	Calculate the distance between a pair of atoms.
ENERGY	Calculate the total energy of the simulation box.
FAKE	This is a fake colvar container used by cltools or various other actionsand just support input and period definition
GPROPERTYMAP	Property maps but with a more flexible framework for the distance metric being used.
GYRATION	Calculate the radius of gyration, or other properties related to it.
NOE	Calculates the deviation of current distances from experimental NOE derived distances.
PARABETARMSD	Probe the parallel beta sheet content of your protein structure.
PATHMSD	This Colvar calculates path collective variables.
PATH	Path collective variables with a more flexible framework for the distance metric being used.
POSITION	Calculate the components of the position of an atom.
PROPERTYMAP	Calculate generic property maps.
RDC	Calculates the Residual Dipolar Coupling between two atoms.
TEMPLATE	This file provides a template for if you want to introduce a new CV.

TORSION	Calculate a torsional angle.
VOLUME	Calculate the volume of the simulation box.

5.2.1 ALPHABETA

	This is part of the multicolvar module

Measures a distance including pbc between the instantaneous values of a set of torsional angles and set of reference values.

This colvar calculates the following quantity.

$$s = \frac{1}{2}\sum_i \left[1+\cos(\phi_i - \phi_i^{\mathrm{Ref}})\right]$$

where the ϕ_i values are the instantaneous values for the TORSION angles of interest. The ϕ_i^{Ref} values are the user-specified reference values for the torsional angles.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Compulsory keywords

REFERENCE	the reference values for each of the torsional angles. If you use a single REFERENCE value the same reference value is used for all torsions You can use multiple instances of this keyword i.e. REFERENCE1, REFERENCE2, REFERENCE3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements

VERBOSE	(default=off) write a more detailed output

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

Examples

The following provides an example of the input for an alpha beta similarity.

```
ALPHABETA ...
ATOMS1=168,170,172,188 REFERENCE1=3.14
ATOMS2=170,172,188,190 REFERENCE2=3.14
ATOMS3=188,190,192,230 REFERENCE3=3.14
LABEL=ab
... ALPHABETA
PRINT ARG=ab FILE=colvar STRIDE=10
```

Because all the reference values are the same we can calculate the same quantity using

```
ALPHABETA ...
ATOMS1=168,170,172,188 REFERENCE=3.14
ATOMS2=170,172,188,190
ATOMS3=188,190,192,230
LABEL=ab
... ALPHABETA
PRINT ARG=ab FILE=colvar STRIDE=10
```

Writing out the atoms involved in all the torsions in this way can be rather tedious. Thankfully if you are working with protein you can avoid this by using the MOLINFO command. PLUMED uses the pdb file that you provide to this command to learn about the topology of the protein molecule. This means that you can specify torsion angles using the following syntax:

```
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
ALPHABETA ...
ATOMS1=@phi-3 REFERENCE=3.14
ATOMS2=@psi-3
ATOMS3=@phi-4
LABEL=ab
... ALPHABETA
PRINT ARG=ab FILE=colvar STRIDE=10
```

Here, @phi-3 tells plumed that you would like to calculate the ϕ angle in the third residue of the protein. Similarly @psi-4 tells plumed that you want to calculate the ψ angle of the 4th residue of the protein.

5.2.2 ALPHARMSD

	This is part of the secondarystructure module

Probe the alpha helical content of a protein structure.

Any chain of six contiguous residues in a protein chain can form an alpha helix. This colvar thus generates the set of all possible six residue sections and calculates the RMSD distance between the configuration in which the residues find themselves and an idealized alpha helical structure. These distances can be calculated by either aligning the

instantaneous structure with the reference structure and measuring each atomic displacement or by calculating differences between the set of interatomic distances in the reference and instantaneous structures.

This colvar is based on the following reference [3]. The authors of this paper use the set of distances from the alpha helix configurations to measure the number of segments that have an alpha helical configuration. This is done by calculating the following sum of functions of the rmsd distances:

$$s = \sum_i \frac{1-\left(\frac{r_i-d_0}{r_0}\right)^n}{1-\left(\frac{r_i-d_0}{r_0}\right)^m}$$

where the sum runs over all possible segments of alpha helix. By default the NN, MM and D_0 parameters are set equal to those used in [3]. The R_0 parameter must be set by the user - the value used in [3] was 0.08 nm.

If you change the function in the above sum you can calculate quantities such as the average distance from a purely the alpha helical configuration or the distance between the set of residues that is closest to an alpha helix and the reference configuration. To do these sorts of calculations you can use the AVERAGE and MIN keywords. In addition you can use the LESS_THAN keyword if you would like to change the form of the switching function. If you use any of these options you no longer need to specify NN, R_0, MM and D_0.

Please be aware that for codes like gromacs you must ensure that plumed reconstructs the chains involved in your CV when you calculate this CV using anthing other than TYPE=DRMSD. For more details as to how to do this see WHOLEMOLECULES.

Description of components

By default this Action calculates the number of structural units that are within a certain distance of a idealised secondary structure element. This quantity can then be referenced elsewhere in the input by using the label of the action. However, thes Action can also be used to calculate the following quantities by using the keywords as described below. The quantities then calculated can be referened using the label of the action followed by a dot and then the name from the table below. Please note that you can use the LESS_THAN keyword more than once. The resulting components will be labelled *label*.lessthan-1, *label*.lessthan-2 and so on unless you exploit the fact that these labels are customizable. In particular, by using the LABEL keyword in the description of you LESS_THAN function you can set name of the component that you are calculating

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.

The atoms involved can be specified using

RESIDUES	this command is used to specify the set of residues that could conceivably form part of the secondary structure. It is possible to use residues numbers as the various chains and residues should have been identified else using an instance of the MOLINFO action. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers. Please be aware that to form secondary structure elements your chain must contain at least N residues, where N is dependent on the particular secondary structure you are interested in. As such if you define portions of the chain with fewer than N residues the code will crash.

Compulsory keywords

TYPE	(default=DRMSD) the manner in which RMSD alignment is performed. Should be OPTIMAL, SIMPLE or DRMSD. For more details on the OPTIMAL and SIMPLE methods see RMSD. For more details on the DRMSD method see DRMSD.
R_0	The r_0 parameter of the switching function.
D_0	(default=0.0) The d_0 parameter of the switching function
NN	(default=8) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
VERBOSE	(default=off) write a more detailed output
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.

Examples

The following input calculates the number of six residue segments of protein that are in an alpha helical configuration.

```
MOLINFO STRUCTURE=helix.pdb
ALPHARMSD RESIDUES=all TYPE=DRMSD LESS_THAN={RATIONAL R_0=0.08 NN=8 MM=12} LABEL=a
```

(see also MOLINFO)

5.2.3 ANGLE

	This is part of the colvar module

Calculate an angle.

This command can be used to compute the angle between three atoms. Alternatively if four atoms appear in the atom specification it calculates the angle between two vectors identified by two pairs of atoms.

If *three* atoms are given, the angle is defined as:

$$\theta = \arccos\left(\frac{\mathbf{r}_{21}\cdot\mathbf{r}_{23}}{|\mathbf{r}_{21}||\mathbf{r}_{23}|}\right)$$

Here $\mathbf{r}_{ij}$ is the distance vector among the i-th and the j-th listed atom.

If *four* atoms are given, the angle is defined as:

$$\theta = \arccos\left(\frac{\mathbf{r}_{21}\cdot\mathbf{r}_{34}}{|\mathbf{r}_{21}||\mathbf{r}_{34}|}\right)$$

Notice that angles defined in this way are non-periodic variables and their value is limited by definition between 0 and π.

The vectors $\mathbf{r}_{ij}$ are by default evaluated taking periodic boundary conditions into account. This behavior can be changed with the NOPBC flag.

The atoms involved can be specified using

ATOMS	the list of atoms involved in this collective variable (either 3 or 4 atoms). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

Examples

This command tells plumed to calculate the angle between the vector connecting atom 1 to atom 2 and the vector connecting atom 2 to atom 3 and to print it on file COLVAR1. At the same time, the angle between vector connecting atom 1 to atom 2 and the vector connecting atom 3 to atom 4 is printed on file COLVAR2.

```
a: ANGLE ATOMS=1,2,3
# equivalently one could state:
# a: ANGLE ATOMS=1,2,2,3

b: ANGLE ATOMS=1,2,3,4

PRINT ARG=a FILE=COLVAR1
PRINT ARG=b FILE=COLVAR2
```

(see also PRINT)

5.2.4 ANTIBETARMSD

	This is part of the secondarystructure module

Probe the antiparallel beta sheet content of your protein structure.

Two protein segments containing three continguous residues can form an antiparallel beta sheet. Although if the two segments are part of the same protein chain they must be separated by a minimum of 2 residues to make room for the turn. This colvar thus generates the set of all possible six residue sections that could conceivably form an antiparallel beta sheet and calculates the RMSD distance between the configuration in which the residues find themselves and an idealized antiparallel beta sheet structure. These distances can be calculated by either aligning the instantaneous structure with the reference structure and measuring each atomic displacement or by calculating differences between the set of interatomic distances in the reference and instantaneous structures.

This colvar is based on the following reference [3]. The authors of this paper use the set of distances from the anti parallel beta sheet configurations to measure the number of segments that have an configuration that resemebles an anti paralel beta sheet. This is done by calculating the following sum of functions of the rmsd distances:

$$s = \sum_i \frac{1-\left(\frac{r_i-d_0}{r_0}\right)^n}{1-\left(\frac{r_i-d_0}{r_0}\right)^m}$$

where the sum runs over all possible segments of antiparallel beta sheet. By default the NN, MM and D_0 parameters are set equal to those used in [3]. The R_0 parameter must be set by the user - the value used in [3] was 0.08 nm.

If you change the function in the above sum you can calculate quantities such as the average distance from a purely configuration composed of pure anti-parallel beta sheets or the distance between the set of residues that is closest

to an anti-parallel beta sheet and the reference configuration. To do these sorts of calculations you can use the AVERAGE and MIN keywords. In addition you can use the LESS_THAN keyword if you would like to change the form of the switching function. If you use any of these options you no longer need to specify NN, R_0, MM and D_0.

Please be aware that for codes like gromacs you must ensure that plumed reconstructs the chains involved in your CV when you calculate this CV using anthing other than TYPE=DRMSD. For more details as to how to do this see WHOLEMOLECULES.

Description of components

By default this Action calculates the number of structural units that are within a certain distance of a idealised secondary structure element. This quantity can then be referenced elsewhere in the input by using the label of the action. However, thes Action can also be used to calculate the following quantities by using the keywords as described below. The quantities then calculated can be referened using the label of the action followed by a dot and then the name from the table below. Please note that you can use the LESS_THAN keyword more than once. The resulting components will be labelled *label*.lessthan-1, *label*.lessthan-2 and so on unless you exploit the fact that these labels are customizable. In particular, by using the LABEL keyword in the description of you LESS_THAN function you can set name of the component that you are calculating

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.

The atoms involved can be specified using

RESIDUES	this command is used to specify the set of residues that could conceivably form part of the secondary structure. It is possible to use residues numbers as the various chains and residues should have been identified else using an instance of the MOLINFO action. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers. Please be aware that to form secondary structure elements your chain must contain at least N residues, where N is dependent on the particular secondary structure you are interested in. As such if you define portions of the chain with fewer than N residues the code will crash.

Compulsory keywords

TYPE	(default=DRMSD) the manner in which RMSD alignment is performed. Should be OPTIMAL, SIMPLE or DRMSD. For more details on the OPTIMAL and SIMPLE methods see RMSD. For more details on the DRMSD method see DRMSD.
R_0	The r_0 parameter of the switching function.
D_0	(default=0.0) The d_0 parameter of the switching function
NN	(default=8) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
STYLE	(default=all) Antiparallel beta sheets can either form in a single chain or from a pair of chains. If STYLE=all all chain configuration with the appropriate geometry are counted. If STYLE=inter only sheet-like configurations involving two chains are counted, while if STYLE=intra only sheet-like configurations involving a single chain are counted

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
VERBOSE	(default=off) write a more detailed output
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...

MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
STRANDS_CUTOFF	If in a segment of protein the two strands are further apart then the calculation of the actual RMSD is skipped as the structure is very far from being beta-sheet like. This keyword speeds up the calculation enormously when you are using the LESS_THAN option. However, if you are using some other option, then this cannot be used

Examples

The following input calculates the number of six residue segments of protein that are in an antiparallel beta sheet configuration.

```
MOLINFO STRUCTURE=helix.pdb
ANTIBETARMSD RESIDUES=all TYPE=DRMSD LESS_THAN={RATIONAL R_0=0.08 NN=8 MM=12} LABEL=a
```

(see also MOLINFO)

5.2.5 CELL

	This is part of the colvar module

Calculate the components of the simulation cell

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
ax	the ax component of the cell matrix
ay	the ay component of the cell matrix
az	the az component of the cell matrix
bx	the bx component of the cell matrix
by	the by component of the cell matrix
bz	the bz component of the cell matrix
cx	the cx component of the cell matrix
cy	the cy component of the cell matrix
cz	the cz component of the cell matrix

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input tells plumed to print the squared modulo of each of the three lattice vectors

```
cell: CELL
aaa:    COMBINE ARG=cell.ax,cell.ay,cell.az POWERS=2,2,2 PERIODIC=NO
bbb:    COMBINE ARG=cell.bx,cell.by,cell.bz POWERS=2,2,2 PERIODIC=NO
ccc:    COMBINE ARG=cell.cx,cell.cy,cell.cz POWERS=2,2,2 PERIODIC=NO
PRINT ARG=aaa,bbb,ccc
```

(See also COMBINE and PRINT).

5.2.6 CH3SHIFTS

	This is part of the colvar module

This collective variable calculates a scoring function based on the comparison of calculated and experimental methyl chemical shifts.

CH3Shift [4] is employed to back calculate the chemical shifts of methyl groups (ALA:HB; ILE:HD,HG2; LEU:H↩D1,HD2; THR:HG2; VAL:HG1,HG2) that are then compared with a set of experimental values to generate a score [5] [6].

It is also possible to backcalculate the chemical shifts from multiple replicas and then average them to perform Replica-Averaged Restrained MD simulations [7] [8].

In general the system for which chemical shifts are to be calculated must be completly included in ATOMS. It should also be made whole WHOLEMOLECULES before the the back-calculation.

HOW TO COMPILE IT

Installing PLUMED with ALMOST on how to compile PLUMED with ALMOST.

HOW TO USE IT

CH3Shift reads from a text file the experimental chemical shifts:

```
CH3shifts.dat:
1.596 28
0.956 46
0.576 3 HG2
0.536 3 HD1
0.836 13 HG2
0.666 13 HD1
0.716 23 HG2
0.506 23 HD1
```

A template.pdb file is needed to the generate a topology of the protein within ALMOST. For histidines in protonation states different from D the HIE/HIP name should be used in the template.pdb. GLH and ASH can be used for the alternative protonation of GLU and ASP. Non-standard amino acids and other molecules are not yet supported! If multiple chains are present the chain identifier must be in the standard PDB format, together with the TER keyword at the end of each chain. Residues numbering should always go from 1 to N in both the chemical shifts files as well as in the template.pdb file. Two more keywords can be used to setup the topology: CYS-DISU to tell ALMOST to look for disulphide bridges and TERMINI to define the protonation state of the the chain's termini (currently only DEFAULT (NH3+, CO2-) and NONE (NH, CO)).

One more standard file is also needed, that is an ALMOST force-field file, corresponding to the force-field family used in the MD code, (a03_cs2_gromacs.mdb for the amber family or all22_gromacs.mdb for the charmm family).

All the above files must be in a single folder that must be specified with the keyword DATA (multiple definition of the CV can point to different folders).

Examples

case 1:

```
WHOLEMOLECULES ENTITY0=1-174
cs: CH3SHIFTS ATOMS=1-174 DATA=data/ FF=a03_gromacs.mdb FLAT=0.0 NRES=13 ENSEMBLE
cse: RESTRAINT ARG=cs SLOPE=24 KAPPA=0 AT=0.
PRINT ARG=cs,cse.bias
```

case 2:

```
WHOLEMOLECULES ENTITY0=1-174
cs: CH3SHIFTS ATOMS=1-174 DATA=data/ FF=a03_gromacs.mdb FLAT=1.0 NRES=13 TERMINI=DEFAULT,NONE CYS-DISU WRITE_C
PRINT ARG=cs
```

(See also WHOLEMOLECULES, RESTRAINT and PRINT)

5.2.7 CONSTANT

	This is part of the colvar module

Return a constant quantity.

Useful in combination with functions.

Compulsory keywords

VALUE	The value of the constant

Examples

The following input instructs plumed to compute the distance between atoms 1 and 2. If this distance is between 1.0 and 2.0, it is printed. If it is lower than 1.0 (larger than 2.0), 1.0 (2.0) is printed

```
one: CONSTANT VALUE=1.0
two: CONSTANT VALUE=2.0
dis: DISTANCE ATOMS=1,2
sss: SORT ARG=one,dis,two
PRINT ARG=sss.2
```

(See also DISTANCE, SORT, and PRINT).

5.2.8 CONTACTMAP

	This is part of the colvar module

Calculate the distances between a number of pairs of atoms and transform each distance by a switching function. The transformed distance can be compared with a reference value in order to calculate the squared distance between two contact maps. Each distance can also be weighted for a given value. CONTACTMAP can be used together with FUNCPATHMSD to define a path in the contactmap space.

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to be used to calculate the following quantities by employing the keywords listed below. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
contact	By not using SUM or CMDIST each contact will be stored in a component

The atoms involved can be specified using

ATOMS	the atoms involved in each of the contacts you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one contact will be calculated for each ATOM keyword you specify. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Compulsory keywords

SWITCH	The switching functions to use for each of the contacts in your map. You can either specify a global switching function using SWITCH or one switching function for each contact. Details of the various switching functions you can use are provided on switchingfunction. You can use multiple instances of this keyword i.e. SWITCH1, SWITCH2, SWITCH3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SUM	(default=off) calculate the sum of all the contacts in the input
CMDIST	(default=off) calculate the distance with respect to the provided reference contant map
SERIAL	(default=off) Perform the calculation in serial - for debug purpose

REFERENCE	A reference value for a given contact, by default is 0.0 You can either specify a global reference value using REFERENCE or one reference value for each contact. You can use multiple instances of this keyword i.e. REFERENCE1, REFERENCE2, REFERENCE3...
WEIGHT	A weight value for a given contact, by default is 1.0 You can either specify a global weight value using WEIGHT or one weight value for each contact. You can use multiple instances of this keyword i.e. WEIGHT1, WEIGHT2, WEIGHT3...

Examples

The following example calculates switching functions based on the distances between atoms 1 and 2, 3 and 4 and 4 and 5. The values of these three switching functions are then output to a file named colvar.

```
CONTACTMAP ATOMS1=1,2 ATOMS2=3,4 ATOMS3=4,5 ATOMS4=5,6 SWITCH=(RATIONAL R_0=1.5) LABEL=f1
PRINT ARG=f1.* FILE=colvar
```

The following example calculates the difference of the current contact map with respect to a reference provided. In this case REFERENCE is the fraction of contact that is formed (i.e. the distance between two atoms transformed with the SWITH), while R_0 is the contact distance. WEIGHT gives the relative weight of each contact to the final distance measure.

```
CONTACTMAP ...
ATOMS1=1,2 REFERENCE1=0.1 WEIGHT1=0.5
ATOMS2=3,4 REFERENCE2=0.5 WEIGHT2=1.0
ATOMS3=4,5 REFERENCE3=0.25 WEIGHT3=1.0
ATOMS4=5,6 REFERENCE4=0.0 WEIGHT4=0.5
SWITCH=(RATIONAL R_0=1.5)
LABEL=cmap
CMDIST
... CONTACTMAP

PRINT ARG=cmap FILE=colvar
```

(See also PRINT)

5.2.9 COORDINATION

	This is part of the colvar module

Calculate coordination numbers.

This keyword can be used to calculate the number of contacts between two groups of atoms and is defined as

$$\sum_{i\in A}\sum_{i\in B} s_{ij}$$

where s_{ij} is 1 if the contact between atoms i and j is formed, zero otherwise. In practise, s_{ij} is replaced with a switching function to make it differentiable. The default switching function is:

$$s_{ij} = \frac{1-\left(\frac{\mathbf{r}_{ij}-d_0}{r_0}\right)^n}{1-\left(\frac{\mathbf{r}_{ij}-d_0}{r_0}\right)^m}$$

but it can be changed using the optional SWITCH option.

To make your calculation faster you can use a neighbor list, which makes it that only a relevant subset of the pairwise distance are calculated at every step.

If GROUPB is empty, it will sum the $\frac{N(N-1)}{2}$ pairs in GROUPA. This avoids computing twice permuted indexes (e.g. pair (i,j) and (j,i)) thus running at twice the speed.

Notice that if there are common atoms between GROUPA and GROUPB the switching function should be equal to one. These "self contacts" are discarded by plumed (since version 2.1), so that they actually count as "zero".

The atoms involved can be specified using

GROUPA	First list of atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	Second list of atoms (if empty, N∗(N-1)/2 pairs in GROUPA are counted). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose
PAIR	(default=off) Pair only 1st element of the 1st group with 1st element in the second, etc
NLIST	(default=off) Use a neighbour list to speed up the calculation

NL_CUTOFF	The cutoff for the neighbour list
NL_STRIDE	The frequency with which we are updating the atoms in the neighbour list
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

Examples

The following example instructs plumed to calculate the total coordination number of the atoms in group 1-10 with the atoms in group 20-100. For atoms 1-10 coordination numbers are calculated that count the number of atoms from the second group that are within 0.3 nm of the central atom. A neighbour list is used to make this calculation faster, this neighbour list is updated every 100 steps.

```
COORDINATION GROUPA=1-10 GROUPB=20-100 R_0=0.3 NLIST NL_CUTOFF=0.5 NL_STRIDE=100
```

The following is a dummy example which should compute the value 0 because the self interaction of atom 1 is skipped. Notice that in plumed 2.0 "self interactions" were not skipped, and the same calculation should return 1.

```
c: COORDINATION GROUPA=1 GROUPB=1 R_0=0.3
PRINT ARG=c STRIDE=10

c1: COORDINATION GROUPA=1-10 GROUPB=1-10 R_0=0.3
x: COORDINATION GROUPA=1-10 R_0=0.3
c2: COMBINE ARG=x COEFFICIENTS=2
# the two variables c1 and c2 should be identical, but the calculation of c2 is twice faster
# since it runs on half of the pairs. Notice that to get the same result you
# should double it
PRINT ARG=c1,c2 STRIDE=10
```

See also PRINT and COMBINE

5.2.10 CS2BACKBONE

	This is part of the colvar module

This collective variable calculates a scoring function based on the comparison of backcalculated and experimental backbone chemical shifts for a protein (CA, CB, C', H, HA, N).

CamShift [9] is employed to back calculate the chemical shifts that are then compared with a set of experimental values to generate a score [5] [6].

It is also possible to back-calculate the chemical shifts from multiple replicas and then average them to perform Replica-Averaged Restrained MD simulations [7] [8].

In general the system for which chemical shifts are to be calculated must be completly included in ATOMS. It should also be made whole WHOLEMOLECULES before the the back-calculation.

HOW TO COMPILE IT

Installing PLUMED with ALMOST on how to compile PLUMED with ALMOST.

HOW TO USE IT

To use CamShift a set of experimental data is needed. CamShift uses backbone and Cb chemical shifts that must be provided as text files:

```
CAshifts.dat:
CBshifts.dat:
Cshifts.dat:
Hshifts.dat:
HAshifts.dat:
Nshifts.dat:
#1 0.0
2 55.5
3 58.4
.
.
#last 0.0
#last+1 (first) of second chain
.
#last of second chain
```

All of them must always be there. If a chemical shift for an atom of a residue is not available 0.0 must be used. So if for example all the Cb are not available all the chemical shifts for all the residues should be set as 0.0.

A template.pdb file is needed to the generate a topology of the protein within ALMOST. For histidines in protonation states different from D the HIE/HIP name should be used in the template.pdb. GLH and ASH can be used for the alternative protonation of GLU and ASP. Non-standard amino acids and other molecules are not yet supported! If multiple chains are present the chain identifier must be in the standard PDB format, together with the TER keyword at the end of each chain. Residues numbering should always go from 1 to N in both the chemical shifts files as well as in the template.pdb file. Two more keywords can be used to setup the topology: CYS-DISU to tell ALMOST to look for disulphide bridges and TERMINI to define the protonation state of the the chain's termini (currently only DEFAULT (NH3+, CO2-) and NONE (NH, CO)).

Two more standard files are also needed: an ALMOST force-field file, corresponding to the force-field family used in the MD code, (a03_cs2_gromacs.mdb for the amber family or all22_gromacs.mdb for the charmm family) and camshift.db, both these files can be copied from almost/branches/almost-2.1/toppar.

All the above files must be in a single folder that must be specified with the keyword DATA.

Additional material and examples can be also found in the tutorial Belfast tutorial: NMR constraints

Examples

case 1:

```
WHOLEMOLECULES ENTITY0=1-174
cs: CS2BACKBONE ATOMS=1-174 DATA=data/ FF=a03_gromacs.mdb FLAT=0.0 NRES=13 ENSEMBLE
cse: RESTRAINT ARG=cs SLOPE=24 KAPPA=0 AT=0.
PRINT ARG=cs,cse.bias
```

case 2:

```
WHOLEMOLECULES ENTITY0=1-174
cs: CS2BACKBONE ATOMS=1-174 DATA=data/ FF=a03_gromacs.mdb FLAT=1.0 NRES=13 TERMINI=DEFAULT,NONE CYS-DISU WRITE
PRINT ARG=cs
```

(See also WHOLEMOLECULES, RESTRAINT and PRINT)

5.2.11 DHENERGY

	This is part of the colvar module

Calculate Debye-Huckel interaction energy among GROUPA and GROUPB.

This variable calculates the electrostatic interaction among GROUPA and GROUPB using a Debye-Huckel approximation defined as

$$\frac{1}{4\pi\varepsilon_r\varepsilon_0}\sum_{i\in A}\sum_{j\in B}q_iq_j\frac{e^{-\kappa|\mathbf{r}_{ij}|}}{|\mathbf{r}_{ij}|}$$

This collective variable can be used to analyze or induce electrostatically driven reactions [10]. Notice that the value of the DHENERGY is returned in plumed units (see UNITS).

If GROUPB is empty, it will sum the N∗(N-1)/2 pairs in GROUPA. This avoids computing twice permuted indexes (e.g. pair (i,j) and (j,i)) thus running at twice the speed.

Notice that if there are common atoms between GROUPA and GROUPB their interaction is discarded.

The atoms involved can be specified using

GROUPA	First list of atoms. For more information on how to specify lists of atoms see Groups and Virtual Atoms
GROUPB	Second list of atoms (if empty, N∗(N-1)/2 pairs in GROUPA are counted). For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

I	(default=1.0) Ionic strength (M)
TEMP	(default=300.0) Simulation temperature (K)
EPSILON	(default=80.0) Dielectric constant of solvent

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) Perform the calculation in serial - for debug purpose

PAIR	(default=off) Pair only 1st element of the 1st group with 1st element in the second, etc
NLIST	(default=off) Use a neighbour list to speed up the calculation

NL_CUTOFF	The cutoff for the neighbour list
NL_STRIDE	The frequency with which we are updating the atoms in the neighbour list

Examples

```
# this is printing the electrostatic interaction between two groups of atoms
dh: DHENERGY GROUPA=1-10 GROUPB=11-20 EPSILON=80.0 I=0.1 TEMP=300.0
PRINT ARG=dh
```

(see also PRINT)

5.2.12 DIHCOR

	This is part of the multicolvar module

Measures the degree of similarity between dihedral angles.

This colvar calculates the following quantity.

$$s = \frac{1}{2}\sum_i [1+\cos(\phi_i - \psi_i)]$$

where the ϕ_i and ψ values and the instantaneous values for the TORSION angles of interest.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize

LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

Examples

The following provides an example input for the dihcor action

```
DIHCOR ...
  ATOMS1=1,2,3,4,5,6,7,8
  ATOMS2=5,6,7,8,9,10,11,12
  LABEL=dih
... DIHCOR
PRINT ARG=dih FILE=colvar STRIDE=10
```

In the above input we are calculating the correation between the torsion angle involving atoms 1, 2, 3 and 4 and the torsion angle involving atoms 5, 6, 7 and 8. This is then added to the correlation betwene the torsion angle involving atoms 5, 6, 7 and 8 and the correlation angle involving atoms 9, 10, 11 and 12.

Writing out the atoms involved in all the torsions in this way can be rather tedious. Thankfully if you are working with protein you can avoid this by using the MOLINFO command. PLUMED uses the pdb file that you provide to this command to learn about the topology of the protein molecule. This means that you can specify torsion angles using the following syntax:

```
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
DIHCOR ...
ATOMS1=@phi-3,@psi-3
ATOMS2=@psi-3,@phi-4
ATOMS4=@phi-4,@psi-4
... DIHCOR
PRINT ARG=dih FILE=colvar STRIDE=10
```

Here, @phi-3 tells plumed that you would like to calculate the ϕ angle in the third residue of the protein. Similarly @psi-4 tells plumed that you want to calculate the ψ angle of the 4th residue of the protein.

5.2.13 DIPOLE

	This is part of the colvar module

Calculate the dipole moment for a group of atoms.

The atoms involved can be specified using

GROUP	the group of atoms we are calculating the dipole moment for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following tells plumed to calculate the dipole of the group of atoms containing the atoms from 1-10 and print it every 5 steps

```
d: DIPOLE GROUP=1-10
PRINT FILE=output STRIDE=5 ARG=5
```

(see also PRINT)

Attention

If the total charge Q of the group in non zero, then a charge Q/N will be subtracted to every atom, where N is the number of atoms. This implies that the dipole (which for a charged system depends on the position) is computed on the geometric center of the group.

5.2.14 DISTANCE

	This is part of the colvar module

Calculate the distance between a pair of atoms.

By default the distance is computed taking into account periodic boundary conditions. This behavior can be changed with the NOPBC flag. Moreover, single components in cartesian space (x,y, and z, with COMPONENTS) or single components projected to the three lattice vectors (a,b, and c, with SCALED_COMPONENTS) can be also computed.

Notice that Cartesian components will not have the proper periodicity! If you have to study e.g. the permeation of a molecule across a membrane, better to use SCALED_COMPONENTS.

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to be used to calculate the following quantities by employing the keywords listed below. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Keyword	Description
x	**COMPONENTS**	the x-component of the vector connecting the two atoms
y	**COMPONENTS**	the y-component of the vector connecting the two atoms
z	**COMPONENTS**	the z-component of the vector connecting the two atoms
a	**SCALED_COMPONENTS**	the normalized projection on the first lattice vector of the vector connecting the two atoms
b	**SCALED_COMPONENTS**	the normalized projection on the second lattice vector of the vector connecting the two atoms

c	**SCALED_COMPONENTS**	the normalized projection on the third lattice vector of the vector connecting the two atoms

The atoms involved can be specified using

ATOMS	the pair of atom that we are calculating the distance between. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COMPONENTS	(default=off) calculate the x, y and z components of the distance separately and store them as label.x, label.y and label.z
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the distance separately and store them as label.a, label.b and label.c

Examples

The following input tells plumed to print the distance between atoms 3 and 5, the distance between atoms 2 and 4 and the x component of the distance between atoms 2 and 4.

```
d1:  DISTANCE ATOMS=3,5
d2:  DISTANCE ATOMS=2,4
d2c: DISTANCE ATOMS=2,4 COMPONENTS
PRINT ARG=d1,d2,d2c.x
```

(See also PRINT).

The following input computes the end-to-end distance for a polymer of 100 atoms and keeps it at a value around 5.

```
WHOLEMOLECULES ENTITY0=1-100
e2e: DISTANCE ATOMS=1,100 NOPBC
RESTRAINT ARG=e2e KAPPA=1 AT=5
```

(See also WHOLEMOLECULES and RESTRAINT).

Notice that NOPBC is used to be sure that if the end-to-end distance is larger than half the simulation box the distance is compute properly. Also notice that, since many MD codes break molecules across cell boundary, it might be necessary to use the WHOLEMOLECULES keyword (also notice that it should be *before* distance). The list of atoms provided to WHOLEMOLECULES here contains all the atoms between 1 and 100. Strictly speaking, this is not necessary. If you know for sure that atoms with difference in the index say equal to 10 are *not* going to be farther than half cell you can e.g. use

```
WHOLEMOLECULES ENTITY0=1,10,20,30,40,50,60,70,80,90,100
e2e: DISTANCE ATOMS=1,100 NOPBC
RESTRAINT ARG=e2e KAPPA=1 AT=5
```

Just be sure that the ordered list provide to WHOLEMOLECULES has the following properties:

- Consecutive atoms should be closer than half-cell throughout the entire simulation.

- Atoms required later for the distance (e.g. 1 and 100) should be included in the list

The following example shows how to take into account periodicity e.g. in z-component of a distance

```
# this is a center of mass of a large group
c: COM ATOMS=1-100
# this is the distance between atom 101 and the group
d: DISTANCE ATOMS=c,101 COMPONENTS
# this makes a new variable, dd, equal to d and periodic, with domain -10,10
# this is the right choise if e.g. the cell is orthorombic and its size in
# z direction is 20.
dz: COMBINE ARG=d.z PERIODIC=-10,10
# metadynamics on dd
METAD ARG=dz SIGMA=0.1 HEIGHT=0.1 PACE=200
```

(see also COM, COMBINE, and METAD)

Using SCALED_COMPONENTS this problem should not arise because they are always periodic with domain (-0.5,+0.5).

5.2.15 ENERGY

	This is part of the colvar module

Calculate the total energy of the simulation box.

Total energy can be biased with umbrella sampling [11] or with well tempered metadynamics [12].

Notice that this CV could be unavailable with some MD code. When it is available, and when also replica exchange is available, metadynamics applied to ENERGY can be used to decrease the number of required replicas.

Bug Acceptance for replica exchange when ENERGY is biased is computed correctly only of all the replicas has the same potential energy function. This is for instance not true when using GROMACS with lambda replica exchange of with plumed-hrex branch.

Examples

The following input instructs plumed to print the energy of the system

```
ENERGY LABEL=ene
PRINT ARG=ene
```

(See also PRINT).

5.2.16 FAKE

	This is part of the colvar module

This is a fake colvar container used by cltools or various other actions and just support input and period definition

The atoms involved can be specified using

ATOMS	the fake atom index, a number is enough. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO,NO (one for the lower and the other for the upper boundary). For multicomponents then it is PERIODI↩ C=mincomp1,maxcomp1,mincomp2,maxcomp2 etc

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

COMPONENTS	additional componnets that this variable is supposed to have. Periodicity is ruled by PERIODIC keyword

Examples

```
FAKE ATOMS=1 PERIODIC=-3.14,3.14   LABEL=d2
```

(See also PRINT).

5.2.17 GPROPERTYMAP

	This is part of the mapping module

Property maps but with a more flexible framework for the distance metric being used.

This colvar calculates a property map using the formalism developed by Spiwork [13]. In essence if you have the value of some property, X_i, that it takes at a set of high-dimensional positions then you calculate the value of the property at some arbitrary point in the high-dimensional space using:

$$X = \frac{\sum_i X_i * \exp(-\lambda D_i(x))}{\sum_i \exp(-\lambda D_i(x))}$$

Within PLUMED there are multiple ways to define the distance from a high-dimensional configuration, D_i. You could calculate the RMSD distance or you could calculate the ammount by which a set of collective variables change. As such this implementation of the propertymap allows one to use all the different distance metric that are discussed in Distances from reference configurations. This is as opposed to the alternative implementation PROPERTYMAP which is a bit faster but which only allows one to use the RMSD distance.

Compulsory keywords

REFERENCE	a pdb file containing the set of reference configurations

PROPERTY	the property to be used in the index. This should be in the REMARK of the reference
TYPE	(default=OPTIMAL) the manner in which distances are calculated. More information on the different metrics that are available in PLUMED can be found in the section of the manual on Distances from reference configurations
LAMBDA	the value of the lambda parameter for paths

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
DISABLE_CHECKS	(default=off) disable checks on reference input structures.
NOZPATH	(default=off) do not calculate the zpath position
NOMAPPING	(default=off) do not calculate the position on the manifold

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

Examples

5.2.18 GYRATION

	This is part of the colvar module

Calculate the radius of gyration, or other properties related to it.

The different properties can be calculated and selected by the TYPE keyword: the Radius of Gyration (RADIUS); the Trace of the Gyration Tensor (TRACE); the Largest Principal Moment of the Gyration Tensor (GTPC_1); the middle Principal Moment of the Gyration Tensor (GTPC_2); the Smallest Principal Moment of the Gyration Tensor (GTPC_3); the Asphericiry (ASPHERICITY); the Acylindricity (ACYLINDRICITY); the Relative Shape Anisotropy (KAPPA2); the Smallest Principal Radius Of Gyration (GYRATION_3); the Middle Principal Radius of Gyration (GYRATION_2); the Largest Principal Radius of Gyration (GYRATION_1). A derivation of all these different variants can be found in [14]

The radius of gyration is calculated using:

$$s_{\mathrm{Gyr}} = \left(\frac{\sum_i^n m_i |r_i - r_{\mathrm{COM}}|^2}{\sum_i^n m_i} \right)^{1/2}$$

with the position of the center of mass r_{COM} given by:

$$r_{\mathrm{COM}} = \frac{\sum_i^n r_i \, m_i}{\sum_i^n m_i}$$

The radius of gyration is calculated without applying periodic boundary conditions so the atoms used for the calculation should all be part of the same molecule that should be made whole before calculating the cv, WHOLEMOLECULES.

The atoms involved can be specified using

ATOMS	the group of atoms that you are calculating the Gyration Tensor for. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TYPE	(default=RADIUS) The type of calculation relative to the Gyration Tensor you want to perform

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOT_MASS_WEIGHTED	(default=off) set the masses of all the atoms equal to one

Examples

The following input tells plumed to print the radius of gyration of the chain containing atoms 10 to 20.

```
WHOLEMOLECULES ENTITY0=10-20
GYRATION TYPE=RADIUS ATOMS=10-20 LABEL=rg
PRINT ARG=rg STRIDE=1 FILE=colvar
```

(See also PRINT)

5.2.19 NOE

	This is part of the colvar module

Calculates the deviation of current distances from experimental NOE derived distances.

NOE distances are calculated between couple of atoms, averaging over equivalent couples, and compared with a set of reference distances. Distances can also be averaged over multiple replicas to perform replica-averaged simulations. Each NOE is defined by two groups containing the same number of atoms and by a reference distance, distances are calculated in pairs.

$$NOE() = \sum_{i}^{noes} ((\frac{1}{N_{eq}} \sum_{j}^{N_{eq}} (\frac{1}{r_j^6}))^{\frac{-1}{6}} - d_i^{exp})^2$$

Reference distances can also be considered as upper limits only, in this case the sum is over a half parabola.

The atoms involved can be specified using

GROUPA	the atoms involved in each of the contacts you wish to calculate. Keywords like GROUPA1, GROUPA2, GROUPA3,... should be listed and one contact will be calculated for each ATOM keyword you specify. You can use multiple instances of this keyword i.e. GROUPA1, GROUPA2, GROUPA3...
GROUPB	the atoms involved in each of the contacts you wish to calculate. Keywords like GROUPB1, GROUPB2, GROUPB3,... should be listed and one contact will be calculated for each ATOM keyword you specify. You can use multiple instances of this keyword i.e. GROUPB1, GROUPB2, GROUPB3...

Compulsory keywords

WRITE_NOE	(default=0) Write the back-calculated chemical shifts every # steps.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
UPPER_LIMITS	(default=off) Set to TRUE if you want to consider the reference distances as upper limits.
ENSEMBLE	(default=off) Set to TRUE if you want to average over multiple replicas.
SERIAL	(default=off) Perform the calculation in serial - for debug purpose

NOEDIST	A compulsory reference distance for a given NOEYou can either specify a global reference value using NOEDIST or one reference value for each contact. You can use multiple instances of this keyword i.e. NOEDIST1, NOEDIST2, NOEDIST3...

Examples

In the following examples three noes are defined, the first is calculated based on the distances of atom 1-2 and 3-2; the second is defined by the distance 5-7 and the third by the distances 4-15,4-16,8-15,8-16.

```
NOE ...
GROUPA1=1,3 GROUPB1=2,2 NOEDIST1=0.5
GROUPA2=5 GROUPB2=7 NOEDIST2=0.4
GROUPA3=4,4,8,8 GROUPB3=15,16,15,16 NOEDIST3=0.3
LABEL=noes
... NOE

PRINT ARG=noes FILE=colvar
```

(See also PRINT)

5.2.20 PARABETARMSD

	This is part of the secondarystructure module

Probe the parallel beta sheet content of your protein structure.

Two protein segments containing three continguous residues can form a parallel beta sheet. Although if the two segments are part of the same protein chain they must be separated by a minimum of 3 residues to make room for the turn. This colvar thus generates the set of all possible six residue sections that could conceivably form a parallel beta sheet and calculates the RMSD distance between the configuration in which the residues find themselves and an idealized parallel beta sheet structure. These distances can be calculated by either aligning the instantaneous structure with the reference structure and measuring each atomic displacement or by calculating differences between the set of interatomic distances in the reference and instantaneous structures.

This colvar is based on the following reference [3]. The authors of this paper use the set of distances from the parallel beta sheet configurations to measure the number of segments whose configuration resembles a parallel beta sheet. This is done by calculating the following sum of functions of the rmsd distances:

$$s = \sum_i \frac{1-\left(\frac{r_i-d_0}{r_0}\right)^n}{1-\left(\frac{r_i-d_0}{r_0}\right)^m}$$

where the sum runs over all possible segments of parallel beta sheet. By default the NN, MM and D_0 parameters are set equal to those used in [3]. The R_0 parameter must be set by the user - the value used in [3] was 0.08 nm.

If you change the function in the above sum you can calculate quantities such as the average distance from a structure composed of only parallel beta sheets or the distance between the set of residues that is closest to a parallel beta sheet and the reference configuration. To do these sorts of calculations you can use the AVERAGE and MIN keywords. In addition you can use the LESS_THAN keyword if you would like to change the form of the switching function. If you use any of these options you no longer need to specify NN, R_0, MM and D_0.

Please be aware that for codes like gromacs you must ensure that plumed reconstructs the chains involved in your CV when you calculate this CV using anthing other than TYPE=DRMSD. For more details as to how to do this see WHOLEMOLECULES.

Description of components

By default this Action calculates the number of structural units that are within a certain distance of a idealised secondary structure element. This quantity can then be referenced elsewhere in the input by using the label of the action. However, thes Action can also be used to calculate the following quantities by using the keywords as described below. The quantities then calculated can be referened using the label of the action followed by a dot and then the name from the table below. Please note that you can use the LESS_THAN keyword more than once. The resulting components will be labelled *label*.lessthan-1, *label*.lessthan-2 and so on unless you exploit the fact that these labels are customizable. In particular, by using the LABEL keyword in the description of you LESS_THAN function you can set name of the component that you are calculating

Quantity	Keyword	Description
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.

The atoms involved can be specified using

RESIDUES	this command is used to specify the set of residues that could conceivably form part of the secondary structure. It is possible to use residues numbers as the various chains and residues should have been identified else using an instance of the MOLINFO action. If you wish to use all the residues from all the chains in your system you can do so by specifying all. Alternatively, if you wish to use a subset of the residues you can specify the particular residues you are interested in as a list of numbers. Please be aware that to form secondary structure elements your chain must contain at least N residues, where N is dependent on the particular secondary structure you are interested in. As such if you define portions of the chain with fewer than N residues the code will crash.

Compulsory keywords

TYPE	(default=DRMSD) the manner in which RMSD alignment is performed. Should be OPTIMAL, SIMPLE or DRMSD. For more details on the OPTIMAL and SIMPLE methods see RMSD. For more details on the DRMSD method see DRMSD.
R_0	The r_0 parameter of the switching function.
D_0	(default=0.0) The d_0 parameter of the switching function
NN	(default=8) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
STYLE	(default=all) Parallel beta sheets can either form in a single chain or from a pair of chains. If STYLE=all all chain configuration with the appropriate geometry are counted. If STYLE=inter only sheet-like configurations involving two chains are counted, while if STYLE=intra only sheet-like configurations involving a single chain are counted

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

VERBOSE	(default=off) write a more detailed output
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
STRANDS_CUTOFF	If in a segment of protein the two strands are further apart then the calculation of the actual RMSD is skipped as the structure is very far from being beta-sheet like. This keyword speeds up the calculation enormously when you are using the LESS_THAN option. However, if you are using some other option, then this cannot be used

Examples

The following input calculates the number of six residue segments of protein that are in an parallel beta sheet configuration.

```
MOLINFO STRUCTURE=helix.pdb
PARABETARMSD RESIDUES=all TYPE=DRMSD LESS_THAN={RATIONAL R_0=0.08 NN=8 MM=12} LABEL=a
```

(see also MOLINFO)

5.2.21 PATHMSD

	This is part of the colvar module

This Colvar calculates path collective variables.

This is the Path Collective Variables implementation (see [15]). This variable computes the progress along a given set of frames that is provided in input ("sss" component) and the distance from them ("zzz" component). (see below).

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
sss	the position on the path
zzz	the distance from the path

Compulsory keywords

LAMBDA	the lambda parameter is needed for smoothing, is in the units of plumed
REFERENCE	the pdb is needed to provide the various milestones

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

NEIGH_SIZE	size of the neighbor list
NEIGH_STRIDE	how often the neighbor list needs to be calculated in time units

Examples

Here below is a case where you have defined three frames and you want to calculate the progress along the path and the distance from it in p1

```
p1: PATHMSD REFERENCE=file.pdb  LAMBDA=500.0 NEIGH_STRIDE=4 NEIGH_SIZE=8
PRINT ARG=p1.sss,p1.zzz STRIDE=1 FILE=colvar FMT=%8.4f
```

note that NEIGH_STRIDE=4 NEIGH_SIZE=8 control the neighborlist parameter (optional but recommended for performance) and states that the neighbor list will be calculated every 4 timesteps and consider only the closest 8 member to the actual md snapshots.

In the REFERENCE PDB file the frames must be separated either using END or ENDMDL.

Note

The implementation of this collective variable and of PROPERTYMAP is shared, as well as most input options.

5.2.22 PATH

	This is part of the mapping module

Path collective variables with a more flexible framework for the distance metric being used.

The Path Collective Variables developed by Branduardi and co-workers [15] allow one to compute the progress along a high-dimensional path and the distance from the high-dimensional path. The progress along the path (s) is computed using:

$$s = \frac{\sum_{i=1}^{N} i \exp(-\lambda R[X - X_i])}{\sum_{i=1}^{N} \exp(-\lambda R[X - X_i])}$$

while the distance from the path (z) is measured using:

$$z = -\frac{1}{\lambda} \ln \left[\sum_{i=1}^{N} \exp(-\lambda R[X - X_i]) \right]$$

In these expressions N high-dimensional frames (X_i) are used to describe the path in the high-dimensional space. The two expressions above are then functions of the distances from each of the high-dimensional frames $R[X - X_i]$. Within PLUMED there are multiple ways to define the distance from a high-dimensional configuration. You could calculate the RMSD distance or you could calculate the ammount by which a set of collective variables change. As such this implementation of the path cv allows one to use all the difference distance metrics that are discussed in Distances from reference configurations. This is as opposed to the alternative implementation of path (PATHMSD) which is a bit faster but which only allows one to use the RMSD distance.

Compulsory keywords

REFERENCE	a pdb file containing the set of reference configurations
TYPE	(default=OPTIMAL) the manner in which distances are calculated. More information on the different metrics that are available in PLUMED can be found in the section of the manual on Distances from reference configurations
LAMBDA	the value of the lambda parameter for paths

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
DISABLE_CHECKS	(default=off) disable checks on reference input structures.
NOZPATH	(default=off) do not calculate the zpath position
NOSPATH	(default=off) do not calculate the spath position

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

Examples

5.2.23 POSITION

	This is part of the colvar module

Calculate the components of the position of an atom.

Notice that single components will not have the proper periodicity! If you need the values to be consistent through PBC you should use SCALED_COMPONENTS, which defines values that by construction are in the -0.5,0.5 domain. This is similar to the equivalent flag for DISTANCE. Also notice that by default the minimal image distance from the origin is considered (can be changed with NOPBC).

Attention

This variable should be used with extreme care since it allows to easily go into troubles. See comments below.

This variable can be safely used only if Hamiltonian is not invariant for translation (i.e. there are other absolute positions which are biased, e.g. by position restraints) and cell size and shapes are fixed through the simulation.

If you are not in this situation and still want to use the absolute position of an atom you should first fix the reference frame. This can be done e.g. using FIT_TO_TEMPLATE.

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
x	the x-component of the atom position
y	the y-component of the atom position
z	the z-component of the atom position

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
a	**SCALED_COMPONENTS**	the normalized projection on the first lattice vector of the atom position
b	**SCALED_COMPONENTS**	the normalized projection on the second lattice vector of the atom position
c	**SCALED_COMPONENTS**	the normalized projection on the third lattice vector of the atom position

The atoms involved can be specified using

ATOM	the atom number. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SCALED_COMPONENTS	(default=off) calculate the a, b and c scaled components of the position separately and store them as label.a, label.b and label.c

Examples

```
# align to a template
FIT_TO_TEMPLATE REFERENCE=ref.pdb
p: POSITION ATOM=3
PRINT ARG=p.x,p.y,p.z
```

(see also FIT_TO_TEMPLATE)

5.2.24 PROPERTYMAP

	This is part of the colvar module

Calculate generic property maps.

This Colvar calculates the property maps according to the work of Spiwok [13].

Basically it calculates

$$X = \frac{\sum_i X_i * \exp(-\lambda D_i(x))}{\sum_i \exp(-\lambda D_i(x))} \tag{5.1}$$

$$Y = \frac{\sum_i Y_i * \exp(-\lambda D_i(x))}{\sum_i \exp(-\lambda D_i(x))} \tag{5.2}$$

$$\cdots \tag{5.3}$$

$$zzz = -\frac{1}{\lambda}\log(\sum_i \exp(-\lambda D_i(x))) \tag{5.4}$$

where the parameters X_i and Y_i are provided in the input pdb (allv.pdb in this case) and $D_i(x)$ is the MSD after optimal alignment calculated on the pdb frames you input (see Kearsley).

Description of components

The names of the components in this action can be customized by the user in the actions input file. However, in addition to these customizable components the following quantities will always be output

Quantity	Description
zzz	the minimum distance from the reference points

Compulsory keywords

LAMBDA	the lambda parameter is needed for smoothing, is in the units of plumed

REFERENCE	the pdb is needed to provide the various milestones
PROPERTY	the property to be used in the indexing: this goes in the REMARK field of the reference

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

NEIGH_SIZE	size of the neighbor list
NEIGH_STRIDE	how often the neighbor list needs to be calculated in time units

Examples

```
p3: PROPERTYMAP REFERENCE=../../trajectories/path_msd/allv.pdb PROPERTY=X,Y LAMBDA=69087 NEIGH_SIZE=8 NEIG
PRINT ARG=p3.X,p3.Y,p3.zzz STRIDE=1 FILE=colvar FMT=%8.4f
```

note that NEIGH_STRIDE=4 NEIGH_SIZE=8 control the neighborlist parameter (optional but recommended for performance) and states that the neighbor list will be calculated every 4 timesteps and consider only the closest 8 member to the actual md snapshots.

In this case the input line instructs plumed to look for two properties X and Y with attached values in the REMARK line of the reference pdb (Note: No spaces from X and = and 1 !!!!). e.g.

```
REMARK X=1 Y=2
ATOM      1  CL  ALA     1      -3.171   0.295   2.045  1.00  1.00
ATOM      5  CLP ALA     1      -1.819  -0.143   1.679  1.00  1.00
.......
END
REMARK X=2 Y=3
ATOM      1  CL  ALA     1      -3.175   0.365   2.024  1.00  1.00
ATOM      5  CLP ALA     1      -1.814  -0.106   1.685  1.00  1.00
....
END
```

Note

The implementation of this collective variable and of PATHMSD is shared, as well as most input options.

5.2.25 RDC

	This is part of the colvar module

Calculates the Residual Dipolar Coupling between two atoms.

The RDC between two atomic nuclei depends on the θ angle between the inter-nuclear vector and the external magnetic field. In isotropic media RDCs average to zero because of the orientational averaging, but when the rotational symmetry is broken, either through the introduction of an alignment medium or for molecules with highly anisotropic paramagnetic susceptibility, RDCs become measurable.

$$D = D_{max} 0.5(3\cos^2(\theta) - 1)$$

where

$$D_{max} = -\mu_0 \gamma_1 \gamma_2 h/(8\pi^3 r^3)$$

that is the maximal value of the dipolar coupling for the two nuclear spins with gyromagnetic ratio γ. μ is the magnetic constant and h is the Planck constant.

Common Gyromagnetic Ratios (C.G.S)

- H(1) 26.7513
- C(13) 6.7261
- N(15) -2.7116
- NH -72.5388
- CH 179.9319
- CN -18.2385
- CC 45.2404

This collective variable calculates the Residual Dipolar Coupling for a set of couple of atoms using the above definition. From the calculated RDCs and a set of experimental values it calculates either their correlation or the squared quality factor

$$Q^2 = \frac{\sum_i (D_i - D_i^{exp})^2}{\sum_i (D_i^{exp})^2}$$

RDCs report only on the fraction of molecules that is aligned, this means that comparing the RDCs from a single structure in a MD simulation to the experimental values is not particularly meaningfull, from this point of view it is better to compare their correlation. The fraction of aligned molecules can be obtained by maximising the correlation between the calculated and the experimental RDCs. This fraction can be used as a scaling factor in the calculation of the RDCs in order to compare their values. The averaging of the RDCs calculated with the above definition from a standard MD should result to 0 because of the rotational diffusion, but this variable can be used to break the rotational symmetry.

RDCs can also be calculated using a Single Value Decomposition approach, in this case the code rely on the a set of function from the GNU Scientific Library (GSL). (With SVD forces are not currently implemented).

Replica-Averaged restrained simulations can be performed with this CV using the ENSEMBLE flag.

Additional material and examples can be also found in the tutorial Belfast tutorial: NMR constraints

The atoms involved can be specified using

ATOMS	the couple of atoms involved in each of the bonds for which you wish to calculate the RDC. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one dipolar coupling will be calculated for each ATOMS keyword you specify. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Compulsory keywords

WRITE_DC	(default=0) Write the back-calculated dipolar couplings every # steps.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
ENSEMBLE	(default=off) Set to TRUE if you want to average over multiple replicas.
CORRELATION	(default=off) Set to TRUE if you want to kept constant the bonds distances.
SERIAL	(default=off) Set to TRUE if you want to run the CV in serial.
SVD	(default=off) Set to TRUE if you want to backcalculate using Single Value Decomposition (need GSL at compilation time).

COUPLING	Add an experimental value for each coupling. You can use multiple instances of this keyword i.e. COUPLING1, COUPLING2, COUPLING3...
GYROM	Add the product of the gyromagnetic constants for each bond. You can use multiple instances of this keyword i.e. GYROM1, GYROM2, GYROM3...
SCALE	Add a scaling factor to take into account concentration and other effects. You can use multiple instances of this keyword i.e. SCALE1, SCALE2, SCALE3...
BONDLENGTH	Set a fixed length for for the bonds distances. You can use multiple instances of this keyword i.e. BONDLENGTH1, BONDLENGTH2, BONDLENGTH3...

Examples

In the following example five N-H RDCs are defined and their correlation with respect to a set of experimental data is calculated.

```
RDC ...
GYROM=-72.5388
SCALE=1.0
CORRELATION
ATOMS1=20,21 COUPLING1=8.17
ATOMS2=37,38 COUPLING2=-8.271
ATOMS3=56,57 COUPLING3=-10.489
ATOMS4=76,77 COUPLING4=-9.871
ATOMS5=92,93 COUPLING5=-9.152
LABEL=nh
... RDC

rdce: RESTRAINT ARG=nh KAPPA=0. SLOPE=-25000.0 AT=1.

PRINT ARG=nh,rdce.bias FILE=colvar
```

(See also PRINT, RESTRAINT)

5.2.26 TEMPLATE

	This is part of the colvar module

This file provides a template for if you want to introduce a new CV.

The atoms involved can be specified using

TEMPLATE_INPUT	the keyword with which you specify what atoms to use should be added like this. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

TEMPLATE_COMPULSORY	all compulsory keywords should be added like this with a description here

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
TEMPLATE_DEFAULT_OFF_FLAG	(default=off) flags that are by default not performed should be specified like this
TEMPLATE_DEFAULT_ON_FLAG	(default=on) flags that are by default performed should be specified like this

TEMPLATE_OPTIONAL	all optional keywords that have input should be added like a description here

Examples

5.2.27 TORSION

	This is part of the colvar module

Calculate a torsional angle.

This command can be used to compute the torsion between four atoms or alternatively to calculate the angle between two vectors projected on the plane orthogonal to an axis.

The atoms involved can be specified using

ATOMS	the four atoms involved in the torsional angle

Or alternatively by using

AXIS	two atoms that define an axis. You can use this to find the angle in the plane perpendicular to the axis between the vectors specified using the VECTOR1 and VECTOR2 keywords.

VECTOR1	two atoms that define a vector. You can use this in combination with VECTOR2 and AXIS
VECTOR2	two atoms that define a vector. You can use this in combination with VECTOR1 and AXIS

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
COSINE	(default=off) calculate cosine instead of dihedral

Examples

This input tells plumed to print the torsional angle between atoms 1, 2, 3 and 4 on file COLVAR.

```
t: TORSION ATOMS=1,2,3,4
# this is an alternative, equivalent, definition:
# t: TORSION VECTOR1=2,1 AXIS=2,3 VECTOR2=3,4
PRINT ARG=t FILE=COLVAR
```

If you are working with a protein you can specify the special named torsion angles ϕ, ψ, ω and χ_1 by using TORSION in combination with the MOLINFO command. This can be done by using the following syntax.

```
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
t1: TORSION ATOMS=@phi-3
t2: TORSION ATOMS=@psi-4
PRINT ARG=t1,t2 FILE=colvar STRIDE=10
```

Here, @phi-3 tells plumed that you would like to calculate the ϕ angle in the third residue of the protein. Similarly @psi-4 tells plumed that you want to calculate the ψ angle of the 4th residue of the protein.

5.2.28 VOLUME

	This is part of the colvar module

Calculate the volume of the simulation box.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input tells plumed to print the volume of the system

```
VOLUME LABEL=vol
PRINT ARG=vol
```

(See also PRINT).

5.3 Distances from reference configurations

One colvar that has been shown to be very sucessful in studying protein folding is the distance between the instantaneous configuration and a reference configuration - often the structure of the folded state. When the free energy of a protein is shown as a function of this collective variable there is a minima for low values of the CV, which is due to the folded state of the protein. There is then a second minima at higher values of the CV, which is the minima corresponding to the unfolded state.

A slight problem with this sort of collective variable is that there are many different ways of calculating the distance from a particular reference structure. The simplest - adding together the distances by which each of the atoms has been translated in going from the reference configuration to the instantanous configuration - is not particularly sensible. A distance calculated in this way does not neglect translation of the center of mass of the molecule and rotation of the frame of reference. A common practise is thus to remove these components by calculating the RMSD distance between the reference and instantaneous configurations. This is not the only way to calculate the distance, however. One could also calculate the total ammount by which a large number of collective variables change in moving from the reference to the instaneous configurations. One could even combine RMSD distances with the ammount the collective variables change. A full list of the ways distances can be measured in PLUMED is given below:

DRMSD	Calculate the distance RMSD with respect to a reference structure.
MULTI-RMSD	Calculate the RMSD distance moved by a number of separated domains from their positions in a reference structure.
RMSD	Calculate the RMSD with respect to a reference structure.
TARGET	This function measures the pythagorean distance from a particular structure measured in the space defined by some set of collective variables.

These options for calculating distances are re-used in a number of places in the code. For instance they are used in some of the analysis algorithms that are implemented in PLUMED and in PATH collective variables.

5.3.1 DRMSD

	This is part of the colvar module

Calculate the distance RMSD with respect to a reference structure.

To calculate the root-mean-square deviation between the atoms in two configurations you must first superimpose the two structures in some ways. Obviously, it is the internal vibrational motions of the structure - i.e. not the translations and rotations - that are interesting. However, aligning two structures by removing the translational and rotational motions is not easy. Furthermore, in some cases there can be alignment issues caused by so-called frame-fitting problems. It is thus often cheaper and easier to calculate the distances between all the pairs of atoms. The distance between the two structures, $\mathbf{X}^a$ and $\mathbf{X}^b$ can then be measured as:

$$d(\mathbf{X}^A,\mathbf{X}^B) = \frac{1}{N(N-1)} \sum_{i \neq j} [d(\mathbf{x}_i^a,\mathbf{x}_j^a) - d(\mathbf{x}_i^b,\mathbf{x}_j^b)]^2$$

where N is the number of atoms and $d(\mathbf{x}_i,\mathbf{x}_j)$ represents the distance between atoms i and j. Clearly, this representation of the configuration is invariant to translation and rotation. However, it can become expensive to calculate when the number of atoms is large. This can be resolved within the DRMSD colvar by setting LOWER_CUT↩ OFF and UPPER_CUTOFF. These keywords ensure that only pairs of atoms that are within a certain range are incorporated into the above sum.

In PDB files the atomic coordinates and box lengths should be in Angstroms unless you are working with natural units. If you are working with natural units then the coordinates should be in your natural length unit. For more details on the PDB file format visit http://www.wwpdb.org/docs.html

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
LOWER_CUTOFF	only pairs of atoms further than LOWER_CUTOFF are considered in the calculation.
UPPER_CUTOFF	only pairs of atoms closer than UPPER_CUTOFF are considered in the calculation.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

Examples

The following tells plumed to calculate the distance RMSD between the positions of the atoms in the reference file and their instantaneous position. Only pairs of atoms whose distance in the reference structure is within 0.1 and 0.8 nm are considered.

```
DRMSD REFERENCE=file.pdb LOWER_CUTOFF=0.1 UPPER_CUTOFF=0.8
```

...

5.3.2 MULTI-RMSD

	This is part of the colvar module

Calculate the RMSD distance moved by a number of separated domains from their positions in a reference structure.

When you have large proteins the calculation of the root mean squared deviation between all the atoms in a reference structure and the instantaneous configuration becomes prohibitively expensive. You may thus instead want to calculate the RMSD between the atoms in a set of domains of your protein and your reference structure. That is to say:

$$d(X,X_r) = \sqrt{\sum_i w_i |X_i - X_i'|^2}$$

where here the sum is over the domains of the protein, X_i represents the positions of the atoms in domain i in the instantaneous configuration and X_i' is the positions of the atoms in domain i in the reference configuration. w_i is an optional weight.

The distances for each of the domains in the above sum can be calculated using the DRMSD or RMSD measures or using a combination of these distance. The reference configuration is specified in a pdb file like the one below:

```
ATOM      2  O   ALA     2      -0.926  -2.447  -0.497  1.00  1.00      DIA  O
ATOM      4  HNT ALA     2       0.533  -0.396   1.184  1.00  1.00      DIA  H
ATOM      6  HT1 ALA     2      -0.216  -2.590   1.371  1.00  1.00      DIA  H
ATOM      7  HT2 ALA     2      -0.309  -1.255   2.315  1.00  1.00      DIA  H
ATOM      8  HT3 ALA     2      -1.480  -1.560   1.212  1.00  1.00      DIA  H
ATOM      9  CAY ALA     2      -0.096   2.144  -0.669  1.00  1.00      DIA  C
```

```
ATOM      10  HY1 ALA     2       0.871   2.385  -0.588  1.00  1.00      DIA  H
TER
ATOM      12  HY3 ALA     2      -0.520   2.679  -1.400  1.00  1.00      DIA  H
ATOM      14  OY  ALA     2      -1.139   0.931  -0.973  1.00  1.00      DIA  O
ATOM      16  HN  ALA     2       1.713   1.021  -0.873  1.00  1.00      DIA  H
ATOM      18  HA  ALA     2       0.099  -0.774  -2.218  1.00  1.00      DIA  H
ATOM      19  CB  ALA     2       2.063  -1.223  -1.276  1.00  1.00      DIA  C
ATOM      20  HB1 ALA     2       2.670  -0.716  -2.057  1.00  1.00      DIA  H
ATOM      21  HB2 ALA     2       2.556  -1.051  -0.295  1.00  1.00      DIA  H
ATOM      22  HB3 ALA     2       2.070  -2.314  -1.490  1.00  1.00      DIA  H
END
```

with the TER keyword being used to separate the various domains in you protein.

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
TYPE	(default=MULTI-SIMPLE) the manner in which RMSD alignment is performed. Should be MULTI-OPTIMAL, MULTI-OPTIMAL-FAST, MULTI-SIMPLE or MULTI-DRMSD.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SQUARED	(default=off) This should be setted if you want MSD instead of RMSD

Examples

The following tells plumed to calculate the RMSD distance between the positions of the atoms in the reference file and their instantaneous position. The Kearseley algorithm for each of the domains.

```
MULTI-RMSD REFERENCE=file.pdb TYPE=MULTI-OPTIMAL
```

The following tells plumed to calculate the RMSD distance btween the positions of the atoms in the domains of reference the reference structure and their instantaneous positions. Here distances are calculated using the DR↩ MSD measure.

```
MULTI-RMSD REFERENCE=file.pdb TYPE=MULTI-DRMSD
```

...

5.3.3 RMSD

	This is part of the colvar module

Calculate the RMSD with respect to a reference structure.

The aim with this colvar it to calculate something like:

$$d(X,X') = |X - X'|$$

where X is the instantaneous position of all the atoms in the system and X' is the positions of the atoms in some reference structure provided as input. $d(X,X')$ thus measures the distance all the atoms have moved away from this reference configuration. Oftentimes, it is only the internal motions of the structure - i.e. not the translations of the center of mass or the rotations of the reference frame - that are interesting. Hence, when calculating the the root-mean-square deviation between the atoms in two configurations you must first superimpose the two structures in some way. At present PLUMED provides two distinct ways of performing this superposition. The first method is applied when you use TYPE=SIMPLE in the input line. This instruction tells PLUMED that the root mean square deviation is to be calculated after the positions of the geometric centers in the reference and instantaneous configurations are aligned. In other words $d(X,x')$ is to be calculated using:

$$d(X,X') = \sqrt{\sum_i \sum_\alpha^{x,y,z} \frac{w_i}{\sum_j w_j} (X_{i,\alpha} - com_\alpha(X) - X'_{i,\alpha} + com_\alpha(X'))^2}$$

with

$$com_\alpha(X) = \sum_i \frac{w'_i}{\sum_j w'_j} X_{i,\alpha}$$

and

$$com_\alpha(X') = \sum_i \frac{w'_i}{\sum_j w'_j} X'_{i,\alpha}$$

Obviously, $com_\alpha(X)$ and $com_\alpha(X')$ represent the positions of the center of mass in the reference and instantaneous configurations if the weights w' are set equal to the atomic masses. If the weights are all set equal to one, however, $com_\alpha(X)$ and $com_\alpha(X')$ are the positions of the geometric centers. Notice that there are sets of weights: w' and w. The first is used to calculate the position of the center of mass (so it determines how the atoms are *aligned*). Meanwhile, the second is used when calculating how far the atoms have actually been *displaced*. These weights are assigned in the reference configuration that you provide as input (i.e. the appear in the input file to this action that you set using REFERENCE=whatever.pdb). This input reference configuration consists of a simple pdb file containing the set of atoms for which you want to calculate the RMSD displacement and their positions in the reference configuration. It is important to note that the indices in this pdb need to be set correctly. The indices in this file determine the indices of the instantaneous atomic positions that are used by PLUMED when calculating this colvar. As such if you want to calculate the RMSD distance moved by the 1st, 4th, 6th and 28th atoms in the MD codes input file then the indices of the corresponding refernece positions in this pdb file should be set equal to 1, 4, 6 and 28.

The pdb input file should also contain the values of w and w'. In particular, the OCCUPANCY column (the first column after the coordinates) is used provides the values of w' that are used to calculate the position of the centre of mass. The BETA column (the second column after the Cartesian coordinates) is used to provide the w values which are used in the the calculation of the displacement. Please note that it is possible to use fractional values for beta and for the occupancy. However, we recommend you only do this when you really know what you are doing however as the results can be rather strange.

In PDB files the atomic coordinates and box lengths should be in Angstroms unless you are working with natural units. If you are working with natural units then the coordinates should be in your natural length unit. For more details on the PDB file format visit http://www.wwpdb.org/docs.html.

A different method is used to calculate the RMSD distance when you use TYPE=OPTIMAL on the input line. In this case the root mean square deviation is calculated after the positions of geometric centers in the reference and instantaneous configurations are aligned AND after an optimal alignment of the two frames is performed so that motion due to rotation of the reference frame between the two structures is removed. The equation for $d(X,X')$ in this case reads:

$$d(X,X') = \sqrt{\sum_i \sum_\alpha^{x,y,z} \frac{w_i}{\sum_j w_j} [X_{i,\alpha} - com_\alpha(X) - \sum_\beta M(X,X',w')_{\alpha,\beta} (X'_{i,\beta} - com_\beta(X'))]^2}$$

where $M(X,X',w')$ is the optimal alignment matrix which is calculated using the Kearsley [16] algorithm. Again different sets of weights are used for the alignment (w') and for the displacement calcuations (w). This gives a great deal of flexibility as it allows you to use a different sets of atoms (which may or may not overlap) for the alignment and displacement parts of the calculation. This may be very useful when you want to calculate how a ligand moves about in a protein cavity as you can use the protein as a reference system and do no alignment of the ligand.

(Note: when this form of RMSD is used to calculate the secondary structure variables (ALPHARMSD, ANTIBETARMSD and PARABETARMSD all the atoms in the segment are assumed to be part of both the alignment and displacement sets and all weights are set equal to one)

Please note that there are a number of other methods for calculating the distance between the instantaneous configuration and a reference configuration that are available in plumed. More information on these various methods can be found in the section of the manual on Distances from reference configurations.

Compulsory keywords

REFERENCE	a file in pdb format containing the reference structure and the atoms involved in the CV.
TYPE	(default=SIMPLE) the manner in which RMSD alignment is performed. Should be OPTIMAL or SIMPLE.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
SQUARED	(default=off) This should be setted if you want MSD instead of RMSD

Examples

The following tells plumed to calculate the RMSD distance between the positions of the atoms in the reference file and their instantaneous position. The Kearseley algorithm is used so this is done optimally.

```
RMSD REFERENCE=file.pdb TYPE=OPTIMAL
```

...

5.3.4 TARGET

	This is part of the function module

This function measures the pythagorean distance from a particular structure measured in the space defined by some set of collective variables.

Compulsory keywords

TYPE	(default=EUCLIDEAN) the manner in which the distance should be calculated
REFERENCE	a file in pdb format containing the reference structure. In the PDB file the atomic coordinates and box lengths should be in Angstroms unless you are working with natural units. If you are working with natural units then the coordinates should be in your natural length unit. The charges and masses of the atoms (if required) should be inserted in the beta and occupancy columns respectively. For more details on the PDB file format visit http://www.wwpdb.org/docs.html

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

5.4 Functions

When performing biased dynamics or analysing a trajectory you may wish to analyse/bias the value of some function of a set of collective variables rather than the values of the collective variables directly. You can do this with PLUMED by using any one of the following list of functions:

COMBINE	Calculate a polynomial combination of a set of other variables.
ENSEMBLE	Calculates the replica averaging of a collective variable over multiple replicas.
FUNCPATHMSD	This function calculates path collective variables.
FUNCSUMHILLS	This function is intended to be called by the command line tool sum_hillsand it is meant to integrate a HILLS file or an HILLS file interpreted as a histogram i a variety of ways. Therefore it is not expected that you use this during your dynamics (it will crash!)
MATHEVAL	Calculate a combination of variables using a matheval expression.
PIECEWISE	Compute a piecewise straight line through its arguments that passes througha set of ordered control points.
SORT	This function can be used to sort colvars according to their magnitudes.

5.4.1 COMBINE

	This is part of the function module

Calculate a polynomial combination of a set of other variables.

The functional form of this function is

$$C = \sum_{i=1}^{N_{arg}} c_i x_i^{p_i}$$

The coefficients c and powers p are provided as vectors.

Notice that COMBINE is not able to predict which will be periodic domain of the computed value automatically. The user is thus forced to specify it explicitly. Use PERIODIC=NO if the resulting variable is not periodic, and PERIODIC=A,B where A and B are the two boundaries if the resulting variable is periodic.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.
PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
COEFFICIENTS	(default=1.0) the coefficients of the arguments in your function
POWERS	(default=1.0) the powers to which you are raising each of the arguments in your function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NORMALIZE	(default=off) normalize all the coefficents so that in total they are equal to one

Examples

The following input tells plumed to print the distance between atoms 3 and 5 its square (as computed from the x,y,z components) and the distance again as computed from the square root of the square.

```
DISTANCE LABEL=dist      ATOMS=3,5 COMPONENTS
COMBINE  LABEL=distance2 ARG=dist.x,dist.y,dist.z POWERS=2,2,2 PERIODIC=NO
COMBINE  LABEL=distance  ARG=distance2 POWERS=0.5 PERIODIC=NO
PRINT ARG=distance,distance2
```

(See also PRINT and DISTANCE).

The following input tells plumed to add a restraint on the cube of a dihedral angle. Notice that since the angle has a periodic domain -pi,pi its cube has a domain -pi$**$3,pi$**$3.

```
t: TORSION ATOMS=1,3,5,7
c: COMBINE ARG=t POWERS=3 PERIODIC=-31.0062766802998,31.0062766802998
RESTRAINT ARG=c KAPPA=10 AT=0
```

5.4.2 ENSEMBLE

	This is part of the function module

Calculates the replica averaging of a collective variable over multiple replicas.

Each collective variable is averaged separately and stored in a component labelled *label*.cvlabel.

Note that in case of variables such as CS2BACKBONE, CH3SHIFTS, NOE and RDC it is possible to perform the replica-averaging inside the variable, in fact in those cases are the single experimental values that averaged before calculating the collective variable.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input tells plumed to calculate the distance between atoms 3 and 5 and the average it over the available replicas.

```
dist: DISTANCE ATOMS=3,5
ens: ENSEMBLE ARG=dist
PRINT ARG=dist,ens.dist
```

(See also PRINT and DISTANCE).

5.4.3 FUNCPATHMSD

	This is part of the function module

This function calculates path collective variables.

This is the Path Collective Variables implementation (see [15]). This variable computes the progress along a given set of frames that is provided in input ("s" component) and the distance from them ("z" component). It is a function of MSD that are obtained by the joint use of MSD variable and SQUARED flag (see below).

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
s	the position on the path
z	the distance from the path

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

LAMBDA	the lambda parameter is needed for smoothing, is in the units of plumed

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

NEIGH_SIZE	size of the neighbor list
NEIGH_STRIDE	how often the neighbor list needs to be calculated in time units

Examples

Here below is a case where you have defined three frames and you want to calculate the progress alng the path and the distance from it in p1

```
t1: RMSD REFERENCE=frame_1.dat TYPE=OPTIMAL SQUARED
t2: RMSD REFERENCE=frame_21.dat TYPE=OPTIMAL SQUARED
t3: RMSD REFERENCE=frame_42.dat TYPE=OPTIMAL SQUARED
p1: FUNCPATHMSD ARG=t1,t2,t3 LAMBDA=500.0
PRINT ARG=t1,t2,t3,p1.s,p1.z STRIDE=1 FILE=colvar FMT=%8.4f
```

In this second example is shown how to define a PATH in the CONTACTMAP space:

```
CONTACTMAP ...
ATOMS1=1,2 REFERENCE1=0.1
ATOMS2=3,4 REFERENCE2=0.5
ATOMS3=4,5 REFERENCE3=0.25
ATOMS4=5,6 REFERENCE4=0.0
SWITCH=(RATIONAL R_0=1.5)
LABEL=c1
CMDIST
... CONTACTMAP

CONTACTMAP ...
ATOMS1=1,2 REFERENCE1=0.3
ATOMS2=3,4 REFERENCE2=0.9
ATOMS3=4,5 REFERENCE3=0.45
ATOMS4=5,6 REFERENCE4=0.1
SWITCH=(RATIONAL R_0=1.5)
LABEL=c2
CMDIST
... CONTACTMAP

CONTACTMAP ...
ATOMS1=1,2 REFERENCE1=1.0
ATOMS2=3,4 REFERENCE2=1.0
ATOMS3=4,5 REFERENCE3=1.0
ATOMS4=5,6 REFERENCE4=1.0
SWITCH=(RATIONAL R_0=1.5)
LABEL=c3
CMDIST
... CONTACTMAP

p1: FUNCPATHMSD ARG=c1,c2,c3 LAMBDA=500.0
PRINT ARG=c1,c2,c3,p1.s,p1.z STRIDE=1 FILE=colvar FMT=%8.4f
```

5.4.4 FUNCSUMHILLS

	This is part of the function module

This function is intended to be called by the command line tool sum_hills and it is meant to integrate a HILLS file or an HILLS file interpreted as a histogram i a variety of ways. Therefore it is not expected that you use this during your dynamics (it will crash!)

In the future one could implement periodic integration during the metadynamics or straightforward MD as a tool to check convergence

5.4.5 MATHEVAL

	This is part of the function module

Calculate a combination of variables using a matheval expression.

This action computes an arbitrary function of one or more precomputed collective variables. Arguments are chosen with the ARG keyword, and the function is provided with the FUNC string. Notice that this string should contain no space. Within FUNC, one can refer to the arguments as x,y,z, and t (up to four variables provided as ARG). This names can be customized using the VAR keyword (see examples below).

If you want a function that depends not only on collective variables but also on time you can use the TIME action.

Attention

The MATHEVAL object only works if libmatheval is installed on the system and PLUMED has been linked to it

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

PERIODIC	if the output of your function is periodic then you should specify the periodicity of the function. If the output is not periodic you must state this using PERIODIC=NO
FUNC	the function you wish to evaluate

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

VAR	the names to give each of the arguments in the function. If you have up to three arguments in your function you can use x, y and z to refer to them. Otherwise you must use this flag to give your variables names.

Examples

The following input tells plumed to perform a metadynamics using as a CV the difference between two distances.

```
dAB: DISTANCE ARG=10,12
dAC: DISTANCE ARG=10,15
diff: MATHEVAL ARG=dAB,dAC FUNC=y-x PERIODIC=NO
# notice: the previous line could be replaced with the following
# diff: COMBINE ARG=dAB,dAC COEFFICIENTS=-1,1
METAD ARG=diff WIDTH=0.1 HEIGHT=0.5 BIASFACTOR=10 PACE=100
```

(see also DISTANCE, COMBINE, and METAD). Notice that forces applied to diff will be correctly propagated to atoms 10, 12, and 15. Also notice that since MATHEVAL is used without the VAR option the two arguments should be referred to as x and y in the expression FUNC. For simple functions such as this one it is possible to use COMBINE, which does not require libmatheval to be installed on your system.

The following input tells plumed to print the angle between vectors identified by atoms 1,2 and atoms 2,3 its square (as computed from the x,y,z components) and the distance again as computed from the square root of the square.

```
DISTANCE LABEL=d1 ATOMS=1,2 COMPONENTS
DISTANCE LABEL=d2 ATOMS=2,3 COMPONENTS
MATHEVAL ...
  LABEL=theta
  ARG=d1.x,d1.y,d1.z,d2.x,d2.y,d2.z
  VAR=ax,ay,az,bx,by,bz
  FUNC=acos((ax*bx+ay*by+az*bz)/sqrt((ax*ax+ay*ay+az*az)*(bx*bx+by*by+bz*bz))
  PERIODIC=NO
... MATHEVAL
PRINT ARG=theta
```

(See also PRINT and DISTANCE).

Notice that the matheval library implements a large number of functions (trigonometric, exp, log, etc). Among the useful functions, have a look at the step function (that is the Heaviside function). `step(x)` is defined as 1 when `x` is positive and `0` when x is negative. This allows for a straightforward implementation of if clauses.

For example, imagine that you want to implement a restraint that only acts when a distance is larger than 0.5. You can do it with

```
d: DISTANCE ATOMS=10,15
m: MATHEVAL ARG=d FUNC=0.5*step(0.5-x)+x*step(x-0.5) PERIODIC=NO
# check the function you are applying:
PRINT ARG=d,n FILE=checkme
RESTRAINT ARG=d AT=0.5 KAPPA=10.0
```

(see also DISTANCE, PRINT, and RESTRAINT)

The meaning of the function `0.5*step(0.5-x)+x*step(x-0.5)` is:

- If x<0.5 (step(0.5-x)!=0) use 0.5
- If x>0.5 (step(x-0.5)!=0) use x Notice that the same could have been obtained using an UPPER_WALLS However, with MATHEVAL you can create way more complex definitions.

Warning

If you apply forces on the variable (as in the previous example) you should make sure that the variable is continuous! Conversely, if you are just analyzing a trajectory you can safely use discontinuous variables.

A possible continuity check with gnuplot is

```
# this allow to step function to be used in gnuplot:
gnuplot> step(x)=0.5*(erf(x*10000000)+1)
# here you can test your function
gnuplot> p 0.5*step(0.5-x)+x*step(x-0.5)
```

Also notice that you can easily make logical operations on the conditions that you create. The equivalent of the AND operator is the product: `step(1.0-x)*step(x-0.5)` is only equal to 1 when x is between 0.5 and 1.0. By combining negation and AND you can obtain an OR. That is, `1-step(1.0-x)*step(x-0.5)` is only equal to 1 when x is outside the 0.5-1.0 interval.

5.4.5.1 TIME

	This is part of the generic module

retrieve the time of the simulation to be used elsewere

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

```
TIME            LABEL=t1
PRINT ARG=t1
```

(See also PRINT).

5.4.6 PIECEWISE

	This is part of the function module

Compute a piecewise straight line through its arguments that passes through a set of ordered control points.

For variables less than the first (greater than the last) point, the value of the first (last) point is used.

$$\frac{y_{i+1}-y_i}{x_{i+1}-x_i}(s-x_i)+y_i; if x_i < s < x_{i+1}$$

$$y_N; if x > x_{N-1}$$
$$y_1; if x < x_0$$

Control points are passed using the POINT0=... POINT1=... syntax as in the example below

If one argument is supplied, it results in a scalar quantity. If multiple arguments are supplied, it results in a vector of values. Each value will be named as the name of the original argument with suffix _pfunc.

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
_pfunc	one or multiple instances of this quantity will be referenceable elsewhere in the input file. These quantities will be named with the arguments of the function followed by the character string _pfunc. These quantities tell the user the values of the piecewise functions of each of the arguments.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

POINT	This keyword is used to specify the various points in the function above. You can use multiple instances of this keyword i.e. POINT1, POINT2, POINT3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

```
dist1: DISTANCE ATOMS=1,10
dist2: DISTANCE ATOMS=2,11

pw: PIECEWISE POINT0=1,10 POINT1=1,PI POINT2=3,10 ARG=dist1
ppww: PIECEWISE POINT0=1,10 POINT1=1,PI POINT2=3,10 ARG=dist1,dist2
PRINT ARG=pw,ppww.dist1_pfunc,ppww.dist2_pfunc
```

(See also PRINT and DISTANCE).

5.4.7 SORT

	This is part of the function module

This function can be used to sort colvars according to their magnitudes.

Description of components

This function sorts its arguments according to their magnitudes. The lowest argument will be labelled *label*.1, the second lowest will be labelled *label*.2 and so on.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input tells plumed to print the distance of the closest and of the farthest atoms to atom 1, chosen among atoms from 2 to 5

```
d12:  DISTANCE ATOMS=1,2
d13:  DISTANCE ATOMS=1,3
d14:  DISTANCE ATOMS=1,4
d15:  DISTANCE ATOMS=1,5
sort: SORT ARG=d12,d13,d14,d15
PRINT ARG=sort.1,sort.4
```

(See also PRINT and DISTANCE).

5.5 MultiColvar Documentation

Oftentimes, when you do not need one of the collective variables described elsewhere in the manual, what you want instead is a function of a distribution of collective variables of a particular type. For instance you might need to calculate a minimum distance or the number of coordination numbers greater than a 3.0. To avoid dupilcating the code to calculate an angle or distance many times and to make it easier to implement very complex collective variables PLUMED provides these sort of collective variables using so-called MultiColvars. MultiColvars are named in this way because a single PLUMED action can be used to calculate a number of different collective variables. For instance the DISTANCES action can be used to calculate the minimum distance, the number of distances less than a certain value, the number of distances within a certain range... A more detailed introduction to multicolvars is provided in this 10-minute video. Descriptions of the various multicolvars that are implemented in PLUMED 2 are given below:

ANGLES	Calculate functions of the distribution of angles .
BRIDGE	Calculate the number of atoms that bridge two parts of a structure
COORDINATIONNUMBER	Calculate the coordination numbers of atoms so that you can then calculate functions of the distribution ofcoordination numbers such as the minimum, the number less than a certain quantity and so on.
DENSITY	Calculate functions of the density of atoms as a function of the box. This allows one to calculatethe number of atoms in half the box.
DISTANCES	Calculate the distances between one or many pairs of atoms. You can then calculate functions of the distribution ofdistances such as the minimum, the number less than a certain quantity and so on.
FCCUBIC	
MOLECULES	Calculate the vectors connecting a pair of atoms in order to represent the orientation of a molecule.
Q3	Calculate 3rd order Steinhardt parameters.
Q4	Calculate 4th order Steinhardt parameters.
Q6	Calculate 6th order Steinhardt parameters.
SIMPLECUBIC	Calculate whether or not the coordination spheres of atoms are arranged as they would be in a simplecubic structure.

TETRAHEDRAL	
TORSIONS	Calculate whether or not a set of torsional angles are within a particular range.
XDISTANCES	Calculate the x components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution ofvalues such as the minimum, the number less than a certain quantity and so on.
YDISTANCES	Calculate the y components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution ofvalues such as the minimum, the number less than a certain quantity and so on.
ZDISTANCES	Calculate the z components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution ofvalues such as the minimum, the number less than a certain quantity and so on.

To instruct PLUMED to calculate a multicolvar you give an instruction that looks something like this:

```
NAME <atoms involved> <parameters> <what am I calculating> TOL=0.001 LABEL=label
```

Oftentimes the simplest way to specify the atoms involved is to use multiple instances of the ATOMS keyword i.e. ATOMS1, ATOMS2, ATOMS3,... Separate instances of the quantity specified by NAME are then calculated for each of the sets of atoms. For example if the command issued contains the following:

```
DISTANCES ATOMS1=1,2 ATOMS2=3,4 ATOMS3=5,6
```

The distances between atoms 1 and 2, atoms 3 and 4, and atoms 5 and 6 are calculated. Obviously, generating this sort of input is rather tedious so short cuts are also available many of the collective variables. These are described on the manual pages for the actions.

After specifying the atoms involved you sometimes need to specify some parameters that required in the calculation. For instance, for COORDINATIONNUMBER - the number of atoms in the first coordination sphere of each of the atoms in the system - you need to specify the parameters for a switchingfunction that will tell us whether or not an atom is in the first coordination sphere. Details as to how to do this are provided on the manual pages.

One of the most important keywords for multicolvars is the TOL keyword. This specifies that terms in sums that contribute less than a certain value can be ignored. In addition, it is assumed that the derivative with respect to these terms are essentially zero. By increasing the TOL parameter you can increase the speed of the calculation. Be aware, however, that this increase in speed is only possible because you are lowering the accuracy with which you are computing the quantity of interest.

Once you have specified the base quanties that are to be calculated from the atoms involved and any parameters you need to specify what function of these base quanties is to be calculated. For most multicolvars you can calculate the minimum, the number less than a target value, the number within a certain range, the number more than a target value and the average value directly.

5.5.1 MultiColvar functions

As well as the relatively simple quantities described above more complex functions of the distribution of values for the base colvars can be computed by employing multicolvars in conjuction with the following actions:

AROUND	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.

LOCAL_AVERAGE	Calculate averages over spherical regions centered on atoms
LOCAL_Q3	Calculate the local degree of order around an atoms by taking the average dot product between the q_3 vector on the central atom and the q_3 vectoron the atoms in the first coordination sphere.
LOCAL_Q4	Calculate the local degree of order around an atoms by taking the average dot product between the q_4 vector on the central atom and the q_4 vectoron the atoms in the first coordination sphere.
LOCAL_Q6	Calculate the local degree of order around an atoms by taking the average dot product between the q_6 vector on the central atom and the q_6 vectoron the atoms in the first coordination sphere.
NLINKS	Calculate number of pairs of atoms/molecules that are "linked"
SPRINT	Calculate SPRINT topological variables.

5.5.2 MultiColvar bias

There may be occasitions when you want add restraints on many collective variables. For instance if you are studying a cluster you might want to add a wall on the distances between each of the atoms and the center of mass of the cluster in order to prevent the cluster subliming. Alternatively, you may wish to insist that a particular set of atoms in your system all have a coordination number greater than 2. You can add these sorts of restraints by employing the following biases, which all act on the set of collective variable values calculated by a multicolvar. So for example the following set of commands:

```
COM ATOMS=1-20 LABEL=c1
DISTANCES GROUPA=c1 GROUPB=1-20 LABEL=d1
UWALLS DATA=d1 AT=2.5 KAPPA=0.2 LABEL=sr
```

creates the aforementioned set of restraints on the distances between the 20 atoms in a cluster and the center of mass of the cluster.

The list of biases of this type are as follows:

UWALLS	Add UPPER_WALLS restraints on all the multicolvar values

Notice that (in theory) you could also use this functionality to add additional terms to your forcefield or to implement your forcefield.

5.5.3 ANGLES

	This is part of the multicolvar module

Calculate functions of the distribution of angles .

You can use this command to calculate functions such as:

$$f(x) = \sum_{ijk} g(\theta_{ijk})$$

Alternatively you can use this command to calculate functions such as:

$$f(x) = \sum_{ijk} s(r_{ij}) s(r_{jk}) g(\theta_{ijk})$$

where $s(r)$ is a switchingfunction. This second form means that you can use this to calculate functions of the angles in the first coordination sphere of an atom / molecule [17].

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Or alternatively by using

GROUP	Calculate angles for each distinct set of three atoms in the group

Or alternatively by using

GROUPA	A group of central atoms about which angles should be calculated
GROUPB	When used in conjuction with GROUPA this keyword instructs plumed to calculate all distinct angles involving one atom from GROUPA and two atoms from GROUPB. The atom from GROUPA is the central atom.

Or alternatively by using

GROUPC	This must be used in conjuction with GROUPA and GROUPB. All angles involving one atom from GROUPA, one atom from GROUPB and one atom from GROUPC are calculated. The GROUPA atoms are assumed to be the central atoms

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
SWITCH	A switching function that ensures that only angles between atoms that are within a certain fixed cutoff are calculated. The following provides information on the switchingfunction that are available.
SWITCHA	A switching function on the distance between the atoms in group A and the atoms in group B
SWITCHB	A switching function on the distance between the atoms in group A and the atoms in group B

Examples

The following example instructs plumed to find the average of two angles and to print it to a file

```
ANGLES ATOMS1=1,2,3 ATOMS2=4,5,6 MEAN LABEL=a1
PRINT ARG=a1.mean FILE=colvar
```

The following example tells plumed to calculate all angles involving at least one atom from GROUPA and two atoms from GROUPB in which the distances are less than 1.0. The number of angles between $\frac{\pi}{4}$ and $\frac{3\pi}{4}$ is then output

```
ANGLES GROUPA=1-10 GROUPB=11-100 BETWEEN={GAUSSIAN LOWER=0.25pi UPPER=0.75pi} SWITCH={GAUSSIAN R_0=1.0} LABEL=
PRINT ARG=a1.between FILE=colvar
```

This final example instructs plumed to calculate all the angles in the first coordination spheres of the atoms. A discretized-normalized histogram of the distribution is then output

```
ANGLES GROUP=1-38 HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=pi NBINS=20} SWITCH={GAUSSIAN R_0=1.0} LABEL=a1
PRINT ARG=a1.* FILE=colvar
```

5.5.4 BRIDGE

	This is part of the multicolvar module

Calculate the number of atoms that bridge two parts of a structure

This quantity calculates:

$$f(x) = \sum_{ijk} s_A(r_{ij}) s_B(r_{ik})$$

where the sum over i is over all the “bridging atoms" and s_A and s_B are switchingfunction.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Or alternatively by using

BRIDGING_ATOMS	The list of atoms that can form the bridge between the two interesting parts of the structure.
GROUPA	The list of atoms that are in the first interesting part of the structure
GROUPB	The list of atoms that are in the second interesting part of the structure

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

SWITCH	The parameters of the two switchingfunction in the above formula
SWITCHA	The switchingfunction on the distance between bridging atoms and the atoms in group A
SWITCHB	The switchingfunction on the distance between the bridging atoms and the atoms in group B

Examples

The following example instructs plumed to calculate the number of water molecules that are bridging betweeen atoms 1-10 and atoms 11-20 and to print the value to a file

```
BRIDGE BRIDGING_ATOMS=100-200 GROUPA=1-10 GROUPB=11-20 LABEL=w1
PRINT ARG=a1.mean FILE=colvar
```

5.5.5 COORDINATIONNUMBER

	This is part of the multicolvar module

Calculate the coordination numbers of atoms so that you can then calculate functions of the distribution of coordination numbers such as the minimum, the number less than a certain quantity and so on.

To make the calculation of coordination numbers differentiable the following function is used:

$$s = \frac{1-\left(\frac{r-d_0}{r_0}\right)^n}{1-\left(\frac{r-d_0}{r_0}\right)^m}$$

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
max	**MAX**	the maximum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom.

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPECIESB is within the specified cutoff
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta \log \sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.max.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

The following input tells plumed to calculate the coordination numbers of atoms 1-100 with themselves. The minimum coordination number is then calculated.

```
COORDINATIONNUMBER SPECIES=1-100 R_0=1.0 MIN={BETA=0.1}
```

The following input tells plumed to calculate how many atoms from 1-100 are within 3.0 of each of the atoms from

101-110. In the first 101 is the central atom, in the second 102 is the central atom and so on. The number of coordination numbers more than 6 is then computed.

```
COORDINATIONNUMBER SPECIESA=101-110 SPECIESB=1-100 R_0=3.0 MORE_THAN={RATIONAL R_0=6.0 NN=6 MM=12 D_0=0}
```

5.5.6 DENSITY

	This is part of the multicolvar module

Calculate functions of the density of atoms as a function of the box. This allows one to calculate the number of atoms in half the box.

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

Examples

The following example calculates the number of atoms in one half of the simulation box.

```
DENSITY SPECIES=1-100 LABEL=d
SUBCELL ARG=d XLOWER=0.0 XUPPER=0.5 LABEL=d1
PRINT ARG=d1.* FILE=colvar1 FMT=%8.4f
```

5.5.7 DISTANCES

	This is part of the multicolvar module

Calculate the distances between one or many pairs of atoms. You can then calculate functions of the distribution of distances such as the minimum, the number less than a certain quantity and so on.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
dhenergy	**DHENERGY**	the Debye-Huckel interaction energy. You can calculate this quantity multiple times using different parameters
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
max	**MAX**	the maximum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean

min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Or alternatively by using

GROUP	Calculate the distance between each distinct pair of atoms in the group

Or alternatively by using

GROUPA	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjuction with GROUPB.
GROUPB	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjuction with GROUPA.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta \log \sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.max.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
DHENERGY	calculate the Debye-Huckel interaction energy. This is a alternative implementation of DHENERGY that is particularly useful if you want to calculate the Debye-Huckel interaction energy and some other function of set of distances between the atoms in the two groups. The input for this keyword should read DHENERGY={I= I TEMP= T EPSILON= ε}. You can use multiple instances of this keyword i.e. DHENERGY1, DHENERGY2, DHENERGY3...

MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i-\bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

The following input tells plumed to calculate the distances between atoms 3 and 5 and between atoms 1 and 2 and to print the minimum for these two distances.

```
DISTANCES ATOMS1=3,5 ATOMS2=1,2 MIN={BETA=0.1} LABEL=d1
PRINT ARG=d1.min
```

(See also PRINT).

The following input tells plumed to calculate the distances between atoms 3 and 5 and between atoms 1 and 2 and then to calculate the number of these distances that are less than 0.1 nm. The number of distances less than 0.1nm is then printed to a file.

```
DISTANCES ATOMS1=3,5 ATOMS2=1,2 LABEL=d1 LESS_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.lt0.1
```

(See also PRINT switchingfunction).

The following input tells plumed to calculate all the distances between atoms 1, 2 and 3 (i.e. the distances between atoms 1 and 2, atoms 1 and 3 and atoms 2 and 3). The average of these distances is then calculated.

```
DISTANCES GROUP=1-3 MEAN LABEL=d1
PRINT ARG=d1.mean
```

(See also PRINT)

The following input tells plumed to calculate all the distances between the atoms in GROUPA and the atoms in GROUPB. In other words the distances between atoms 1 and 2 and the distance between atoms 1 and 3. The number of distances more than 0.1 is then printed to a file.

```
DISTANCES GROUPA=1 GROUPB=2,3 MORE_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.gt0.1
```

(See also PRINT switchingfunction)

Calculating minimum distances

To calculate and print the minimum distance between two groups of atoms you use the following commands

```
d1: DISTANCES GROUPA=1-10 GROUPB=11-20 MIN={BETA=500.}
PRINT ARG=d1.min FILE=colvar STRIDE=10
```

(see DISTANCES and PRINT)

In order to ensure differentiability the minimum is calculated using the following function:

$$s = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$$

where β is a user specified parameter.

This input is used rather than a separate MINDIST colvar so that the same routine and the same input style can be used to calculate minimum coordinatetion numbers (see COORDINATIONNUMBER), minimum angles (see ANGLES) and many other variables.

This new way of calculating mindist is part of plumed 2's multicolvar functionality. These special actions allow you to calculate multiple functions of a distribution of simple collective variables. As an example you can calculate the number of distances less than 1.0, the minimum distance, the number of distances more than 2.0 and the number of distances between 1.0 and 2.0 by using the following command:

```
DISTANCES ...
 GROUPA=1-10 GROUPB=11-20
 LESS_THAN={RATIONAL R_0=1.0}
 MORE_THAN={RATIONAL R_0=2.0}
 BETWEEN={GAUSSIAN LOWER=1.0 UPPER=2.0}
 MIN={BETA=500.}
... DISTANCES
PRINT ARG=d1.lessthan,d1.morethan,d1.between,d1.min FILE=colvar STRIDE=10
```

(see DISTANCES and PRINT)

A calculation performed this way is fast because the expensive part of the calculation - the calculation of all the distances - is only done once per step. Furthermore, it can be made faster by using the TOL keyword to discard those distance that make only a small contributions to the final values together with the NL_STRIDE keyword, which ensures that the distances that make only a small contribution to the final values aren't calculated at every step.

5.5.8 FCCUBIC

	This is part of the crystallization module

5.5.9 MOLECULES

	This is part of the crystallization module

Calculate the vectors connecting a pair of atoms in order to represent the orientation of a molecule.

At its simplest this command can be used to calculate the average length of an internal vector in a collection of different molecules. When used in conjunction with MutiColvarFunctions in can be used to do a variety of more complex tasks.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
vmean	**VMEAN**	the norm of the mean vector. The output component can be refererred to elsewhere in the input file by using the label.vmean

The atoms involved can be specified using

MOL	The numerical indices of the atoms in the molecule. The orientation of the molecule is equal to the vector connecting the first two atoms specified. If a third atom is specified its position is used to specify where the molecule is. If a third atom is not present the molecule is assumed to be at the center of the vector connecting the first two atoms. You can use multiple instances of this keyword i.e. MOL1, MOL2, MOL3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
VMEAN	(default=off) calculate the norm of the mean vector. The final value can be referenced using *label*.vmean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

Examples

The following input tells plumed to calculate the distances between two of the atoms in a molecule. This is done for the same set of atoms four different molecules and the average separation is then calculated.

```
MOLECULES MOL1=1,2 MOL2=3,4 MOL3=5,6 MOL4=7,8 MEAN LABEL=mm
PRINT ARG=mm.mean FILE=colvar
```

5.5.10 Q3

	This is part of the crystallization module

Calculate 3rd order Steinhardt parameters.

The 3rd order Steinhardt parameters allow us to measure the degree to which the first coordination shell around an atom is ordered. The Steinhardt parameter for atom, i is complex vector whose components are calculated using the following formula:

$$q_{3m}(i) = \frac{\sum_j \sigma(r_{ij}) Y_{3m}(\mathbf{r}_{ij})}{\sum_j \sigma(r_{ij})}$$

where Y_{3m} is one of the 3rd order spherical harmonics so m is a number that runs from -3 to $+3$. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

The Steinhardt parameters can be used to measure the degree of order in the system in a variety of different ways. The simplest way of measuring whether or not the coordination sphere is ordered is to simply take the norm of the above vector i.e.

$$Q_3(i) = \sqrt{\sum_{m=-3}^{3} q_{3m}(i)^* q_{3m}(i)}$$

This norm is small when the coordination shell is disordered and larger when the coordination shell is ordered. Furthermore, when the keywords LESS_THAN, MIN, MAX, HISTOGRAM, MEAN and so on are used with this colvar it is the distribution of these normed quantities that is investigated.

Other measures of order can be taken by averaging the components of the individual q_3 vectors individually or by taking dot products of the q_3 vectors on adjacent atoms. More information on these variables can be found in the documentation for LOCAL_Q3, LOCAL_AVERAGE and NLINKS.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
vmean	**VMEAN**	the norm of the mean vector. The output component can be refererred to elsewhere in the input file by using the label.vmean
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom.

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPECIESB is within the specified cutoff
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=12) The n parameter of the switching function
MM	(default=24) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean
VMEAN	(default=off) calculate the norm of the mean vector. The final value can be referenced using *label*.vmean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.

Examples

The following command calculates the average Q3 parameter for the 64 atoms in a box of Lennard Jones and prints this quantity to a file called colvar:

```
Q3 SPECIES=1-64 D_0=1.3 R_0=0.2 MEAN LABEL=q3
PRINT ARG=q3.mean FILE=colvar
```

The following command calculates the histogram of Q3 parameters for the 64 atoms in a box of Lennard Jones and prints these quantities to a file called colvar:

```
Q3 SPECIES=1-64 D_0=1.3 R_0=0.2 HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1} LABEL=q3
PRINT ARG=q3.* FILE=colvar
```

The following command could be used to measure the Q3 paramters that describe the arrangement of chlorine ions around the sodium atoms in NaCl. The imagined system here is composed of 64 NaCl formula units and the atoms are arranged in the input with the 64 Na^+ ions followed by the 64 Cl^- ions. Once again the average Q3 paramter is calculated and output to a file called colvar

```
Q3 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN LABEL=q3
PRINT ARG=q3.mean FILE=colvar
```

5.5.11 Q4

	This is part of the crystallization module

Calculate 4th order Steinhardt parameters.

The 4th order Steinhardt parameters allow us to measure the degree to which the first coordination shell around an atom is ordered. The Steinhardt parameter for atom, i is complex vector whose components are calculated using the following formula:

$$q_{4m}(i) = \frac{\sum_j \sigma(r_{ij}) Y_{4m}(\mathbf{r}_{ij})}{\sum_j \sigma(r_{ij})}$$

where Y_{4m} is one of the 4th order spherical harmonics so m is a number that runs from -4 to $+4$. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

The Steinhardt parameters can be used to measure the degree of order in the system in a variety of different ways. The simplest way of measuring whether or not the coordination sphere is ordered is to simply take the norm of the above vector i.e.

$$Q_4(i) = \sqrt{\sum_{m=-4}^{4} q_{4m}(i)^* q_{4m}(i)}$$

This norm is small when the coordination shell is disordered and larger when the coordination shell is ordered. Furthermore, when the keywords LESS_THAN, MIN, MAX, HISTOGRAM, MEAN and so on are used with this colvar it is the distribution of these normed quantities that is investigated.

Other measures of order can be taken by averaging the components of the individual q_4 vectors individually or by taking dot products of the q_4 vectors on adjacent atoms. More information on these variables can be found in the documentation for LOCAL_Q4, LOCAL_AVERAGE and NLINKS.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using

this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
vmean	**VMEAN**	the norm of the mean vector. The output component can be refererred to elsewhere in the input file by using the label.vmean
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom.

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPECIESB is within the specified cutoff
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=12) The n parameter of the switching function
MM	(default=24) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean
VMEAN	(default=off) calculate the norm of the mean vector. The final value can be referenced using *label*.vmean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.

Examples

The following command calculates the average Q4 parameter for the 64 atoms in a box of Lennard Jones and prints this quantity to a file called colvar:

```
Q4 SPECIES=1-64 D_0=1.3 R_0=0.2 MEAN LABEL=q4
PRINT ARG=q4.mean FILE=colvar
```

The following command calculates the histogram of Q4 parameters for the 64 atoms in a box of Lennard Jones and prints these quantities to a file called colvar:

```
Q4 SPECIES=1-64 D_0=1.3 R_0=0.2 HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1} LABEL=q4
PRINT ARG=q4.* FILE=colvar
```

The following command could be used to measure the Q4 paramters that describe the arrangement of chlorine ions around the sodium atoms in NaCl. The imagined system here is composed of 64 NaCl formula units and the atoms are arranged in the input with the 64 Na^+ ions followed by the 64 Cl^- ions. Once again the average Q4 paramter is calculated and output to a file called colvar

```
Q4 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN LABEL=q4
PRINT ARG=q4.mean FILE=colvar
```

5.5.12 Q6

	This is part of the crystallization module

Calculate 6th order Steinhardt parameters.

The 6th order Steinhardt parameters allow us to measure the degree to which the first coordination shell around an atom is ordered. The Steinhardt parameter for atom, i is complex vector whose components are calculated using the following formula:

$$q_{6m}(i) = \frac{\sum_j \sigma(r_{ij}) Y_{6m}(\mathbf{r}_{ij})}{\sum_j \sigma(r_{ij})}$$

where Y_{6m} is one of the 6th order spherical harmonics so m is a number that runs from -6 to $+6$. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

The Steinhardt parameters can be used to measure the degree of order in the system in a variety of different ways. The simplest way of measuring whether or not the coordination sphere is ordered is to simply take the norm of the above vector i.e.

$$Q_6(i) = \sqrt{\sum_{m=-6}^{6} q_{6m}(i)^* q_{6m}(i)}$$

This norm is small when the coordination shell is disordered and larger when the coordination shell is ordered. Furthermore, when the keywords LESS_THAN, MIN, MAX, HISTOGRAM, MEAN and so on are used with this colvar it is the distribution of these normed quantities that is investigated.

Other measures of order can be taken by averaging the components of the individual q_6 vectors individually or by taking dot products of the q_6 vectors on adjacent atoms. More information on these variables can be found in the documentation for LOCAL_Q6, LOCAL_AVERAGE and NLINKS.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using

this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
vmean	**VMEAN**	the norm of the mean vector. The output component can be refererred to elsewhere in the input file by using the label.vmean
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom.

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPECIESB is within the specified cutoff
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=12) The n parameter of the switching function
MM	(default=24) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean
VMEAN	(default=off) calculate the norm of the mean vector. The final value can be referenced using *label*.vmean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log\sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.

Examples

The following command calculates the average Q6 parameter for the 64 atoms in a box of Lennard Jones and prints this quantity to a file called colvar:

```
Q6 SPECIES=1-64 D_0=1.3 R_0=0.2 MEAN LABEL=q6
PRINT ARG=q6.mean FILE=colvar
```

The following command calculates the histogram of Q6 parameters for the 64 atoms in a box of Lennard Jones and prints these quantities to a file called colvar:

```
Q6 SPECIES=1-64 D_0=1.3 R_0=0.2 HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1} LABEL=q6
PRINT ARG=q6.* FILE=colvar
```

The following command could be used to measure the Q6 paramters that describe the arrangement of chlorine ions around the sodium atoms in NaCl. The imagined system here is composed of 64 NaCl formula units and the atoms are arranged in the input with the 64 Na^+ ions followed by the 64 Cl^- ions. Once again the average Q6 paramter is calculated and output to a file called colvar

```
Q6 SPECIESA=1-64 SPECIESB=65-128 D_0=1.3 R_0=0.2 MEAN LABEL=q6
PRINT ARG=q6.mean FILE=colvar
```

5.5.13 SIMPLECUBIC

	This is part of the crystallization module

Calculate whether or not the coordination spheres of atoms are arranged as they would be in a simple cubic structure.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

max	**MAX**	the maximum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom.

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPECIESB is within the specified cutoff

SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...

LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta \log \sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.max.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

The following input tells plumed to calculate the simple cubic parameter for the atoms 1-100 with themselves. The mean value is then calculated.

```
SIMPLECUBIC SPECIES=1-100 R_0=1.0 MEAN
```

The following input tells plumed to look at the ways atoms 1-100 are within 3.0 are arranged about atoms from 101-110. The number of simple cubic parameters that are greater than 0.8 is then output

```
SIMPLECUBIC SPECIESA=101-110 SPECIESB=1-100 R_0=3.0 MORE_THAN={RATIONAL R_0=0.8 NN=6 MM=12 D_0=0}
```

5.5.14 TETRAHEDRAL

	This is part of the crystallization module

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
max	**MAX**	the maximum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.

morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

SPECIES	this keyword is used for colvars such as coordination number. In that context it specifies that plumed should calculate one coordination number for each of the atoms specified. Each of these coordination numbers specifies how many of the other specified atoms are within a certain cutoff of the central atom.

Or alternatively by using

SPECIESA	this keyword is used for colvars such as the coordination number. In that context it species that plumed should calculate one coordination number for each of the atoms specified in SPECIESA. Each of these cooordination numbers specifies how many of the atoms specifies using SPECIESB is within the specified cutoff
SPECIESB	this keyword is used for colvars such as the coordination number. It must appear with SPECIESA. For a full explanation see the documentation for that keyword

Compulsory keywords

NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances

SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MAX	calculate the maximum value. To make this quantity continuous the maximum is calculated using $\max = \beta \log \sum_i \exp\left(\frac{s_i}{\beta}\right)$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.max.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.

BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i-\bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

5.5.15 TORSIONS

	This is part of the multicolvar module

Calculate whether or not a set of torsional angles are within a particular range.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

Examples

The following provides an example of the input for the torsions command

```
TORSIONS ...
ATOMS1=168,170,172,188
ATOMS2=170,172,188,190
ATOMS3=188,190,192,230
LABEL=ab
... TORSIONS
PRINT ARG=ab.* FILE=colvar STRIDE=10
```

Writing out the atoms involved in all the torsions in this way can be rather tedious. Thankfully if you are working with protein you can avoid this by using the MOLINFO command. PLUMED uses the pdb file that you provide to this command to learn about the topology of the protein molecule. This means that you can specify torsion angles using the following syntax:

```
MOLINFO MOLTYPE=protein STRUCTURE=myprotein.pdb
TORSIONS ...
ATOMS1=@phi-3
ATOMS2=@psi-3
ATOMS3=@phi-4
LABEL=ab
... TORSIONS
PRINT ARG=ab FILE=colvar STRIDE=10
```

Here, @phi-3 tells plumed that you would like to calculate the ϕ angle in the third residue of the protein. Similarly @psi-4 tells plumed that you want to calculate the ψ angle of the 4th residue of the protein.

5.5.16 XDISTANCES

	This is part of the multicolvar module

Calculate the x components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
dhenergy	**DHENERGY**	the Debye-Huckel interaction energy. You can calculate this quantity multiple times using different parameters
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Or alternatively by using

GROUP	Calculate the distance between each distinct pair of atoms in the group

Or alternatively by using

GROUPA	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjuction with GROUPB.
GROUPB	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjuction with GROUPA.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
DHENERGY	calculate the Debye-Huckel interaction energy. This is a alternative implementation of DHENERGY that is particularly useful if you want to calculate the Debye-Huckel interaction energy and some other function of set of distances between the atoms in the two groups. The input for this keyword should read DHENERGY={I= I TEMP= T EPSILON= ε}. You can use multiple instances of this keyword i.e. DHENERGY1, DHENERGY2, DHENERGY3...

MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

The following input tells plumed to calculate the x-component of the vector connecting atom 3 to atom 5 and the x-component of the vector connecting atom 1 to atom 2. The minimum of these two quantities is then printed

```
XDISTANCES ATOMS1=3,5 ATOMS2=1,2 MIN={BETA=0.1} LABEL=d1
PRINT ARG=d1.min
```

(See also PRINT).

The following input tells plumed to calculate the x-component of the vector connecting atom 3 to atom 5 and the x-component of the vector connecting atom 1 to atom 2. The number of values that are less than 0.1nm is then printed to a file.

```
XDISTANCES ATOMS1=3,5 ATOMS2=1,2 LABEL=d1 LESS_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.lt0.1
```

(See also PRINT switchingfunction).

The following input tells plumed to calculate the x-components of all the distinct vectors that can be created between atoms 1, 2 and 3 (i.e. the vectors between atoms 1 and 2, atoms 1 and 3 and atoms 2 and 3). The average of these quantities is then calculated.

```
XDISTANCES GROUP=1-3 AVERAGE LABEL=d1
PRINT ARG=d1.average
```

(See also PRINT)

The following input tells plumed to calculate all the vectors connecting the the atoms in GROUPA to the atoms in GROUPB. In other words the vector between atoms 1 and 2 and the vector between atoms 1 and 3. The number of values more than 0.1 is then printed to a file.

```
XDISTANCES GROUPA=1 GROUPB=2,3 MORE_THAN={RATIONAL R_0=0.1}
PRINT ARG=d1.gt0.1
```

(See also PRINT switchingfunction)

5.5.17 YDISTANCES

	This is part of the multicolvar module

Calculate the y components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
dhenergy	**DHENERGY**	the Debye-Huckel interaction energy. You can calculate this quantity multiple times using different parameters
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Or alternatively by using

GROUP	Calculate the distance between each distinct pair of atoms in the group

Or alternatively by using

GROUPA	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjuction with GROUPB.
GROUPB	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjuction with GROUPA.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
DHENERGY	calculate the Debye-Huckel interaction energy. This is a alternative implementation of DHENERGY that is particularly useful if you want to calculate the Debye-Huckel interaction energy and some other function of set of distances between the atoms in the two groups. The input for this keyword should read DHENERGY={I= I TEMP= T EPSILON= ε}. You can use multiple instances of this keyword i.e. DHENERGY1, DHENERGY2, DHENERGY3...
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...

BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histograbead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i-\bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

See documentation for XDISTANCES for examples of how to use this command. You just need to substitute YDI↩ STANCES for XDISTANCES to investigate the y component rather than the x component.

5.5.18 ZDISTANCES

	This is part of the multicolvar module

Calculate the z components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution of values such as the minimum, the number less than a certain quantity and so on.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
dhenergy	**DHENERGY**	the Debye-Huckel interaction energy. You can calculate this quantity multiple times using different parameters

between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

ATOMS	the atoms involved in each of the collective variables you wish to calculate. Keywords like ATOMS1, ATOMS2, ATOMS3,... should be listed and one CV will be calculated for each ATOM keyword you specify (all ATOM keywords should define the same number of atoms). The eventual number of quantities calculated by this action will depend on what functions of the distribution you choose to calculate. You can use multiple instances of this keyword i.e. ATOMS1, ATOMS2, ATOMS3...

Or alternatively by using

GROUP	Calculate the distance between each distinct pair of atoms in the group

Or alternatively by using

GROUPA	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjuction with GROUPB.
GROUPB	Calculate the distances between all the atoms in GROUPA and all the atoms in GROUPB. This must be used in conjuction with GROUPA.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
VERBOSE	(default=off) write a more detailed output
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...

DHENERGY	calculate the Debye-Huckel interaction energy. This is a alternative implementation of DHENERGY that is particularly useful if you want to calculate the Debye-Huckel interaction energy and some other function of set of distances between the atoms in the two groups. The input for this keyword should read DHENERGY={I= I TEMP= T EPSILON= ε}. You can use multiple instances of this keyword i.e. DHENERGY1, DHENERGY2, DHENERGY3...
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

See documentation for XDISTANCES for examples of how to use this command. You just need to substitute ZDI↩ STANCES for XDISTANCES to investigate the z component rather than the x component.

5.5.19 AROUND

	This is part of the multicolvar module

This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.

Each of the base quantities calculated by a multicolvar can can be assigned to a particular point in three dimensional space. For example, if we have the coordination numbers for all the atoms in the system each coordination number can be assumed to lie on the position of the central atom. Because each base quantity can be assigned to a particular point in space we can calculate functions of the distribution of base quantities in a particular part of the box by using:

$$\bar{s}_\tau = \frac{\sum_i f(s_i) w(x_i, y_i, z_i)}{\sum_i w(x_i, y_i, z_i)}$$

where the sum is over the collective variables, s_i, each of which can be thought to be at (x_i, y_i, z_i). The function $w(x_i, y_i, z_i)$ measures whether or not the system is in the subregion of interest. It is equal to:

$$w(x_i, y_i, z_i) = \int_{xl}^{xu} \int_{yl}^{yu} \int_{zl}^{zu} \mathrm{d}x\mathrm{d}y\mathrm{d}z K\left(\frac{x - x_i}{\sigma}\right) K\left(\frac{y - y_i}{\sigma}\right) K\left(\frac{z - z_i}{\sigma}\right)$$

where K is one of the kernel functions described on histograмbead and σ is a bandwidth parameter. The function (s_i) can be any of the usual LESS_THAN, MORE_THAN, WITHIN etc that are used in all other multicolvars.

When AROUND is used with the DENSITY action the number of atoms in the specified region is calculated

Description of components

This Action can be used to calculate the following quantities by employing the keywords listed below. You must select which from amongst these quantities you wish to calculate - this command cannot be run unless one of the quantities below is being calculated.These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.less_than-1, *label*.less_than-2 etc.

Quantity	Keyword	Description
vmean	**VMEAN**	the norm of the mean vector. The output component can be refererred to elsewhere in the input file by using the label.vmean
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean

morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

The atoms involved can be specified using

ATOM	the atom whose vicinity we are interested in examining. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

DATA	certain actions in plumed work by calculating a list of variables and summing over them. This particular action can be used to calculate functions of these base variables or prints them to a file. This keyword thus takes the label of one of those such variables as input.
SIGMA	the width of the function to be used for kernel density estimation
KERNEL	(default=gaussian) the type of kernel function to be used
XLOWER	(default=0.0) the lower boundary in x relative to the x coordinate of the atom (0 indicates use full extent of box).
XUPPER	(default=0.0) the upper boundary in x relative to the x coordinate of the atom (0 indicates use full extent of box).
YLOWER	(default=0.0) the lower boundary in y relative to the y coordinate of the atom (0 indicates use full extent of box).
YUPPER	(default=0.0) the upper boundary in y relative to the y coordinate of the atom (0 indicates use full extent of box).
ZLOWER	(default=0.0) the lower boundary in z relative to the z coordinate of the atom (0 indicates use full extent of box).
ZUPPER	(default=0.0) the upper boundary in z relative to the z coordinate of the atom (0 indicates use full extent of box).

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean
VMEAN	(default=off) calculate the norm of the mean vector. The final value can be referenced using *label*.vmean
OUTSIDE	(default=off) calculate quantities for colvars that are on atoms outside the region of interest

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0 - \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...

HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.

Examples

The following commands tell plumed to calculate the average coordination number for the atoms that have x (in fractional coordinates) within 2.0 nm of the com of mass c1. The final value will be labeled s.mean.

```
COM ATOMS=1-100 LABEL=c1
COORDINATIONNUMBER SPECIES=1-100 R_0=1.0 LABEL=c
AROUND ARG=c ORIGIN=c1 XLOWER=-2.0 XUPPER=2.0 SIGMA=0.1 MEAN LABEL=s
```

5.5.20 LOCAL_AVERAGE

	This is part of the multicolvar module

Calculate averages over spherical regions centered on atoms

As is explained in this video certain multicolvars calculate one scalar quantity or one vector for each of the atoms in the system. For example COORDINATIONNUMBER measures the coordination number of each of the atoms in the system and Q4 measures the 4th order Steinhardt parameter for each of the atoms in the system. These quantities provide tell us something about the disposition of the atoms in the first coordination sphere of each of the atoms of interest. Lechner and Dellago [18] have suggested that one can probe local order in a system by taking the average value of such symmetry functions over the atoms within a spherical cutoff of each of these atoms in the systems. When this is done with Steinhardt parameters they claim this gives a coordinate that is better able to distinguish solid and liquid configurations of Lennard-Jones atoms.

You can calculate such locally averaged quantities within plumed by using the LOCAL_AVERAGE command. This command calculates the following atom-centered quantities:

$$s_i = \frac{c_i + \sum_j \sigma(r_{ij}) c_j}{1 + \sum_j \sigma(r_{ij})}$$

where the c_i and c_j values can be for any one of the symmetry functions that can be calculated using plumed multicolvars. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. Lechner and Dellago suggest that the parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise.

The s_i quantities calculated using the above command can be again thought of as atom-centred symmetry functions. They thus operate much like multicolvars. You can thus calculate properties of the distribution of s_i values using MEAN, LESS_THAN, HISTOGRAM and so on. You can also probe the value of these averaged variables in regions of the box by using the command in tandem with the AROUND command.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

Compulsory keywords

DATA	the labels of the action that calculates the multicolvars we are interested in
NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize

MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean
LOWMEM	(default=off) lower the memory requirements

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...

HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i-\bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

This example input calculates the coordination numbers for all the atoms in the system. These coordination numbers are then averaged over spherical regions. The number of averaged coordination numbers that are greater than 4 is then output to a file.

```
COORDINATIONNUMBER SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=d1
LOCAL_AVERAGE ARG=d1 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MORE_THAN={RATIONAL R_0=4} LABEL=la
PRINT ARG=la.* FILE=colvar
```

This example input calculates the q_4 (see Q4) vectors for each of the atoms in the system. These vectors are then averaged component by component over a spherical region. The average value for this quantity is then outputeed to a file. This calculates the quantities that were used in the paper by Lechner and Dellago [18]

```
Q4 SPECIES=1-64 SWITCH={RATIONAL D_0=1.3 R_0=0.2} LABEL=q4
LOCAL_AVERAGE ARG=q4 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LABEL=la
PRINT ARG=la.* FILE=colvar
```

5.5.21 LOCAL_Q3

	This is part of the crystallization module

Calculate the local degree of order around an atoms by taking the average dot product between the q_3 vector on the central atom and the q_3 vector on the atoms in the first coordination sphere.

The Q3 command allows one to calculate one complex vectors for each of the atoms in your system that describe the degree of order in the coordination sphere around a particular atom. The difficulty with these vectors comes when combining the order parameters from all of the individual atoms/molecules so as to get a measure of the global degree of order for the system. The simplest way of doing this - calculating the average Steinhardt parameter - can be problematic. If one is examining nucleation say only the order parameters for those atoms in the nucleus will change significantly when the nucleus forms. The order parameters for the atoms in the surrounding liquid will remain pretty much the same. As such if one models a small nucleus embedded in a very large amount of solution/melt any change in the average order parameter will be negligible. Substantial changes in the value of this average can be observed in simulations of nucleation but only because the number of atoms is relatively small.

When the average Q3 parameter is used to bias the dynamics a problems can occur. These averaged coordinates cannot distinguish between the correct, single-nucleus pathway and a concerted pathway in which all the atoms rearrange themselves into their solid-like configuration simultaneously. This second type of pathway would be impossible in reality because there is a large entropic barrier that prevents concerted processes like this from happening. However, in the finite sized systems that are commonly simulated this barrier is reduced substantially. As a result in simulations where average Steinhardt parameters are biased there are often quite dramatic system size effects

If one wants to simulate nucleation using some form on biased dynamics what is really required is an order parameter that measures:

- Whether or not the coordination spheres around atoms are ordered

- Whether or not the atoms that are ordered are clustered together in a crystalline nucleus

LOCAL_AVERAGE and NLINKS are variables that can be combined with the Steinhardt parameteters allow to calculate variables that satisfy these requirements. LOCAL_Q3 is another variable that can be used in these sorts of calculations. The LOCAL_Q3 parameter for a particular atom is a number that measures the extent to which the orientation of the atoms in the first coordination sphere of an atom match the orientation of the central atom. It does this by calculating the following quantity for each of the atoms in the system:

$$s_i = \frac{\sum_j \sigma(r_{ij}) \sum_{m=-3}^{3} q_{3m}^*(i) q_{3m}(j)}{\sum_j \sigma(r_{ij})}$$

where $q_{3m}(i)$ and $q_{3m}(j)$ are the 3rd order Steinhardt vectors calculated for atom i and atom j respectively and the asterix denotes complex conjugation. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise. The sum in the numerator of this expression is the dot product of the Steinhardt parameters for atoms i and j and thus measures the degree to which the orientations of these adjacent atoms is correlated.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

Compulsory keywords

DATA	the labels of the action that calculates the multicolvars we are interested in
NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.

SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i-\bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

The following command calculates the average value of the LOCAL_Q3 parameter for the 64 Lennard Jones atoms in the system under study and prints this quantity to a file called colvar.

```
Q3 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q3
```

```
LOCAL_Q3 ARG=q3 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LABEL=lq3
PRINT ARG=lq3.mean FILE=colvar
```

The following input calculates the distribution of LOCAL_Q3 parameters at any given time and outputs this information to a file.

```
Q3 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q3
LOCAL_Q3 ARG=q3 SWITCH={RATIONAL D_0=1.3 R_0=0.2} HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1}
PRINT ARG=lq3.* FILE=colvar
```

The following calculates the LOCAL_Q3 parameters for atoms 1-5 only. For each of these atoms comparisons of the geometry of the coordination sphere are done with those of all the other atoms in the system. The final quantity is the average and is outputted to a file

```
Q3 SPECIESA=1-5 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q3a
Q3 SPECIESA=6-64 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q3b

LOCAL_Q3 ARG=q3a,q3b SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LOWMEM LABEL=w3
PRINT ARG=w3.* FILE=colvar
```

5.5.22 LOCAL_Q4

	This is part of the crystallization module

Calculate the local degree of order around an atoms by taking the average dot product between the q_4 vector on the central atom and the q_4 vector on the atoms in the first coordination sphere.

The Q4 command allows one to calculate one complex vectors for each of the atoms in your system that describe the degree of order in the coordination sphere around a particular atom. The difficulty with these vectors comes when combining the order parameters from all of the individual atoms/molecules so as to get a measure of the global degree of order for the system. The simplest way of doing this - calculating the average Steinhardt parameter - can be problematic. If one is examining nucleation say only the order parameters for those atoms in the nucleus will change significantly when the nucleus forms. The order parameters for the atoms in the surrounding liquid will remain pretty much the same. As such if one models a small nucleus embedded in a very large amount of solution/melt any change in the average order parameter will be negligible. Substantial changes in the value of this average can be observed in simulations of nucleation but only because the number of atoms is relatively small.

When the average Q4 parameter is used to bias the dynamics a problems can occur. These averaged coordinates cannot distinguish between the correct, single-nucleus pathway and a concerted pathway in which all the atoms rearrange themselves into their solid-like configuration simultaneously. This second type of pathway would be impossible in reality because there is a large entropic barrier that prevents concerted processes like this from happening. However, in the finite sized systems that are commonly simulated this barrier is reduced substantially. As a result in simulations where average Steinhardt parameters are biased there are often quite dramatic system size effects

If one wants to simulate nucleation using some form on biased dynamics what is really required is an order parameter that measures:

- Whether or not the coordination spheres around atoms are ordered
- Whether or not the atoms that are ordered are clustered together in a crystalline nucleus

LOCAL_AVERAGE and NLINKS are variables that can be combined with the Steinhardt parameteters allow to calculate variables that satisfy these requirements. LOCAL_Q4 is another variable that can be used in these sorts of calculations. The LOCAL_Q4 parameter for a particular atom is a number that measures the extent to which the orientation of the atoms in the first coordination sphere of an atom match the orientation of the central atom. It does this by calculating the following quantity for each of the atoms in the system:

$$s_i = \frac{\sum_j \sigma(r_{ij}) \sum_{m=-4}^{4} q_{4m}^*(i) q_{4m}(j)}{\sum_j \sigma(r_{ij})}$$

where $q_{4m}(i)$ and $q_{4m}(j)$ are the 4th order Steinhardt vectors calculated for atom i and atom j respectively and the asterix denotes complex conjugation. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise. The sum in the numerator of this expression is the dot product of the Steinhardt parameters for atoms i and j and thus measures the degree to which the orientations of these adjacent atoms is correlated.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.

morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

Compulsory keywords

DATA	the labels of the action that calculates the multicolvars we are interested in
NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...

LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...
MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

The following command calculates the average value of the LOCAL_Q4 parameter for the 64 Lennard Jones atoms in the system under study and prints this quantity to a file called colvar.

```
Q4 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q4
LOCAL_Q4 ARG=q4 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LABEL=lq4
PRINT ARG=lq4.mean FILE=colvar
```

The following input calculates the distribution of LOCAL_Q4 parameters at any given time and outputs this information to a file.

```
Q4 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q4
LOCAL_Q4 ARG=q4 SWITCH={RATIONAL D_0=1.3 R_0=0.2} HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1}
PRINT ARG=lq4.* FILE=colvar
```

The following calculates the LOCAL_Q4 parameters for atoms 1-5 only. For each of these atoms comparisons of the geometry of the coordination sphere are done with those of all the other atoms in the system. The final quantity is the average and is outputted to a file

```
Q4 SPECIESA=1-5 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q4a
```

```
Q4 SPECIESA=6-64 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q4b

LOCAL_Q4 ARG=q4a,q4b SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LOWMEM LABEL=w4
PRINT ARG=w4.* FILE=colvar
```

5.5.23 LOCAL_Q6

	This is part of the crystallization module

Calculate the local degree of order around an atoms by taking the average dot product between the q_6 vector on the central atom and the q_6 vector on the atoms in the first coordination sphere.

The Q6 command allows one to calculate one complex vectors for each of the atoms in your system that describe the degree of order in the coordination sphere around a particular atom. The difficulty with these vectors comes when combining the order parameters from all of the individual atoms/molecules so as to get a measure of the global degree of order for the system. The simplest way of doing this - calculating the average Steinhardt parameter - can be problematic. If one is examining nucleation say only the order parameters for those atoms in the nucleus will change significantly when the nucleus forms. The order parameters for the atoms in the surrounding liquid will remain pretty much the same. As such if one models a small nucleus embedded in a very large amount of solution/melt any change in the average order parameter will be negligible. Substantial changes in the value of this average can be observed in simulations of nucleation but only because the number of atoms is relatively small.

When the average Q6 parameter is used to bias the dynamics a problems can occur. These averaged coordinates cannot distinguish between the correct, single-nucleus pathway and a concerted pathway in which all the atoms rearrange themselves into their solid-like configuration simultaneously. This second type of pathway would be impossible in reality because there is a large entropic barrier that prevents concerted processes like this from happening. However, in the finite sized systems that are commonly simulated this barrier is reduced substantially. As a result in simulations where average Steinhardt parameters are biased there are often quite dramatic system size effects

If one wants to simulate nucleation using some form on biased dynamics what is really required is an order parameter that measures:

- Whether or not the coordination spheres around atoms are ordered
- Whether or not the atoms that are ordered are clustered together in a crystalline nucleus

LOCAL_AVERAGE and NLINKS are variables that can be combined with the Steinhardt parameteters allow to calculate variables that satisfy these requirements. LOCAL_Q6 is another variable that can be used in these sorts of calculations. The LOCAL_Q6 parameter for a particular atom is a number that measures the extent to which the orientation of the atoms in the first coordination sphere of an atom match the orientation of the central atom. It does this by calculating the following quantity for each of the atoms in the system:

$$s_i = \frac{\sum_j \sigma(r_{ij}) \sum_{m=-6}^{6} q^*_{6m}(i) q_{6m}(j)}{\sum_j \sigma(r_{ij})}$$

where $q_{6m}(i)$ and $q_{6m}(j)$ are the 6th order Steinhardt vectors calculated for atom i and atom j respectively and the asterix denotes complex conjugation. The function $\sigma(r_{ij})$ is a switchingfunction that acts on the distance between atoms i and j. The parameters of this function should be set so that it the function is equal to one when atom j is in the first coordination sphere of atom i and is zero otherwise. The sum in the numerator of this expression is the dot product of the Steinhardt parameters for atoms i and j and thus measures the degree to which the orientations of these adjacent atoms is correlated.

Description of components

When the label of this action is used as the input for a second you are not referring to a scalar quantity as you are in regular collective variables. The label is used to reference the full set of quantities calculated by the action. This

is usual when using MultiColvar functions. Generally when doing this the previously calculated multicolvar will be referenced using the DATA keyword rather than ARG.

This Action can be used to calculate the following scalar quantities directly. These quantities are calculated by employing the keywords listed below. These quantities can then be referenced elsewhere in the input file by using this Action's label followed by a dot and the name of the quantity. Some amongst them can be calculated multiple times with different parameters. In this case the quantities calculated can be referenced elsewhere in the input by using the name of the quantity followed by a numerical identifier e.g. *label*.lessthan-1, *label*.lessthan-2 etc. When doing this and, for clarity we have made the label of the components customizable. As such by using the LABEL keyword in the description of the keyword input you can customize the component name

Quantity	Keyword	Description
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

Compulsory keywords

DATA	the labels of the action that calculates the multicolvars we are interested in

NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
MEAN	(default=off) take the mean of these variables. The final value can be referenced using *label*.mean

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.
MORE_THAN	calculate the number of variables more than a certain target value. This quantity is calculated using $\sum_i 1.0-\sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.more_than. You can use multiple instances of this keyword i.e. MORE_THAN1, MORE_THAN2, MORE_THAN3... The corresponding values are then referenced using *label*.more_than-1, *label*.more_than-2, *label*.more_than-3...
LESS_THAN	calculate the number of variables less than a certain target value. This quantity is calculated using $\sum_i \sigma(s_i)$, where $\sigma(s)$ is a switchingfunction. The final value can be referenced using *label*.less_than. You can use multiple instances of this keyword i.e. LESS_THAN1, LESS_THAN2, LESS_THAN3... The corresponding values are then referenced using *label*.less_than-1, *label*.less_than-2, *label*.less_than-3...

MIN	calculate the minimum value. To make this quantity continuous the minimum is calculated using $\min = \frac{\beta}{\log \sum_i \exp\left(\frac{\beta}{s_i}\right)}$ The value of β in this function is specified using (BETA= β) The final value can be referenced using *label*.min.
BETWEEN	calculate the number of values that are within a certain range. These quantities are calculated using kernel density estimation as described on histogrambead. The final value can be referenced using *label*.between. You can use multiple instances of this keyword i.e. BETWEEN1, BETWEEN2, BETWEEN3... The corresponding values are then referenced using *label*.between-1, *label*.between-2, *label*.between-3...
HISTOGRAM	calculate a discretized histogram of the distribution of values. This shortcut allows you to calculates NBIN quantites like BETWEEN.
MOMENTS	calculate the moments of the distribution of collective variables. The mth moment of a distribution is calculated using $\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^m$, where $\bar{s}$ is the average for the distribution. The moments keyword takes a lists of integers as input or a range. Each integer is a value of m. The final calculated values can be referenced using moment- m.

Examples

The following command calculates the average value of the LOCAL_Q6 parameter for the 64 Lennard Jones atoms in the system under study and prints this quantity to a file called colvar.

```
Q6 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q6
LOCAL_Q6 ARG=q6 SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LABEL=lq6
PRINT ARG=lq6.mean FILE=colvar
```

The following input calculates the distribution of LOCAL_Q6 parameters at any given time and outputs this information to a file.

```
Q6 SPECIES=1-64 D_0=1.3 R_0=0.2 LABEL=q6
LOCAL_Q6 ARG=q6 SWITCH={RATIONAL D_0=1.3 R_0=0.2} HISTOGRAM={GAUSSIAN LOWER=0.0 UPPER=1.0 NBINS=20 SMEAR=0.1}
PRINT ARG=lq6.* FILE=colvar
```

The following calculates the LOCAL_Q6 parameters for atoms 1-5 only. For each of these atoms comparisons of the geometry of the coordination sphere are done with those of all the other atoms in the system. The final quantity is the average and is outputted to a file

```
Q6 SPECIESA=1-5 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q6a
Q6 SPECIESA=6-64 SPECIESB=1-64 D_0=1.3 R_0=0.2 LABEL=q6b

LOCAL_Q6 ARG=q4a,q4b SWITCH={RATIONAL D_0=1.3 R_0=0.2} MEAN LOWMEM LABEL=w4
PRINT ARG=w6.* FILE=colvar
```

5.5.24 NLINKS

	This is part of the multicolvar module

Calculate number of pairs of atoms/molecules that are "linked"

In its simplest guise this coordinate calculates a coordination number. Each pair of atoms is assumed "linked" if they are within some cutoff of each other. In more complex applications each entity is a vector and this quantity measures whether pairs of vectors are (a) within a certain cutoff and (b) if the two vectors have similar orientations. The vectors on individual atoms could be Steinhardt parameters (see Q3, Q4 and Q6) or they could describe some internal vector in a molecule.

Compulsory keywords

DATA	the labels of the action that calculates the multicolvars we are interested in
NN	(default=6) The n parameter of the switching function
MM	(default=12) The m parameter of the switching function
D_0	(default=0.0) The d_0 parameter of the switching function
R_0	The r_0 parameter of the switching function

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords.

Examples

The following calculates how many bonds there are in a system containing 64 atoms and outputs this quantity to a file.

```
DENSITY SPECIES=1-64 LABEL=d1
NLINKS ARG=d1 SWITCH={RATIONAL D_0=1.3 R_0=0.2} LABEL=dd
PRINT ARG=dd FILE=colvar
```

The following calculates how many pairs of neighbouring atoms in a system containg 64 atoms have similar dispositions for the atoms in their coordination sphere. This calculation uses the dot product of the Q6 vectors on adjacent atoms to measure whether or not two atoms have the same “orientation"

```
Q6 SPECIES=1-64 SWITCH={RATIONAL D_0=1.3 R_0=0.2} LABEL=q6
NLINKS ARG=q6 SWITCH={RATIONAL D_0=1.3 R_0=0.2} LABEL=dd
PRINT ARG=dd FILE=colvar
```

5.5.25 SPRINT

	This is part of the multicolvar module

Calculate SPRINT topological variables.

The SPRINT topological variables are calculated from the largest eigenvalue, λ of an $n \times n$ adjacency matrix and its corresponding eigenvector, $\mathbf{V}$, using:

$$s_i = \sqrt{n}\lambda v_i$$

You can use different quantities to measure whether or not two given atoms/molecules are adjacent or not in the adjacency matrix. The simplest measure of adjacency is is whether two atoms/molecules are within some cutoff of each other. Further complexity can be added by insisting that two molecules are adjacent if they are within a certain distance of each other and if they have similar orientations.

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
coord	all $n $ sprint coordinates are calculated and then stored in increasing order. the smallest sprint coordinate will be labelled *label*.coord-1, the second smallest will be labelleled *label*.coord-1 and so on

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
vmean	**VMEAN**	the norm of the mean vector. The output component can be refererred to elsewhere in the input file by using the label.vmean
spath	**SPATH**	the position on the path
zpath	**ZPATH**	the distance from the path
dhenergy	**DHENERGY**	the Debye-Huckel interaction energy. You can calculate this quantity multiple times using different parameters
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.

lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
max	**MAX**	the maximum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
sum	**SUM**	the sum of values

Compulsory keywords

DATA	the labels of the action that calculates the multicolvars we are interested in

Options

NOPBC	(default=off) ignore the periodic boundary conditions when calculating distances
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements

TOL	this keyword can be used to speed up your calculation. When accumulating sums in which the individual terms are numbers inbetween zero and one it is assumed that terms less than a certain tolerance make only a small contribution to the sum. They can thus be safely ignored as can the the derivatives wrt these small quantities.
SWITCH	This keyword is used if you want to employ an alternative to the continuous swiching function defined above. The following provides information on the switchingfunction that are available. When this keyword is present you no longer need the NN, MM, D_0 and R_0 keywords. You can use multiple instances of this keyword i.e. SWITCH1, SWITCH2, SWITCH3...

Examples

This example input calculates the 7 SPRINT coordinates for a 7 atom cluster of Lennard-Jones atoms and prints their values to a file. In this input the SPRINT coordinates are calculated in the manner described in ?? so two atoms are adjacent if they are within a cutoff:

```
DENSITY SPECIES=1-7 LABEL=d1
SPRINT ARG=d1 SWITCH={RATIONAL R_0=0.1} LABEL=ss
PRINT ARG=ss.* FILE=colvar
```

This example input calculates the 14 SPRINT coordinates foa a molecule composed of 7 hydrogen and 7 carbon atoms. Once again two atoms are adjacent if they are within a cutoff:

```
DENSITY SPECIES=1-7 LABEL=c
DENSITY SPECIES=8-14 LABEL=h

SPRINT ...
 ARG=c,h
 SWITCH11={RATIONAL R_0=2.6 NN=6 MM=12}
 SWITCH12={RATIONAL R_0=2.2 NN=6 MM=12}
 SWITCH22={RATIONAL R_0=2.2 NN=6 MM=12}
 LABEL=ss
... SPRINT

PRINT ARG=ss.* FILE=colvar
```

5.5.26 UWALLS

	This is part of the manyrestraints module

Add UPPER_WALLS restraints on all the multicolvar values

This action takes the set of values calculated by the colvar specified by label in the DATA keyword and places a restraint on each quantity, x, with the following functional form:

$k((x-a+o)/s)^e$

k (KAPPA) is an energy constant in internal unit of the code, s (EPS) a rescaling factor and e (EXP) the exponent determining the power law. By default: EXP = 2, EPS = 1.0, OFF = 0.

Description of components

By default the value of the calculated quantity can be referenced elsewhere in the input file by using the label of the action. Alternatively this Action can be used to be used to calculate the following quantities by employing the keywords listed below. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potentials

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
vmean	**VMEAN**	the norm of the mean vector. The output component can be refererred to elsewhere in the input file by using the label.vmean
spath	**SPATH**	the position on the path
zpath	**ZPATH**	the distance from the path
dhenergy	**DHENERGY**	the Debye-Huckel interaction energy. You can calculate this quantity multiple times using different parameters
between	**BETWEEN**	the number/fraction of values within a certain range. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
lessthan	**LESS_THAN**	the number of values less than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
max	**MAX**	the maximum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.
mean	**MEAN**	the mean value. The output component can be refererred to elsewhere in the input file by using the label.mean
min	**MIN**	the minimum value. This is calculated using the formula described in the description of the keyword so as to make it continuous.

moment	**MOMENTS**	the central moments of the distribution of values. The second moment would be referenced elsewhere in the input file using *label*.moment-2, the third as *label*.moment-3, etc.
morethan	**MORE_THAN**	the number of values more than a target value. This is calculated using one of the formula described in the description of the keyword so as to make it continuous. You can calculate this quantity multiple times using different parameters.
sum	**SUM**	the sum of values

Compulsory keywords

DATA	certain actions in plumed work by calculating a list of variables and summing over them. This particular action can be used to calculate functions of these base variables or prints them to a file. This keyword thus takes the label of one of those such variables as input.
AT	the radius of the sphere
KAPPA	the force constant for the wall. The k_i in the expression for a wall.
OFFSET	(default=0.0) the offset for the start of the wall. The o_i in the expression for a wall.
EXP	(default=2.0) the powers for the walls. The e_i in the expression for a wall.
EPS	(default=1.0) the values for s_i in the expression for a wall

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements

Examples

The following set of commands can be used to stop a cluster composed of 20 atoms subliming. The position of the centre of mass of the cluster is calculated by the COM command labelled c1. The DISTANCES command labelled d1 is then used to calculate the distance between each of the 20 atoms in the cluster and the center of mass of the cluster. These distances are then passed to the UWALLS command, which adds a UPPER_WALLS restraint on each of them and thereby prevents each of them from moving very far from the centre of mass of the cluster.

```
COM ATOMS=1-20 LABEL=c1
DISTANCES GROUPA=c1 GROUPB=1-20 LABEL=d1
UWALLS DATA=d1 AT=2.5 KAPPA=0.2 LABEL=sr
```

Chapter 6

Analysis

PLUMED can be used to analyse trajectories either on the fly during an MD run or via postprocessing a trajectory using driver. The following is a list of the various methods for analysing trajectories contained in PLUMED.

CLASSICAL_MDS	Create a low-dimensional projection of a trajectory using the classical multidimensional scaling algorithm.
COMMITTOR	Does a committor analysis.
DUMPATOMS	Dump selected atoms on a file.
DUMPDERIVATIVES	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
DUMPFORCES	Dump the force acting on one of a values in a file.
DUMPMULTICOLVAR	Dump atom positions and multicolvar on a file.
DUMPPROJECTIONS	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
HISTOGRAM	Calculate the probability density as a function of a few CVs either using kernel density estimation, or a discretehistogram estimation.
PRINT	Print quantities to a file.

6.1 CLASSICAL_MDS

	This is part of the analysis module

Create a low-dimensional projection of a trajectory using the classical multidimensional scaling algorithm.

Multidimensional scaling (MDS) is similar to what is done when you make a map. You start with distances between London, Belfast, Paris and Dublin and then you try to arrange points on a piece of paper so that the (suitably scaled) distances between the points in your map representing each of those cities are related to the true distances between the cities. Stating this more mathematically MDS endeavors to find an `isometry` between points distributed in a high-dimensional space and a set of points distributed in a low-dimensional plane. In other words, if we have M D-dimensional points, $\mathbf{X}$, and we can calculate dissimilarities between pairs them, D_{ij}, we can, with an MDS calculation, try to create M projections, $\mathbf{x}$, of the high dimensionality points in a d-dimensional linear space by trying to arrange the projections so that the Euclidean distances between pairs of them, d_{ij}, resemble the dissimilarities between the high dimensional points. In short we minimize:

$$\chi^2 = \sum_{i \neq j} (D_{ij} - d_{ij})^2$$

where D_{ij} is the distance between point X^i and point X^j and d_{ij} is the distance between the projection of X^i, x^i, and the projection of X^j, x^j. A tutorial on this approach can be used to analyse simulations can be found in the tutorial Belfast tutorial: Adaptive variables II and in the following `short video.`

The atoms involved can be specified using

ATOMS	the atoms whose positions we are tracking for the purpose of analysing the data. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

METRIC	(default=EUCLIDEAN) how are we measuring the distances between configurations
STRIDE	(default=1) the frequency with which data should be stored for analysis
RUN	the frequency with which to run the analysis algorithm. This is not required if you specify USE_ALL_DATA
LANDMARKS	(default=ALL) only use a subset of the data that was collected. For more information on the landmark selection algorithms that are available in plumed see landmarkselection.
NLOW_DIM	number of low-dimensional coordinates required
OUTPUT_FILE	file on which to output the final embedding coordinates
EMBEDDING_OFILE	(default=dont output) file on which to output the embedding in plumed input format

Options

USE_ALL_DATA	(default=off) use the data from the entire trajectory to perform the analysis
REWEIGHT_BIAS	(default=off) reweight the data using all the biases acting on the dynamics. For more information see reweighting.
WRITE_CHECKPOINT	(default=off) write out a checkpoint so that the analysis can be restarted in a later run
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.
FMT	the format that should be used in analysis output files
TEMP	the system temperature. This is required if you are reweighting or doing free energies.
REWEIGHT_TEMP	reweight data from a trajectory at one temperature and output the probability distribution at a second temperature. For more information see reweighting. This is not possible during postprocessing.

Examples

The following command instructs plumed to construct a classical multidimensional scaling projection of a trajectory. The RMSD distance between atoms 1-256 have moved is used to measure the distances in the high-dimensional space.

```
CLASSICAL_MDS ...
  ATOMS=1-256
  METRIC=OPTIMAL-FAST
  USE_ALL_DATA
  NLOW_DIM=2
  OUTPUT_FILE=rmsd-embed
... CLASSICAL_MDS
```

The following section is for people who are interested in how this method works in detail. A solid understanding of this material is not necessary to use MDS.

6.1.1 Method of optimisation

The stress function can be minimized using a standard optimization algorithm such as conjugate gradients or steepest descent. However, it is more common to do this minimization using a technique known as classical scaling. Classical scaling works by recognizing that each of the distances D_{ij} in the above sum can be written as:

$$D_{ij}^2 = \sum_\alpha (X_\alpha^i - X_\alpha^j)^2 = \sum_\alpha (X_\alpha^i)^2 + (X_\alpha^j)^2 - 2X_\alpha^i X_\alpha^j$$

We can use this expression and matrix algebra to calculate multiple distances at once. For instance if we have three points, $\mathbf{X}$, we can write distances between them as:

$$
\begin{aligned}
D^2(\mathbf{X}) &= \begin{bmatrix} 0 & d_{12}^2 & d_{13}^2 \\ d_{12}^2 & 0 & d_{23}^2 \\ d_{13}^2 & d_{23}^2 & 0 \end{bmatrix} \\
&= \sum_\alpha \begin{bmatrix} (X_\alpha^1)^2 & (X_\alpha^1)^2 & (X_\alpha^1)^2 \\ (X_\alpha^2)^2 & (X_\alpha^2)^2 & (X_\alpha^2)^2 \\ (X_\alpha^3)^2 & (X_\alpha^3)^2 & (X_\alpha^3)^2 \end{bmatrix} + \sum_\alpha \begin{bmatrix} (X_\alpha^1)^2 & (X_\alpha^2)^2 & (X_\alpha^3)^2 \\ (X_\alpha^1)^2 & (X_\alpha^2)^2 & (X_\alpha^3)^2 \\ (X_\alpha^1)^2 & (X_\alpha^2)^2 & (X_\alpha^3)^2 \end{bmatrix} - 2\sum_\alpha \begin{bmatrix} X_\alpha^1 X_\alpha^1 & X_\alpha^1 X_\alpha^2 & X_\alpha^1 X_\alpha^3 \\ X_\alpha^2 X_\alpha^1 & X_\alpha^2 X_\alpha^2 & X_\alpha^2 X_\alpha^3 \\ X_\alpha^1 X_\alpha^3 & X_\alpha^3 X_\alpha^2 & X_\alpha^3 X_\alpha^3 \end{bmatrix} \\
&= \mathbf{c1^T} + \mathbf{1c^T} - 2\sum_\alpha \mathbf{x}_a \mathbf{x}_a^T = \mathbf{c1^T} + \mathbf{1c^T} - 2\mathbf{XX^T}
\end{aligned}
$$

This last equation can be extended to situations when we have more than three points. In it $\mathbf{X}$ is a matrix that has one high-dimensional point on each of its rows and $\mathbf{X^T}$ is its transpose. $\mathbf{1}$ is an $M \times 1$ vector of ones and $\mathbf{c}$ is a vector with components given by:

$$
c_i = \sum_\alpha (x_\alpha^i)^2
$$

These quantities are the diagonal elements of $\mathbf{XX^T}$, which is a dot product or Gram Matrix that contains the dot product of the vector X_i with the vector X_j in element i, j.

In classical scaling we introduce a centering matrix $\mathbf{J}$ that is given by:

$$
\mathbf{J} = \mathbf{I} - \frac{1}{M}\mathbf{11^T}
$$

where $\mathbf{I}$ is the identity. Multiplying the equations above from the front and back by this matrix and a factor of a $-\frac{1}{2}$ gives:

$$
\begin{aligned}
-\frac{1}{2}\mathbf{JD}^2(\mathbf{X})\mathbf{J} &= -\frac{1}{2}\mathbf{J}(\mathbf{c1^T} + \mathbf{1c^T} - 2\mathbf{XX^T})\mathbf{J} \\
&= -\frac{1}{2}\mathbf{Jc1^T J} - \frac{1}{2}\mathbf{J1c^T J} + \frac{1}{2}\mathbf{J}(2\mathbf{XX^T})\mathbf{J} \\
&= \mathbf{JXX^T J} = \mathbf{XX^T}
\end{aligned}
$$

The fist two terms in this expression disappear because $\mathbf{1^T J} = \mathbf{J1} = \mathbf{0}$, where $\mathbf{0}$ is a matrix containing all zeros. In the final step meanwhile we use the fact that the matrix of squared distances will not change when we translate all the points. We can thus assume that the mean value, μ, for each of the components, α:

$$
\mu_\alpha = \frac{1}{M}\sum_{i=1}^{N} \mathbf{X}_\alpha^i
$$

is equal to 0 so the columns of $\mathbf{X}$ add up to 0. This in turn means that each of the columns of $\mathbf{XX^T}$ adds up to zero, which is what allows us to write $\mathbf{JXX^T J} = \mathbf{XX^T}$.

The matrix of squared distances is symmetric and positive-definite we can thus use the spectral decomposition to decompose it as:

$$
\Phi = \mathbf{V}\Lambda\mathbf{V}^T
$$

Furthermore, because the matrix we are diagonalizing, $\mathbf{XX^T}$, is the product of a matrix and its transpose we can use this decomposition to write:

$$
\mathbf{X} = \mathbf{V}\Lambda^{\frac{1}{2}}
$$

Much as in PCA there are generally a small number of large eigenvalues in Λ and many small eigenvalues. We can safely use only the large eigenvalues and their corresponding eigenvectors to express the relationship between the coordinates $\mathbf{X}$. This gives us our set of low-dimensional projections.

This derivation makes a number of assumptions about the how the low dimensional points should best be arranged to minimise the stress. If you use an interative optimization algorithm such as SMACOF you may thus be able to find a better (lower-stress) projection of the points. For more details on the assumptions made see this website.

6.2 COMMITTOR

	This is part of the analysis module

Does a committor analysis.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

STRIDE	(default=1) the frequency with which the CVs are analysed
BASIN_A_LOWER	the lower bounds of Basin A
BASIN_A_UPPER	the upper bounds of Basin A
BASIN_B_LOWER	the lower bounds of Basin B
BASIN_B_UPPER	the upper bounds of Basin B

FILE	the name of the file on which to output these quantities
FMT	the format that should be used to output real numbers

Examples

The following input monitors two torsional angles during a simulation, defines two basins (A and B) as a function of the two torsions and stops the simulation when it falls in one of the two. In the log file will be shown the latest values for the CVs and the basin reached.

```
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
COMMITTOR ...
  ARG=r1,r2
  STRIDE=10
  BASIN_A_LOWER=0.15,0.20
  BASIN_A_UPPER=0.25,0.40
  BASIN_B_LOWER=-0.15,-0.20
  BASIN_B_UPPER=-0.25,-0.40
... COMMITTOR
```

6.3 DUMPATOMS

	This is part of the generic module

Dump selected atoms on a file.

This command can be used to output the positions of a particular set of atoms. The atoms required are ouput in a xyz or gro formatted file. The type of file is automatically detected from the file extension, but can be also enforced with TYPE. Importantly, if your input file contains actions that edit the atoms position (e.g. WHOLEMOLECULES) and the DUMPATOMS command appears after this instruction, then the edited atom positions are output. You can control the buffering of output using the FLUSH keyword on a separate line.

Units of the printed file can be controlled with the UNITS keyword. By default PLUMED units as controlled in the UNITS command are used, but one can override it e.g. with UNITS=A. Notice that gro files can only contain coordinates in nm.

The atoms involved can be specified using

ATOMS	the atom indices whose positions you would like to print out. For more information on how to specify lists of atoms see Groups and Virtual Atoms

Compulsory keywords

STRIDE	(default=1) the frequency with which the atoms should be output

FILE	file on which to output coordinates. .gro extension is automatically detected
UNITS	(default=PLUMED) the units in which to print out the coordinates. PLUMED means internal PLUMED units

PRECISION	The number of digits in trajectory file
TYPE	file type, either xyz or gro, can override an automatically detected file extension

Examples

The following input instructs plumed to print out the positions of atoms 1-10 together with the position of the center of mass of atoms 11-20 every 10 steps to a file called file.xyz.

```
COM ATOMS=11-20 LABEL=c1
DUMPATOMS STRIDE=10 FILE=file.xyz ATOMS=1-10,c1
```

(see also COM)

The following input is very similar but dumps a .gro (gromacs) file, which also contains atom and residue names.

```
# this is required to have proper atom names:
MOLINFO STRUCTURE=reference.pdb
# if omitted, atoms will have "X" name...

COM ATOMS=11-20 LABEL=c1
DUMPATOMS STRIDE=10 FILE=file.gro ATOMS=1-10,c1
# notice that last atom is a virtual one and will not have
# a correct name in the resulting gro file
```

(see also COM and MOLINFO)

6.4 DUMPDERIVATIVES

	This is part of the generic module

Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).

For a CV this line in input instructs plumed to print the derivative of the CV with respect to the atom positions and the cell vectors (virial-like form). In contrast, for a function or bias the derivative with respect to the input "CVs" will be output. This command is most often used to test whether or not analytic derivatives have been implemented correctly. This can be done by outputting the derivatives calculated analytically and numerically. You can control the buffering of output using the FLUSH keyword.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.
STRIDE	(default=1) the frequency with which the derivatives should be output
FILE	the name of the file on which to output the derivatives
FMT	(default=%15.10f) the format with which the derivatives should be output

Examples

The following input instructs plumed to write a file called deriv that contains both the analytical and numerical derivatives of the distance between atoms 1 and 2.

```
DISTANCE ATOM=1,2 LABEL=distance
DISTANCE ATOM=1,2 LABEL=distanceN NUMERICAL_DERIVATIVES
DUMPDERIVATIVES ARG=distance,distanceN STRIDE=1 FILE=deriv
```

(See also DISTANCE)

6.5 DUMPFORCES

	This is part of the generic module

Dump the force acting on one of a values in a file.

For a CV this command will dump the force on the CV itself. Be aware that in order to have the forces on the atoms you should multiply the output from this argument by the output from DUMPDERIVATIVES. Furthermore, also note that you can output the forces on multiple quantities simultaneously by specifying more than one argument. You can control the buffering of output using the FLUSH keyword.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If ∗ or ∗.∗ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.∗.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.
STRIDE	(default=1) the frequency with which the forces should be output
FILE	the name of the file on which to output the forces

Examples

The following input instructs plumed to write a file called forces that contains the force acting on the distance between atoms 1 and 2.

```
DISTANCE ATOM=1,2 LABEL=distance
DUMPFORCES ARG=distance STRIDE=1 FILE=forces
```

(See also DISTANCE)

6.6 DUMPMULTICOLVAR

	This is part of the multicolvar module

Dump atom positions and multicolvar on a file.

Compulsory keywords

DATA	certain actions in plumed work by calculating a list of variables and summing over them. This particular action can be used to calculate functions of these base variables or prints them to a file. This keyword thus takes the label of one of those such variables as input.

STRIDE	(default=1) the frequency with which the atoms should be output
FILE	file on which to output coordinates
UNITS	(default=PLUMED) the units in which to print out the coordinates. PLUMED means internal PLUMED units

PRECISION	The number of digits in trajectory file

Examples

In this examples we calculate the distances between the atoms of the first and the second group and we write them in the file MULTICOLVAR.xyz. For each couple it writes the coordinates of their geometric center and their distance.

```
pos:   GROUP ATOMS=220,221,235,236,247,248,438,439,450,451,534,535
neg:   GROUP ATOMS=65,68,138,182,185,267,270,291,313,316,489,583,621,711
DISTANCES GROUPA=pos GROUPB=neg LABEL=slt

DUMPMULTICOLVAR DATA=slt FILE=MULTICOLVAR.xyz
```

(see also DISTANCES)

6.7 DUMPPROJECTIONS

	This is part of the generic module

Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).

6.8 HISTOGRAM

	This is part of the analysis module

Calculate the probability density as a function of a few CVs either using kernel density estimation, or a discrete histogram estimation.

In case a kernel density estimation is used the probability density is estimated as a continuos function on the grid with a BANDWIDTH defined by the user. In this case the normalisation is such that the INTEGRAL over the grid is 1. In case a discrete density estimation is used the probabilty density is estimated as a discrete function on the grid. In this case the normalisation is such that the SUM of over the grid is 1.

Additional material and examples can be also found in the tutorial Belfast tutorial: Analyzing CVs.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.
STRIDE	(default=1) the frequency with which data should be stored for analysis
RUN	the frequency with which to run the analysis algorithm. This is not required if you specify USE_ALL_DATA
GRID_MIN	the lower bounds for the grid
GRID_MAX	the upper bounds for the grid
KERNEL	(default=gaussian) the kernel function you are using. Use discrete/DISCRETE if you want to accumulate a discrete histogram. More details on the kernels available in plumed can be found in kernelfunctions.
GRID_WFILE	(default=histogram) the file on which to write the grid

Options

USE_ALL_DATA	(default=off) use the data from the entire trajectory to perform the analysis
REWEIGHT_BIAS	(default=off) reweight the data using all the biases acting on the dynamics. For more information see reweighting.
WRITE_CHECKPOINT	(default=off) write out a checkpoint so that the analysis can be restarted in a later run
SERIAL	(default=off) do the calculation in serial. Do not parallelize
LOWMEM	(default=off) lower the memory requirements
FREE-ENERGY	(default=off) Set to TRUE if you want a FREE ENERGY instead of a probabilty density (you need to set TEMP).
NOMEMORY	(default=off) analyse each block of data separately

FMT	the format that should be used in analysis output files
TEMP	the system temperature. This is required if you are reweighting or doing free energies.

REWEIGHT_TEMP	reweight data from a trajectory at one temperature and output the probability distribution at a second temperature. For more information see reweighting. This is not possible during postprocessing.
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)
BANDWIDTH	the bandwdith for kernel density estimation

Examples

The following input monitors two torsional angles during a simulation and outputs a continuos histogram as a function of them at the end of the simulation.

```
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  USE_ALL_DATA
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  GRID_WFILE=histo
... HISTOGRAM
```

The following input monitors two torsional angles during a simulation and outputs a discrete histogram as a function of them at the end of the simulation.

```
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  USE_ALL_DATA
  KERNEL=discrete
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  GRID_WFILE=histo
... HISTOGRAM
```

The following input monitors two torsional angles during a simulation and outputs the histogram accumulated thus far every 100000 steps.

```
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  RUN=100000
  GRID_MIN=-3.14,-3.14
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  GRID_WFILE=histo
... HISTOGRAM
```

The following input monitors two torsional angles during a simulation and outputs a separate histogram for each 100000 steps worth of trajectory.

```
TORSION ATOMS=1,2,3,4 LABEL=r1
TORSION ATOMS=2,3,4,5 LABEL=r2
HISTOGRAM ...
  ARG=r1,r2
  RUN=100000 NOMEMORY
  GRID_MIN=-3.14,-3.14
```

```
  GRID_MAX=3.14,3.14
  GRID_BIN=200,200
  BANDWIDTH=0.05,0.05
  GRID_WFILE=histo
... HISTOGRAM
```

Bug Option FREE-ENERGY without USE_ALL_DATA is not working properly. See #175.

6.9 PRINT

	This is part of the generic module

Print quantities to a file.

This directive can be used multiple times in the input so you can print files with different strides or print different quantities to different files. You can control the buffering of output using the FLUSH keyword.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

STRIDE	(default=1) the frequency with which the quantities of interest should be output

FILE	the name of the file on which to output these quantities
FMT	the format that should be used to output real numbers

Examples

The following input instructs plumed to print the distance between atoms 3 and 5 on a file called COLVAR every 10 steps, and the distance and total energy on a file called COLVAR_ALL every 1000 steps.

```
DISTANCE ATOMS=2,5 LABEL=distance
ENERGY             LABEL=energy
PRINT ARG=distance          STRIDE=10   FILE=COLVAR
PRINT ARG=distance,energy   STRIDE=1000 FILE=COLVAR_ALL
```

(See also DISTANCE and ENERGY).

6.9.1 FLUSH

	This is part of the generic module

This command instructs plumed to flush all the open files with a user specified frequency. Notice that all files are flushed anyway every 10000 steps.

This is useful for preventing data loss that would otherwise arrise as a consequence of the code storing data for printing in the buffers. Notice that wherever it is written in the plumed input file, it will flush all the open files.

Compulsory keywords

STRIDE	the frequency with which all the open files should be flushed

Examples

A command like this in the input will instruct plumed to flush all the output files every 100 steps

```
d1: DISTANCE ATOMS=1,10
PRINT ARG=d1 STRIDE=5 FILE=colvar1

FLUSH STRIDE=100

d2: DISTANCE ATOMS=2,11
# also this print is flushed every 100 steps:
PRINT ARG=d2 STRIDE=10 FILE=colvar2
```

(see also DISTANCE and PRINT).

Chapter 7

Bias

PLUMED allows you to run a number of enhanced sampling algorithms. The list of enhanced sampling algorithms contained in PLUMED is as follows:

ABMD	Adds a ratchet-and-pawl like restraint on one or more variables.
BIASVALUE	Takes the value of one variable and use it as a bias
EXTERNAL	Calculate a restraint that is defined on a grid that is read during start up
LOWER_WALLS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
METAD	Used to performed MetaDynamics on one or more collective variables.
MOVINGRESTRAINT	Add a time-dependent, harmonic restraint on one or more variables.
RESTRAINT	Adds harmonic and/or linear restraints on one or more variables.
UPPER_WALLS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.

Methods, such as METAD, that work by introducing a history dependent bias can be restarted using the RESTART keyword

You can also use PLUMED in conjunction with VMD's interactive MD module by taking advantage of the IMD action.

7.1 ABMD

	This is part of the bias module

Adds a ratchet-and-pawl like restraint on one or more variables.

This action can be used to evolve a system towards a target value in CV space using an harmonic potential moving with the thermal fluctuations of the CV [19] [20] [21]. The biasing potential in this method is as follows:

$$V(\rho(t)) = \begin{cases} \frac{K}{2}\left(\rho(t)-\rho_m(t)\right)^2, & \rho(t) > \rho_m(t) \\ 0, & \rho(t) \leq \rho_m(t), \end{cases}$$

where

$$\rho(t) = (CV(t) - TO)^2$$

and

$$\rho_m(t) = \min_{0\leq\tau\leq t} \rho(\tau) + \eta(t).$$

The method is based on the introduction of a biasing potential which is zero when the system is moving towards the desired arrival point and which damps the fluctuations when the system attempts to move in the opposite direction. As in the case of the ratchet and pawl system, propelled by thermal motion of the solvent molecules, the biasing potential does not exert work on the system. $\eta(t)$ is an additional white noise acting on the minimum position of the bias.

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential
_min	one or multiple instances of this quantity will be refereceable elsewhere in the input file. These quantities will be named with the arguments of the bias followed by the character string _min. These quantities tell the user the minimum value assumed by rho_m(t).

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

TO	The array of target values
KAPPA	The array of force constants.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

MIN	Array of starting values for the bias (set rho_m(t), otherwise it is set using the current value of ARG)
NOISE	Array of white noise intensities (add a temperature to the ABMD)
SEED	Array of seeds for the white noise (add a temperature to the ABMD)

Examples

The following input sets up two biases, one on the distance between atoms 3 and 5 and another on the distance between atoms 2 and 4. The two target values are defined using TO and the two strength using KAPPA. The total energy of the bias is printed.

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
ABMD ARG=d1,d2 TO=1.0,1.5 KAPPA=5.0,5.0 LABEL=abmd
PRINT ARG=abmd.bias,abmd.d1_min,abmd.d2_min
```

(See also DISTANCE and PRINT).

7.2 BIASVALUE

	This is part of the bias module

Takes the value of one variable and use it as a bias

This is the simplest possible bias: the bias potential is equal to a collective variable. It is useful to create custom biasing potential, e.g. applying a function (see Functions) to some collective variable then using the value of this function directly as a bias.

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
_bias	one or multiple instances of this quantity will be refereceable elsewhere in the input file. these quantities will named with the arguments of the bias followed by the character string _bias. These quantities tell the user how much the bias is due to each of the colvars.

bias	total bias

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input tells plumed to use the value of the distance between atoms 3 and 5 and the value of the distance between atoms 2 and 4 as biases. It then tells plumed to print the energy of the restraint

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=3,6 LABEL=d2
BIASVALUE ARG=d1,d2 LABEL=b
PRINT ARG=d1,d2,b.d1,b.d2
```

(See also DISTANCE and PRINT).

Another thing one can do is asking one system to follow a circle in sin/cos according a time dependence

```
t: TIME
# this just print cos and sin of time
cos: MATHEVAL ARG=t VAR=t FUNC=cos(t) PERIODIC=NO
sin: MATHEVAL ARG=t VAR=t FUNC=sin(t) PERIODIC=NO
c1: COM ATOMS=1,2
c2: COM ATOMS=3,4
d: DISTANCE COMPONENTS ATOMS=c1,c2
PRINT ARG=t,cos,sin,d.x,d.y,d.z STRIDE=1 FILE=colvar FMT=%8.4f
# this calculates sine and cosine of a projected component of distance
mycos:  MATHEVAL ARG=d.x,d.y  VAR=x,y   FUNC=x/sqrt(x*x+y*y) PERIODIC=NO
mysin:  MATHEVAL ARG=d.x,d.y  VAR=x,y   FUNC=y/sqrt(x*x+y*y) PERIODIC=NO
```

```
# this creates a moving spring so that the system follows a circle-like dynamics
# but it is not a bias, it is a simple value now
vv1:  MATHEVAL ARG=mycos,mysin,cos,sin VAR=mc,ms,c,s  FUNC=100*((mc-c)^2+(ms-s)^2) PERIODIC=NO
# this takes the value calculated with matheval and uses as a bias
cc: BIASVALUE ARG=vv1
# some printout
PRINT ARG=t,cos,sin,d.x,d.y,d.z,mycos,mysin,cc.bias.vv1 STRIDE=1 FILE=colvar FMT=%8.4f
```

(see also TIME, MATHEVAL, COM, DISTANCE, and PRINT).

7.3 EXTERNAL

	This is part of the bias module

Calculate a restraint that is defined on a grid that is read during start up

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	**Description**
bias	the instantaneous value of the bias potential

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

FILE	the name of the file containing the external potential.

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
NOSPLINE	(default=off) specifies that no spline interpolation is to be used when calculating the energy and forces due to the external potential
SPARSE	(default=off) specifies that the external potential uses a sparse grid

Examples

The following is an input for a calculation with an external potential that is defined in the file bias.dat and that acts on the distance between atoms 3 and 5.

```
DISTANCE ATOMS=3,5 LABEL=d1
EXTERNAL ARG=d1 FILENAME=bias.dat LABEL=external
```

(See also DISTANCE PRINT).

The header in the file bias.dat should read:

```
#! FIELDS d1 external.bias der_d1
#! SET min_d1 0.0
#! SET max_d1 1.0
#! SET nbins_d1 100
#! SET periodic_d1 false
```

This should then be followed by the value of the potential and its derivative at 100 equally spaced points along the distance between 0 and 1. If you run with NOSPLINE you do not need to provide derivative information.

You can also include grids that are a function of more than one collective variable. For instance the following would be the input for an external potential acting on two torsional angles:

```
TORSION ATOMS=4,5,6,7 LABEL=t1
TORSION ATOMS=6,7,8,9 LABEL=t2
EXTERNAL ARG=t1,t2 FILENAME=bias.dat LABEL=ext
```

The header in the file bias.dat for this calculation would read:

```
#! FIELDS t1 t2 ext.bias der_t1 der_t2
#! SET min_t1 -pi
#! SET max_t1 +pi
#! SET nbins_t1 100
#! SET periodic_t1 true
#! SET min_t2 -pi
#! SET max_t2 +pi
#! SET nbins_t2 100
#! SET periodic_t2 true
```

This would be then followed by 100 blocks of data. In the first block of data the value of t1 (the value in the first column) is kept fixed and the value of the function is given at 100 equally spaced values for t2 between $-pi$ and $+pi$. In the second block of data t1 is fixed at $-pi+\frac{2pi}{100}$ and the value of the function is given at 100 equally spaced values for t2 between $-pi$ and $+pi$. In the third block of data the same is done but t1 is fixed at $-pi+\frac{4pi}{100}$ and so on untill you get to the 100th block of data where t1 is fixed at $+pi$.

Please note the order that the order of arguments in the plumed.dat file must be the same as the order of arguments in the header of the grid file.

7.4 LOWER_WALLS

	This is part of the bias module

Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.

The restraining potential starts acting on the system when the value of the CV is greater (in the case of UPPER_WALLS) or lower (in the case of LOWER_WALLS) than a certain limit a_i (AT) minus an offset o_i (OFFSET). The expression for the bias due to the wall is given by:

$\sum_i k_i((x_i - a_i + o_i)/s_i)_i^e$

k_i (KAPPA) is an energy constant in internal unit of the code, s_i (EPS) a rescaling factor and e_i (EXP) the exponent determining the power law. By default: EXP = 2, EPS = 1.0, OFFSET = 0.

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

AT	the positions of the wall. The a_i in the expression for a wall.
KAPPA	the force constant for the wall. The k_i in the expression for a wall.
OFFSET	(default=0.0) the offset for the start of the wall. The o_i in the expression for a wall.
EXP	(default=2.0) the powers for the walls. The e_i in the expression for a wall.
EPS	(default=1.0) the values for s_i in the expression for a wall

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input tells plumed to add both a lower and an upper walls on the distance between atoms 3 and 5 and the distance between atoms 2 and 4. The lower and upper limits are defined at different values. The strength of the walls is the same for the four cases. It also tells plumed to print the energy of the walls.

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
UPPER_WALLS ARG=d1,d2 AT=1.0,1.5 KAPPA=150.0,150.0 EXP=2,2 EPS=1,1 OFFSET=0,0 LABEL=uwall
LOWER_WALLS ARG=d1,d2 AT=0.0,1.0 KAPPA=150.0,150.0 EXP=2,2 EPS=1,1 OFFSET=0,0 LABEL=lwall
PRINT ARG=uwall.bias,lwall.bias
```

(See also DISTANCE and PRINT).

7.5 METAD

	This is part of the bias module

Used to performed MetaDynamics on one or more collective variables.

In a metadynamics simulations a history dependent bias composed of intermittently added Gaussian functions is added to the potential [22].

$$V(\vec{s},t) = \sum_{k\tau<t} W(k\tau) \exp\left(-\sum_{i=1}^{d} \frac{(s_i - s_i^{(0)}(k\tau))^2}{2\sigma_i^2}\right).$$

This potential forces the system away from the kinetic traps in the potential energy surface and out into the unexplored parts of the energy landscape. Information on the Gaussian functions from which this potential is composed is output to a file called HILLS, which is used both the restart the calculation and to reconstruct the free energy as a function of the CVs. The free energy can be reconstructed from a metadynamics calculation because the final bias is given by:

$$V(\vec{s}) = -F(\vec{(}s))$$

During post processing the free energy can be calculated in this way using the sum_hills utility.

In the simplest possible implementation of a metadynamics calculation the expense of a metadynamics calculation increases with the length of the simulation as one has to, at every step, evaluate the values of a larger and larger number of Gaussians. To avoid this issue you can store the bias on a grid. This approach is similar to that proposed in [23] but has the advantage that the grid spacing is independent on the Gaussian width. Notice that you should

provide either the number of bins for every collective variable (GRID_BIN) or the desired grid spacing (GRID_SP↩ ACING). In case you provide both PLUMED will use the most conservative choice (highest number of bins) for each dimension. In case you do not provide any information about bin size (neither GRID_BIN nor GRID_SPACING) and if Gaussian width is fixed PLUMED will use 1/5 of the Gaussian width as grid spacing. This default choice should be reasonable for most applications.

Another option that is available in plumed is well-tempered metadynamics [24]. In this varient of metadynamics the heights of the Gaussian hills are rescaled at each step so the bias is now given by:

$$V(s,t) = \sum_{t'=0,\tau_G,2\tau_G,\ldots}^{t'<t} We^{-V(s(q(t'),t')/\Delta T} \exp\left(-\sum_{i=1}^{d}\frac{(s_i(q)-s_i(q(t'))^2}{2\sigma_i^2}\right),$$

This method ensures that the bias converges more smoothly. It should be noted that, in the case of well-tempered metadynamics, in the output printed the Gaussian height is re-scaled using the bias factor. Also notice that with well-tempered metadynamics the HILLS file does not contain the bias, but the negative of the free-energy estimate. This choice has the advantage that one can restart a simulation using a different value for the ΔT. The applied bias will be scaled accordingly.

Note that you can use here also the flexible gaussian approach [25] in which you can adapt the gaussian to the extent of Cartesian space covered by a variable or to the space in collective variable covered in a given time. In this case the width of the deposited gaussian potential is denoted by one value only that is a Cartesian space (ADAP↩ TIVE=GEOM) or a time (ADAPTIVE=DIFF). Note that a specific integration technique for the deposited gaussians should be used in this case. Check the documentation for utility sum_hills.

With the keyword INTERVAL one changes the metadynamics algorithm setting the bias force equal to zero outside boundary [26]. If, for example, metadynamics is performed on a CV s and one is interested only to the free energy for s $>$ sw, the history dependent potential is still updated according to the above equations but the metadynamics force is set to zero for s $<$ sw. Notice that Gaussians are added also if s $<$ sw, as the tails of these Gaussians influence VG in the relevant region s $>$ sw. In this way, the force on the system in the region s $>$ sw comes from both metadynamics and the force field, in the region s $<$ sw only from the latter. This approach allows obtaining a history-dependent bias potential VG that fluctuates around a stable estimator, equal to the negative of the free energy far enough from the boundaries. Note that:

- It works only for one-dimensional biases;
- It works both with and without GRID;
- The interval limit sw in a region where the free energy derivative is not large;
- If in the region outside the limit sw the system has a free energy minimum, the INTERVAL keyword should be used together with a UPPER_WALLS or LOWER_WALLS at sw.

As a final note, since version 2.0.2 when the system is outside of the selected interval the force is set to zero and the bias value to the value at the corresponding boundary. This allows acceptances for replica exchange methods to be computed correctly.

Multiple walkers [27] can also be used. See below the examples.

Additional material and examples can be also found in the tutorials:

- Belfast tutorial: Metadynamics
- Belfast tutorial: Replica exchange I
- Belfast tutorial: Replica exchange II and Multiple walkers

Notice that at variance with PLUMED 1.3 it is now straightforward to apply concurrent metadynamics as done e.g. in Ref. [28] . This indeed can be obtained by using the METAD action multiple times in the same input file.

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential

In addition the following quantities can be calculated by employing the keywords listed below

Quantity	Keyword	Description
acc	**ACCELERATION**	the metadynamics acceleration factor

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.
SIGMA	the widths of the Gaussian hills
PACE	the frequency for hill addition
FILE	(default=HILLS) a file in which the list of added hills is stored

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically
GRID_SPARSE	(default=off) use a sparse grid to store hills
GRID_NOSPLINE	(default=off) don't use spline interpolation with grids
STORE_GRIDS	(default=off) store all the grid files the calculation generates. They will be deleted if this keyword is not present
WALKERS_MPI	(default=off) Switch on MPI version of multiple walkers - not compatible with other WALKERS_$*$ options
ACCELERATION	(default=off) Set to TRUE if you want to compute the metadynamics acceleration factor.

HEIGHT	the heights of the Gaussian hills. Compulsory unless TAU, TEMP and BIASFACTOR are given

FMT	specify format for HILLS files (useful for decrease the number of digits in regtests)
BIASFACTOR	use well tempered metadynamics and use this biasfactor. Please note you must also specify temp
TEMP	the system temperature - this is only needed if you are doing well-tempered metadynamics
TAU	in well tempered metadynamics, sets height to (kb∗DeltaT∗pace∗timestep)/tau
GRID_MIN	the lower bounds for the grid
GRID_MAX	the upper bounds for the grid
GRID_BIN	the number of bins for the grid
GRID_SPACING	the approximate grid spacing (to be used as an alternative or together with GRID_BIN)
GRID_WSTRIDE	write the grid to a file every N steps
GRID_WFILE	the file on which to write the grid
ADAPTIVE	use a geometric (=GEOM) or diffusion (=DIFF) based hills width scheme. Sigma is one number that has distance units or timestep dimensions
WALKERS_ID	walker id
WALKERS_N	number of walkers
WALKERS_DIR	shared directory with the hills files from all the walkers
WALKERS_RSTRIDE	stride for reading hills files
INTERVAL	monodimensional lower and upper limits, outside the limits the system will not feel the biasing force.
GRID_RFILE	a grid file from which the bias should be read at the initial step of the simulation
SIGMA_MAX	the upper bounds for the sigmas (in CV units) when using adaptive hills. Negative number means no bounds
SIGMA_MIN	the lower bounds for the sigmas (in CV units) when using adaptive hills. Negative number means no bounds

Examples

The following input is for a standard metadynamics calculation using as collective variables the distance between atoms 3 and 5 and the distance between atoms 2 and 4. The value of the CVs and the metadynamics bias potential are written to the COLVAR file every 100 steps.

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
METAD ARG=d1,d2 SIGMA=0.2,0.2 HEIGHT=0.3 PACE=500 LABEL=restraint
PRINT ARG=d1,d2,restraint.bias STRIDE=100  FILE=COLVAR
```

(See also DISTANCE PRINT).

If you use adaptive Gaussians, with diffusion scheme where you use a Gaussian that should cover the space of 20 timesteps in collective variables. Note that in this case the histogram correction is needed when summing up hills.

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
METAD ARG=d1,d2 SIGMA=20 HEIGHT=0.3 PACE=500 LABEL=restraint ADAPTIVE=DIFF
PRINT ARG=d1,d2,restraint.bias STRIDE=100  FILE=COLVAR
```

If you use adaptive Gaussians, with geometrical scheme where you use a Gaussian that should cover the space of 0.05 nm in Cartesian space. Note that in this case the histogram correction is needed when summing up hills.

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
METAD ARG=d1,d2 SIGMA=0.05 HEIGHT=0.3 PACE=500 LABEL=restraint ADAPTIVE=GEOM
PRINT ARG=d1,d2,restraint.bias STRIDE=100  FILE=COLVAR
```

When using adaptive Gaussians you might want to limit how the hills width can change. You can use SIGMA_MIN and SIGMA_MAX keywords. The sigmas should specified in terms of CV so you should use the CV units. Note that if you use a negative number, this means that the limit is not set. Note also that in this case the histogram correction is needed when summing up hills.

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
METAD ...
  ARG=d1,d2 SIGMA=0.05 HEIGHT=0.3 PACE=500 LABEL=restraint ADAPTIVE=GEOM
  SIGMA_MIN=0.2,0.1 SIGMA_MAX=0.5,1.0
... METAD
PRINT ARG=d1,d2,restraint.bias STRIDE=100  FILE=COLVAR
```

Multiple walkers can be also use as in [27] These are enabled by setting the number of walker used, the id of the current walker which interprets the input file, the directory where the hills containing files resides, and the frequency to read the other walkers. Here is an example

```
DISTANCE ATOMS=3,5 LABEL=d1
METAD ...
   ARG=d1 SIGMA=0.05 HEIGHT=0.3 PACE=500 LABEL=restraint
   WALKERS_N=10
   WALKERS_ID=3
   WALKERS_DIR=../
   WALKERS_RSTRIDE=100
... METAD
```

where WALKERS_N is the total number of walkers, WALKERS_ID is the id of the present walker (starting from 0) and the WALKERS_DIR is the directory where all the walkers are located. WALKERS_RSTRIDE is the number of step between one update and the other.

The kinetics of the transitions between basins can also be analysed on the fly as in [29]. The flag ACCELERATION turn on accumulation of the acceleration factor that can then be used to determine the rate. This method can be used together with COMMITTOR analysis to stop the simulation when the system get to the target basin. It must be used together with Well-Tempered Metadynamics.

7.6 MOVINGRESTRAINT

	This is part of the bias module

Add a time-dependent, harmonic restraint on one or more variables.

This form of bias can be used to performed steered MD [30] and Jarzynski sampling [31].

The harmonic restraint on your system is given by:

$$V(\vec{s},t) = \frac{1}{2}\kappa(t)(\vec{s}-\vec{s}_0(t))^2$$

The time dependence of κ and $\vec{s}_0$ are specified by a list of STEP, KAPPA and AT keywords. These keywords tell plumed what values κ and $\vec{s}_0$ should have at the time specified by the corresponding STEP keyword. Inbetween these times the values of κ and $\vec{s}_0$ are linearly interpolated.

Additional material and examples can be also found in the tutorial Belfast tutorial: Out of equilibrium dynamics

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential
_cntr	one or multiple instances of this quantity will be refereceable elsewhere in the input file. these quantities will named with the arguments of the bias followed by the character string _cntr. These quantities give the instantaneous position of the center of the harmonic potential.
_work	one or multiple instances of this quantity will be refereceable elsewhere in the input file. These quantities will named with the arguments of the bias followed by the character string _work. These quantities tell the user how much work has been done by the potential in dragging the system along the various colvar axis.
_kappa	one or multiple instances of this quantity will be refereceable elsewhere in the input file. These quantities will named with the arguments of the bias followed by the character string _kappa. These quantities tell the user the time dependent value of kappa.

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

VERSE	(default=B) Tells plumed whether the restraint is only acting for CV larger (U) or smaller (L) than the restraint or whether it is acting on both sides (B)
STEP	This keyword appears multiple times as STEPx with x=0,1,2,...,n. Each value given represents the MD step at which the restraint parameters take the values KAPPAx and ATx. You can use multiple instances of this keyword i.e. STEP1, STEP2, STEP3...
AT	ATx is equal to the position of the restraint at time STEPx. For intermediate times this parameter is linearly interpolated. If no ATx is specified for STEPx then the values of AT are kept constant during the interval of time between STEPx-1 and STEPx. You can use multiple instances of this keyword i.e. AT1, AT2, AT3...
KAPPA	KAPPAx is equal to the value of the force constants at time STEPx. For intermediate times this parameter is linearly interpolated. If no KAPPAx is specified for STEPx then the values of KAPPAx are kept constant during the interval of time between STEPx-1 and STEPx. You can use multiple instances of this keyword i.e. KAPPA1, KAPPA2, KAPPA3...

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input is dragging the distance between atoms 2 and 4 from 1 to 2 in the first 1000 steps, then back in the next 1000 steps. In the following 500 steps the restraint is progressively switched off.

```
DISTANCE ATOMS=2,4 LABEL=d
MOVINGRESTRAINT ...
  ARG=d
  STEP0=0    AT0=1.0 KAPPA0=100.0
  STEP1=1000 AT1=2.0
  STEP2=2000 AT2=1.0
  STEP3=2500         KAPPA3=0.0
... MOVINGRESTRAINT
```

The following input is progressively building restraints distances between atoms 1 and 5 and between atoms 2 and 4 in the first 1000 steps. Afterwards, the restraint is kept static.

```
DISTANCE ATOMS=1,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
MOVINGRESTRAINT ...
  ARG=d1,d2
  STEP0=0    AT0=1.0,1.5 KAPPA0=0.0,0.0
  STEP1=1000 AT1=1.0,1.5 KAPPA1=1.0,1.0
... MOVINGRESTRAINT
```

The following input is progressively bringing atoms 1 and 2 close to each other with an upper wall

```
DISTANCE ATOMS=1,2 LABEL=d1
MOVINGRESTRAINT ...
  ARG=d1
  VERSE=U
  STEP0=0    AT0=1.0 KAPPA0=10.0
  STEP1=1000 AT1=0.0
... MOVINGRESTRAINT
```

By default the Action is issuing some values which are the work on each degree of freedom, the center of the harmonic potential, the total bias deposited

(See also DISTANCE).

Attention

Work is not computed properly when KAPPA is time dependent.

7.7 RESTRAINT

	This is part of the bias module

Adds harmonic and/or linear restraints on one or more variables.

Either or both of SLOPE and KAPPA must be present to specify the linear and harmonic force constants respectively. The resulting potential is given by:

$$\sum_i \frac{k_i}{2}(x_i - a_i)^2 + m_i * (x_i - a_i)$$

.

The number of components for any vector of force constants must be equal to the number of arguments to the action.

Additional material and examples can be also found in the tutorial Belfast tutorial: Umbrella sampling

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.
SLOPE	(default=0.0) specifies that the restraint is linear and what the values of the force constants on each of the variables are
KAPPA	(default=0.0) specifies that the restraint is harmonic and what the values of the force constants on each of the variables are
AT	the position of the restraint

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input tells plumed to restrain the distance between atoms 3 and 5 and the distance between atoms 2 and 4, at different equilibrium values, and to print the energy of the restraint

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
RESTRAINT ARG=d1,d2 AT=1.0,1.5 KAPPA=150.0,150.0 LABEL=restraint
PRINT ARG=restraint.bias
```

(See also DISTANCE and PRINT).

7.8 UPPER_WALLS

	This is part of the bias module

Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.

The restraining potential starts acting on the system when the value of the CV is greater (in the case of UPPER_WALLS) or lower (in the case of LOWER_WALLS) than a certain limit a_i (AT) minus an offset o_i (OFFSET). The expression for the bias due to the wall is given by:

$\sum_i k_i((x_i - a_i + o_i)/s_i)_i^e$

k_i (KAPPA) is an energy constant in internal unit of the code, s_i (EPS) a rescaling factor and e_i (EXP) the exponent determining the power law. By default: EXP = 2, EPS = 1.0, OFFSET = 0.

Description of components

By default this Action calculates the following quantities. These quanties can be referenced elsewhere in the input by using this Action's label followed by a dot and the name of the quantity required from the list below.

Quantity	Description
bias	the instantaneous value of the bias potential
force2	the instantaneous value of the squared force due to this bias potential

Compulsory keywords

ARG	the input for this action is the scalar output from one or more other actions. The particular scalars that you will use are referenced using the label of the action. If the label appears on its own then it is assumed that the Action calculates a single scalar value. The value of this scalar is thus used as the input to this new action. If $*$ or $*.*$ appears the scalars calculated by all the proceding actions in the input file are taken. Some actions have multi-component outputs and each component of the output has a specific label. For example a DISTANCE action labelled dist may have three componets x, y and z. To take just the x component you should use dist.x, if you wish to take all three components then use dist.$*$.More information on the referencing of Actions can be found in the section of the manual on the PLUMED Getting started. Scalar values can also be referenced using POSIX regular expressions as detailed in the section on Regular Expressions. To use this feature you you must compile PLUMED with the appropriate flag.

AT	the positions of the wall. The a_i in the expression for a wall.
KAPPA	the force constant for the wall. The k_i in the expression for a wall.
OFFSET	(default=0.0) the offset for the start of the wall. The o_i in the expression for a wall.
EXP	(default=2.0) the powers for the walls. The e_i in the expression for a wall.
EPS	(default=1.0) the values for s_i in the expression for a wall

Options

NUMERICAL_DERIVATIVES	(default=off) calculate the derivatives for these quantities numerically

Examples

The following input tells plumed to add both a lower and an upper walls on the distance between atoms 3 and 5 and the distance between atoms 2 and 4. The lower and upper limits are defined at different values. The strength of the walls is the same for the four cases. It also tells plumed to print the energy of the walls.

```
DISTANCE ATOMS=3,5 LABEL=d1
DISTANCE ATOMS=2,4 LABEL=d2
UPPER_WALLS ARG=d1,d2 AT=1.0,1.5 KAPPA=150.0,150.0 EXP=2,2 EPS=1,1 OFFSET=0,0 LABEL=uwall
LOWER_WALLS ARG=d1,d2 AT=0.0,1.0 KAPPA=150.0,150.0 EXP=2,2 EPS=1,1 OFFSET=0,0 LABEL=lwall
PRINT ARG=uwall.bias,lwall.bias
```

(See also DISTANCE and PRINT).

7.9 RESTART

	This is part of the setup module

Activate restart.

This is a Setup directive and, as such, should appear at the beginning of the input file.

Examples

Using the following input:

```
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=out
```

a new 'out' file will be created. If an old one is on the way, it will be automatically backed up. On the other hand, using the following input:

```
RESTART
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=out
```

the file 'out' will be appended. (See also DISTANCE and PRINT).

Attention

This directive can have also other side effects, e.g. on METAD

7.10 IMD

	This is part of the imd module

Use interactive molecular dynamics with VMD

Examples

```
# listen to port 1112 of localhost
IMD PORT=1112

# listen to port 1112 of pippo
IMD HOST=pippo PORT=1112

# listen to port 1112 of localhost and run only when connected
IMD PORT=1112 WAIT
```

Attention

The IMB object only works if the IMD routines have been downloaded and properly linked with PLUMED

Chapter 8

Command Line Tools

PLUMED contains a number of simple command line tools. To use one of these tools you issue a command something like:

```
plumed <toolname> <list of input flags for that tool>
```

The following is a list of the various standalone tools that PLUMED contains.

driver	driver is a tool that allows one to to use plumed to post-process an existing trajectory.
gentemplate	gentemplate is a tool that you can use to construct template inputs for the variousactions
info	This tool allows you to obtain information about your plumed version
kt	Print out the value of k_BT at a particular temperature
manual	manual is a tool that you can use to construct the manual page for a particular action
simplemd	simplemd allows one to do molecular dynamics on systems of Lennard-Jones atoms.
sum_hills	sum_hills is a tool that allows one to to use plumed to post-process an existing hills/colvar file

For all these tools and to use PLUMED as a plugin in an MD calculation you will need an input file.

8.1 driver

	This is part of the cltools module

driver is a tool that allows one to to use plumed to post-process an existing trajectory.

The input to driver is specified using the command line arguments described below.

In addition, you can use the special READ command inside your plumed input to read in colvar files that were generated during your MD simulation. The values read in can then be treated like calculated colvars.

The input trajectory is specified using one of the following

--ixyz	the trajectory in xyz format
--igro	the trajectory in gro format

--mf_dcd	molfile: the trajectory in dcd format
--mf_crd	molfile: the trajectory in crd format
--mf_crdbox	molfile: the trajectory in crdbox format
--mf_gro	molfile: the trajectory in gro format
--mf_g96	molfile: the trajectory in g96 format
--mf_trr	molfile: the trajectory in trr format
--mf_trj	molfile: the trajectory in trj format
--mf_xtc	molfile: the trajectory in xtc format
--mf_pdb	molfile: the trajectory in pdb format

The following must be present

--plumed	(default=plumed.dat) specify the name of the plumed input file
--timestep	(default=1.0) the timestep that was used in the calculation that produced this trajectory in picoseconds
--trajectory-stride	(default=1) the frequency with which frames were output to this trajectory during the simulation
--multi	(default=0) set number of replicas for multi environment (needs mpi)

The following options are available

--help/-h	(default=off) print this help
--help-debug	(default=off) print special options that can be used to create regtests
--noatoms	(default=off) don't read in a trajectory. Just use colvar files as specified in plumed.dat
--dump-full-virial	(default=off) with –dump-forces, it dumps the 9 components of the virial

--length-units	units for length, either as a string or a number
--dump-forces	dump the forces on a file
--dump-forces-fmt	(default=%f) the format to use to dump the forces
--pdb	provides a pdb with masses and charges
--box	comma-separated box dimensions (3 for orthorombic, 9 for generic)
--natoms	provides number of atoms - only used if file format does not contain number of atoms

Examples

The following command tells plumed to postprocess the trajectory contained in trajectory.xyz by performing the actions described in the input file plumed.dat. If an action that takes the stride keyword is given a stride equal to n then it will be performed only on every nth frame in the trajectory file.

```
plumed driver --plumed plumed.dat --ixyz trajectory.xyz
```

The following command tells plumed to postprocess the trajectory contained in trajectory.xyz. by performing the actions described in the input file plumed.dat. Here though –trajectory-stride is set equal to the frequency with which frames were output during the trajectory and the –timestep is equal to the simulation timestep. As such the STRIDE parameters in the plumed.dat files are no longer ignored and any files output resemble those that would have been generated had we run the calculation we are running with driver when the MD simulation was running.

```
plumed driver --plumed plumed.dat --ixyz trajectory.xyz --trajectory-stride 100 --timestep 0.001
```

By default you have access to a subset of the trajectory file formats supported by VMD, e.g. xtc and dcd:

```
plumed driver --plumed plumed.dat --pdb diala.pdb --mf_xtc traj.xtc --trajectory-stride 100 --timestep 0.001
```

where –mf_ prefixes the extension of one of the accepted molfile plugin format.

To have support of all of VMD's plugins you need to recompile PLUMED. You need to download the SOURCE of VMD, which contains a plugins directory. Adapt build.sh and compile it. At the end, you should get the molfile plugins compiled as a static library libmolfile_plugin.a. Locate said file and libmolfile_plugin.h, and customize the configure command with something along the lines of:

```
configure [...] LDFLAGS="-ltcl8.5 -L/mypathtomolfilelibrary/ -L/mypathtotcl" CPPFLAGS="-I/mypathtolibmolfile_p
```

and rebuild.

Molfile plugin require periodic cell to be triangular (i.e. first vector oriented along x and second vector in xy plane). This is true for many MD codes. However, it could be false if you rotate the coordinates in your trajectory before reading them in the driver. Also notice that some formats (e.g. amber crd) do not specify atom number. In this case you can use the `--natoms` option:

```
plumed driver --plumed plumed.dat --imf_crd trajectory.crd --natoms 128
```

Check the available molfile plugins and limitations at http://www.ks.uiuc.edu/Research/vmd/plugins/molfile/.

8.1.1 READ

	This is part of the generic module

Read quantities from a colvar file.

This Action can be used with driver to read in a colvar file that was generated during an MD simulation

Description of components

The READ command will read those fields that are labelled with the text string given to the VALUE keyword. It will also read in any fields that are labelleled with the text string given to the VALUE keyword followed by a dot and a further string. If a single Value is read in this value can be referenced using the label of the Action. Alternatively, if multiple quanties are read in, they can be referenced elsewhere in the input by using the label for the Action followed by a dot and the character string that appeared after the dot in the title of the field.

Compulsory keywords

STRIDE	(default=1) the frequency with which the file should be read.
EVERY	(default=1) only read every ith line of the colvar file. This should be used if the colvar was written more frequently than the trajectory.

VALUES	the values to read from the file
FILE	the name of the file from which to read these quantities

Options

IGNORE_TIME	(default=off) ignore the time in the colvar file. When this flag is not present read will be quite strict about the start time of the simulation and the stride between frames

Examples

This input reads in data from a file called input_colvar.data that was generated in a calculation that involved PLU↩ MED. The first command reads in the data from the column headed phi1 while the second reads in the data from the column headed phi2.

```
rphi1:       READ FILE=input_colvar.data  VALUES=phi1
rphi2:       READ FILE=input_colvar.data  VALUES=phi2
PRINT ARG=rphi1,rphi2 STRIDE=500  FILE=output_colvar.data
```

8.2 gentemplate

	This is part of the cltools module

gentemplate is a tool that you can use to construct template inputs for the various actions

The templates generated by this tool are primarily for use with Toni Giorgino's vmd gui. It may be useful however to use this tool as a quick aid memoir.

Options

--help/-h	(default=off) print this help
--list	(default=off) print a list of the available actions
--include-optional	(default=off) also print optional modifiers

--action	print the template for this particular action

Examples

The following generates template input for the action DISTANCE.

```
plumed gentemplate --action DISTANCE
```

8.3 info

	This is part of the cltools module

This tool allows you to obtain information about your plumed version

You can specify the information you require using the following command line arguments

Options

--help/-h	(default=off) print this help
--configuration	(default=off) prints the configuration file
--root	(default=off) print the location of the root directory for the plumed source
--user-doc	(default=off) print the location of user manual (html)
--developer-doc	(default=off) print the location of user manual (html)
--version	(default=off) print the version number
--long-version	(default=off) print the version number (long version)
--git-version	(default=off) print the version number (git version, if available)

Examples

The following command returns the root directory for your plumed distribution.

```
plumed info --root
```

8.4 kt

	This is part of the cltools module

Print out the value of k_BT at a particular temperature

Compulsory keywords

--temp	print the manual for this particular action
--units	(default=kj/mol) the units of energy can be kj/mol, kcal/mol, j/mol, eV or the conversion factor from kj/mol

Options

--help/-h	(default=off) print this help

Examples

The following command will tell you the value of k_BT when T is equal to 300 K in eV

```
plumed kt --temp 300 --units eV
```

8.5 manual

	This is part of the cltools module

manual is a tool that you can use to construct the manual page for a particular action

The manual constructed by this action is in html. In all probability you will never need to use this tool. However, it is used within the scripts that generate plumed's html manual. If you need to use this tool outside those scripts the input is specified using the following command line arguments.

Compulsory keywords

--action	print the manual for this particular action

Options

--help/-h	(default=off) print this help

Examples

The following generates the html manual for the action DISTANCE.

```
plumed manual --action DISTANCE
```

8.6 simplemd

	This is part of the cltools module

simplemd allows one to do molecular dynamics on systems of Lennard-Jones atoms.

The input to simplemd is spcified in an input file. Configurations are input and output in xyz format. The input file should contain one directive per line. The directives available are as follows:

Compulsory keywords

nstep	The number of steps of dynamics you want to run
temperature	(default=NVE) the temperature at which you wish to run the simulation in LJ units
friction	(default=off) The friction (in LJ units) for the langevin thermostat that is used to keep the temperature constant
tstep	(default=0.005) the integration timestep in LJ units
inputfile	An xyz file containing the initial configuration of the system
forcecutoff	(default=2.5)
listcutoff	(default=3.0)
outputfile	An output xyz file containing the final configuration of the system

nconfig	(default=10) The frequency with which to write configurations to the trajectory file followed by the name of the trajectory file
nstat	(default=1) The frequency with which to write the statistics to the statistics file followed by the name of the statistics file
maxneighbours	(default=10000) The maximum number of neighbours an atom can have
idum	(default=0) The random number seed
ndim	(default=3) The dimensionality of the system (some interesting LJ clusters are two dimensional)
wrapatoms	(default=false) If true, atomic coordinates are written wrapped in minimal cell

Examples

You run an MD simulation using simplemd with the following command:

```
plumed simplemd < in
```

The following is an example of an input file for a simplemd calculation. This file instructs simplemd to do 50 steps of MD at a temperature of 0.722

```
nputfile input.xyz
outputfile output.xyz
temperature 0.722
tstep 0.005
friction 1
forcecutoff 2.5
listcutoff  3.0
nstep 50
nconfig 10 trajectory.xyz
nstat   10 energies.dat
```

If you run the following a description of all the directives that can be used in the input file will be output.

```
plumed simplemd --help
```

8.7 sum_hills

	This is part of the cltools module

sum_hills is a tool that allows one to to use plumed to post-process an existing hills/colvar file

Options

--help/-h	(default=off) print this help
--help-debug	(default=off) print special options that can be used to create regtests
--negbias	(default=off) print the negative bias instead of the free energy (only needed with welltempered runs and flexible hills)

--nohistory	(default=off) to be used with –stride: it splits the bias/histogram in pieces without previous history
--mintozero	(default=off) it translate all the minimum value in bias/histogram to zero (usefull to compare results)

--hills	specify the name of the hills file
--histo	specify the name of the file for histogram a colvar/hills file is good
--stride	specify the stride for integrating hills file (default 0=never)
--min	the lower bounds for the grid
--max	the upper bounds for the grid
--bin	the number of bins for the grid
--spacing	grid spacing, alternative to the number of bins
--idw	specify the variables to be used for the free-energy/histogram (default is all). With –hills the other variables will be integrated out, with –histo the other variables won't be considered
--outfile	specify the outputfile for sumhills
--outhisto	specify the outputfile for the histogram
--kt	specify temperature in energy units for integrating out variables
--sigma	a vector that specify the sigma for binning (only needed when doing histogram
--fmt	specify the output format

Examples

a typical case is about the integration of a hills file:

```
plumed sum_hills  --hills PATHTOMYHILLSFILE
```

The default name for the output file will be fes.dat Note that starting from this version plumed will automatically detect the number of the variables you have and their periodicity. Additionally, if you use flexible hills (multivariate gaussians), plumed will understand it from the HILLS file.

now sum_hills tool accepts als multiple files that will be integrated one after the other

```
plumed sum_hills  --hills PATHTOMYHILLSFILE1,PATHTOMYHILLSFILE2,PATHTOMYHILLSFILE3
```

if you want to integrate out some variable you do

```
plumed sum_hills  --hills PATHTOMYHILLSFILE   --idw t1 --kt 0.6
```

where with –idw you define the variables that you want all the others will be integrated out. –kt defines the temperature of the system in energy units. (be consistent with the units you have in your hills: plumed will not check this for you) If you need more variables then you may use a comma separated syntax

```
plumed sum_hills  --hills PATHTOMYHILLSFILE   --idw t1,t2 --kt 0.6
```

You can define the output grid only with the number of bins you want while min/max will be detected for you

```
plumed sum_hills --bin 99,99 --hills PATHTOMYHILLSFILE
```

or full grid specification

```
plumed sum_hills --bin 99,99 --min -pi,-pi --max pi,pi --hills PATHTOMYHILLSFILE
```

You can of course use numbers instead of -pi/pi.

You can use a –stride keyword to have a dump each bunch of hills you read

```
plumed sum_hills --stride 300 --hills PATHTOMYHILLSFILE
```

You can also have, in case of welltempered metadynamics, only the negative bias instead of the free energy through the keyword –negbias

```
plumed sum_hills --negbias --hills PATHTOMYHILLSFILE
```

Here the default name will be negativebias.dat

From time to time you might need to use HILLS or a COLVAR file as it was just a simple set of points from which you want to build a free energy by using -(1/beta)log(P) then you use –histo

```
plumed sum_hills --histo PATHTOMYCOLVARORHILLSFILE  --sigma 0.2,0.2 --kt 0.6
```

in this case you need a –kt to do the reweighting and then you need also some width (with the –sigma keyword) for the histogram calculation (actually will be done with gaussians, so it will be a continuous histogram) Here the default output will be histo.dat. Note that also here you can have multiple input files separated by a comma.

Additionally, if you want to do histogram and hills from the same file you can do as this

```
plumed sum_hills --hills --histo PATHTOMYCOLVARORHILLSFILE  --sigma 0.2,0.2 --kt 0.6
```

The two files can be eventually the same

Another interesting thing one can do is monitor the difference in blocks as a metadynamics goes on. When the bias deposited is constant over the whole domain one can consider to be at convergence. This can be done with the –nohistory keyword

```
plumed sum_hills --stride 300 --hills PATHTOMYHILLSFILE  --nohistory
```

and similarly one can do the same for an histogram file

```
plumed sum_hills --histo PATHTOMYCOLVARORHILLSFILE  --sigma 0.2,0.2 --kt 0.6 --nohistory
```

just to check the hypothetical free energy calculated in single blocks of time during a simulation and not in a cumulative way

Output format can be controlled via the –fmt field

```
plumed sum_hills --hills PATHTOMYHILLSFILE  --fmt %8.3f
```

where here we chose a float with length of 8 and 3 digits

The output can be named in a arbitrary way :

```
plumed sum_hills --hills PATHTOMYHILLSFILE  --outfile myfes.dat
```

will produce a file myfes.dat which contains the free energy.

If you use stride, this keyword is the suffix

```
plumed sum_hills --hills PATHTOMYHILLSFILE  --outfile myfes_ --stride 100
```

will produce myfes_0.dat, myfes_1.dat, myfes_2.dat etc.

The same is true for the output coming from histogram

```
plumed sum_hills --histo HILLS --kt 2.5 --sigma 0.01 --outhisto myhisto.dat
```

is producing a file myhisto.dat while, when using stride, this is the suffix

```
plumed sum_hills --histo HILLS --kt 2.5 --sigma 0.01 --outhisto myhisto_ --stride 100
```

that gives myhisto_0.dat, myhisto_1.dat, myhisto_3.dat etc..

Chapter 9

Miscelaneous

9.1 Comments

If you are an organised sort of person who likes to remember what the hell you were trying to do when you ran a particular simulation you might find it useful to put comments in your input file. In PLUMED you can do this as comments can be added using a # sign. On any given line everything after the # sign is ignored so erm... yes add lines of comments or trailing comments to your hearts content as shown below (using Shakespeare is optional):

```
# This is the distance between two atoms:
DISTANCE ATOM=1,2 LABEL=d1
UPPER_WALLS ARG=d1 AT=3.0 KAPPA=3.0 LABEL=Snout # In this same interlude it doth befall.
# That I, one Snout by name, present a wall.
```

(see DISTANCE and UPPER_WALLS)

An alternative to including comments in this way is to use line starting ENDPLUMED. Everything in the PLUMED input after this keyword will be ignored.

9.2 Continuation lines

If your input lines get very long then editing them using vi and other such text editors becomes a massive pain in the arse. We at PLUMED are aware of this fact and thus have provided a way of doing line continuations so as to make your life that much easier - aren't we kind? Well no not really, we have to use this code too. Anyway, you can do continuations by using the "..." syntax as this makes this:

```
DISTANCES ATOMS1=1,300 ATOMS2=1,400 ATOMS3=1,500
```

(see DISTANCES)

equivalent to this:

```
DISTANCES ...
# we can also insert comments here
  ATOMS1=1,300
# multiple kewords per line are allowed
  ATOMS2=1,400 ATOMS3=1,500
#empty lines are also allowed

... DISTANCES
```

9.3 Including other files

If, for some reason, you want to spread your PLUMED input over a number of files you can use INCLUDE as shown below:

```
INCLUDE FILE=filename
```

So, for example, a single "plumed.dat" file:

```
DISTANCE ATOMS=0,1 LABEL=dist
RESTRAINT ARG=dist
```

(see DISTANCE and RESTRAINT)

could be split up into two files as shown below:

```
DISTANCE ATOMS=0,1 LABEL=dist
INCLUDE FILE=toBeIncluded.dat
```

plus a "toBeIncluded.dat" file

```
RESTRAINT ARG=dist
```

However, when you do this it is important to recognise that INCLUDE is a real directive that is only resolved after all the Comments have been stripped and the Continuation lines have been unrolled. This means it is not possible to do things like:

```
# this is wrong:
DISTANCE INCLUDE FILE=options.dat
RESTRAINT ARG=dist
```

9.3.1 INCLUDE

	This is part of the generic module

Includes an external input file, similar to "#include" in C preprocessor.

Useful to split very large plumed.dat files.

Compulsory keywords

FILE	file to be included

Examples

This input

```
c1: COM ATOMS=1-100
c2: COM ATOMS=101-202
d: DISTANCE ARG=c1,c2
PRINT ARG=d
```

can be replaced with

```
INCLUDE FILE=pippo.dat
d: DISTANCE ARG=c1,c2
PRINT ARG=d
```

where the content of file pippo.dat is

```
c1: COM ATOMS=1-100
c2: COM ATOMS=101-202
```

(see also COM, DISTANCE, and PRINT).

9.4 Loading shared libraries

You can introduce new functionality into PLUMED by placing it directly into the src directory and recompiling the PLUMED libraries. Alternatively, if you want to keep your code independent from the rest of PLUMED (perhaps so you can release it independely - we won't be offended), then you can create your own dynamic library. To use this in conjuction with PLUMED you can then load it at runtime by using the LOAD keyword as shown below:

```
LOAD FILE=library.so
```

N.B. If your system uses a different suffix for dynamic libraries (e.g. macs use .dylib) then PLUMED will try to automatically adjust the suffix accordingly.

9.4.1 LOAD

	This is part of the setup module

Loads a library, possibly defining new actions.

It is available only on systems allowing for dynamic loading. It can also be fed with a cpp file, in which case the file is compiled first.

Compulsory keywords

FILE	file to be loaded

Examples

If you have a shared object named extensions.so and want to use the functionalities implemented in it within PLU↩ MED you can load it with the following syntax

```
LOAD FILE=extensions.so
```

As a more practical example, imagine that you want to make a small change to one collective variable that is already implemented in PLUMED, say DISTANCE . Copy the file `src/colvar/Distance.cpp` into your work directory, rename it as `Distance2.cpp` and edit it as you wish. It might be better to also replace any occurence of the string DISTANCE within the file with DISTANCE2, so that both old and new implementation will be available with different names. Then you can compile it into a shared object using

```
> plumed mklib Distance2.cpp
```

This will generate a file `Distance2.so` (or `Distance2.dylib` on a mac) that can be loaded. Now you can use your new implementation with the following input

```
# load the new library
LOAD FILE=Distance2.so
# compute standard distance
d: DISTANCE ATOMS=1,10
# compute modified distance
d2: DISTANCE2 ATOMS=1,10
# print them on a file
PRINT ARG=d,d2 FILE=compare-them
```

You can even skip the initial step and directly feed PLUMED with the `Distance2.cpp` file: it will be compiled on the fly.

```
# load the new definition
# this is a cpp file so it will be compiled
LOAD FILE=Distance2.cpp
# compute standard distance
d: DISTANCE ATOMS=1,10
# compute modified distance
d2: DISTANCE2 ATOMS=1,10
# print them on a file
PRINT ARG=d,d2 FILE=compare-them
```

This will allow to make quick tests while developing your own variables. Of course, after your implementation is ready you might want to add it to the PLUMED source tree and recompile the whole PLUMED.

9.5 Debugging the code

The DEBUG action provides some functionality for debugging the code that may be useful if you are doing very intensive development of the code of if you are running on a computer with a strange architecture.

9.5.1 DEBUG

	This is part of the generic module

Set some debug options.

Can be used while debugging or optimizing plumed.

Compulsory keywords

STRIDE	(default=1) the frequency with which this action is to be performed

Options

logActivity	(default=off) write in the log which actions are inactive and which are inactive
logRequestedAtoms	(default=off) write in the log which atoms have been requested at a given time
NOVIRIAL	(default=off) switch off the virial contribution for the entirity of the simulation
DETAILED_TIMERS	(default=off) switch on detailed timers

Examples

```
# print detailed (action-by-action) timers at the end of simulation
DEBUG DETAILED_TIMERS
# dump every two steps which are the atoms required from the MD code
DEBUG logRequestedAtoms STRIDE=2
```

9.6 Changing exchange patterns in replica exchange

Using the RANDOM_EXCHANGES keyword it is possible to make exchanges betweem randomly chosen replicas. This is useful e.g. for bias exchange metadynamics [32].

9.6.1 RANDOM_EXCHANGES

	This is part of the generic module

Set random pattern for exchanges.

In this way, exchanges will not be done between replicas with consecutive index, but will be done using a random pattern. Typically used in bias exchange [32].

SEED	seed for random exchanges

Examples

Using the following three input files one can run a bias exchange metadynamics simulation using a different angle in each replica. Exchanges will be randomly tried between replicas 0-1, 0-2 and 1-2

Here is plumed.dat.0

```
RANDOM_EXCHANGES
t: TORSION ATOMS=1,2,3,4
METAD ARG=t HEIGHT=0.1 PACE=100 SIGMA=0.3
```

Here is plumed.dat.1

```
RANDOM_EXCHANGES
t: TORSION ATOMS=2,3,4,5
METAD ARG=t HEIGHT=0.1 PACE=100 SIGMA=0.3
```

Here is plumed.dat.2

```
RANDOM_EXCHANGES
t: TORSION ATOMS=3,4,5,6
METAD ARG=t HEIGHT=0.1 PACE=100 SIGMA=0.3
```

Warning

Multi replica simulations are presently only working with gromacs.
The directive should appear in input files for every replicas. In case SEED is specified, it should be the same in all input files.

9.7 List of modules

The functionality in PLUMED 2 is divided into a small number of modules. Some users may only wish to use a subset of the functionality available within the code while others may wish to use some of PLUMED's more complicated features. For this reason the plumed source code is divided into modules, which users can activate or deactivate to their hearts content.

- To activate a module add a file called modulename.on to the plumed2/src directory
- To deactivate a module add a file called modulename.off to the plumed2/src directory

Obviously, in the above instructions modulename should be replaced by the name of the module that you would like to activate or deactivate.

Some modules are active by default in the version of PLUMED 2 that you download from the website while others are inactive. The following lists all of the modules that are available in plumed and tells you whether or not they are active by default.

Module name	Default behavior
analysis	on
bias	on
cltools	on
colvar	on
crystallization	off
function	on
generic	on
imd	off
manyrestraints	off
mapping	on
molfile	on
multicolvar	on
reference	on
secondarystructure	on
setup	on
vatom	on
vesselbase	on

9.8 Frequently used tools

histograbead	INTERNAL	A function that can be used to calculate whether quantities are between fixed upper and lower bounds.
kernelfunctions	INTERNAL	Functions that are used to construct histograms
landmarkselection	INTERNAL	This is currently a filler page.
reweighting	INTERNAL	Calculate free energies from a biassed/higher temperature trajectory.
switchingfunction	INTERNAL	Functions that measure whether values are less than a certain quantity.

Regular Expressions		POSIX regular expressions can be used to select multiple actions when using ARG (i.e. PRINT).
Files		Dealing with Input/Outpt

9.8.1 histogrambead

A function that can be used to calculate whether quantities are between fixed upper and lower bounds. A function that can be used to calculate whether quantities are between fixed upper and lower bounds.

If we have multiple instances of a variable we can estimate the probability distribution (pdf) for that variable using a process called kernel density estimation:

$$P(s) = \sum_i K\left(\frac{s - s_i}{w}\right)$$

In this equation K is a symmetric funciton that must integrate to one that is often called a kernel function and w is a smearing parameter. From a pdf calculated using kernel density estimation we can calculate the number/fraction of values between an upper and lower bound using:

$$w(s) = \int_a^b \sum_i K\left(\frac{s - s_i}{w}\right)$$

All the input to calculate a quantity like $w(s)$ is generally provided through a single keyword that will have the following form:

KEYWORD={TYPE UPPER= a LOWER= b SMEAR= $\frac{w}{b-a}$}

This will calculate the number of values between a and b. To calculate the fraction of values you add the word NORM to the input specification. If the function keyword SMEAR is not present w is set equal to $0.5(b-a)$. Finally, type should specify one of the kernel types that is present in plumed. These are listed in the table below:

TYPE	FUNCTION
GAUSSIAN	$\frac{1}{\sqrt{2\pi}w}\exp\left(-\frac{(s-s_i)^2}{2w^2}\right)$
TRIANGULAR	$\frac{1}{2w}\left(1.-\left\lvert\frac{s-s_i}{w}\right\rvert\right)$ $\quad \frac{s-s_i}{w} < 1$

Some keywords can also be used to calculate a descretized version of the histogram. That is to say the number of values between a and b, the number of values between b and c and so on. A keyword that specifies this sort of calculation would look something like

KEYWORD={TYPE UPPER= a LOWER= b NBINS= n SMEAR= $\frac{w}{n(b-a)}$}

This specification would calculate the following vector of quantities:

$$w_j(s) = \int_{a+\frac{j-1}{n}(b-a)}^{a+\frac{j}{n}(b-a)} \sum_i K\left(\frac{s - s_i}{w}\right)$$

9.8.2 kernelfunctions

Functions that are used to construct histograms Functions that are used to construct histograms

Constructing histograms is something you learnt to do relatively early in life. You perform an experiment a number of times, count the number of times each result comes up and then draw a bar graph that describes how often each of the results came up. This only works when there are a finite number of possible results. If the result a number between 0 and 1 the bar chart is less easy to draw as there are as many possible results as there are numbers between zero and one - an infinite number. To resolve this problem we replace probability, P with probability density, π, and write the probability of getting a number between a and b as:

$$P = \int_a^b \mathrm{d}x \pi(x)$$

To calculate probability densities from a set of results we use a process called kernel density estimation. Histograms are accumulated by adding up kernel functions, K, with finite spatial extent, that integrate to one. These functions are centered on each of the n-dimensional data points, $\mathbf{x}_i$. The overall effect of this is that each result we obtain in our experiments contributes to the probability density in a finite sized region of the space.

Expressing all this mathematically in kernel density estimation we write the probability density as:

$$\pi(\mathbf{x}) = \sum_i K\left[(\mathbf{x}-\mathbf{x}_i)^T \Sigma (\mathbf{x}-\mathbf{x}_i)\right]$$

where Σ is an $n \times n$ matrix called the bandwidth that controls the spatial extent of the kernel. Whenever we accumulate a histogram (e.g. in HISTOGRAM or in METAD) we use this technique.

There is thus some flexibility in the particular function we use for $K[\mathbf{r}]$ in the above. The following variants are available.

TYPE	FUNCTION
gaussian	$f(r) = \frac{1}{(2\pi)^n \sqrt{\lvert\Sigma^{-1}\rvert}} \exp\left(-0.5r^2\right)$
triangular	$f(r) = \frac{3}{V}(1-\lvert r\rvert)H(1-\lvert r\rvert)$
uniform	$f(r) = \frac{1}{V}H(1-\lvert r\rvert)$

In the above $H(y)$ is a function that is equal to one when $y > 0$ and zero when $y \leq 0$. n is the dimensionality of the vector $\mathbf{x}$ and V is the volume of an elipse in an n dimensional space which is given by:

$$\begin{aligned} V &= |\Sigma^{-1}| \frac{\pi^{\frac{n}{2}}}{\left(\frac{n}{2}\right)!} \qquad \text{for even} \quad n \\ V &= |\Sigma^{-1}| \frac{2^{\frac{n+1}{2}} \pi^{\frac{n-1}{2}}}{n!!} \end{aligned}$$

In METAD the normalization constants are ignored so that the value of the function at $r = 0$ is equal to one. In addition in METAD we must be able to differentiate the bias in order to get forces. This limits the kernels we can use in this method.

9.8.3 landmarkselection

	This is part of the analysis module

This is currently a filler page. This is currently a filler page.

Just use LANDMARKS=ALL. More complex versions will appear in later versions.

9.8.4 reweighting

	This is part of the analysis module

Calculate free energies from a biassed/higher temperature trajectory. Calculate free energies from a biassed/higher temperature trajectory.

We can use our knowledge of the Boltzmann distribution in the cannonical ensemble to reweight the data contained in trajectories. Using this procedure we can take trajectory at temperature T_1 and use it to extract probabilities at a different temperature, T_2, using:

$$P(s',t) = \frac{\sum_{t'}^{t} \delta(s(x)-s') \exp\left(+(\left[\frac{1}{T_1}-\frac{1}{T_2}\right]\frac{U(x,t')}{k_B}\right)}{\sum_{t'}^{t} \exp\left(+\left[\frac{1}{T_1}-\frac{1}{T_2}\right]\frac{U(x,t')}{k_B}\right)}$$

where $U(x,t')$ is the potential energy of the system. Alternatively, if a static or pseudo-static bias $V(x,t')$ is acting on the system we can remove this bias and get the unbiased probability distribution using:

$$P(s',t) = \frac{\sum_{t'}^{t} \delta(s(x)-s') \exp\left(+\frac{V(x,t')}{k_B T}\right)}{\sum_{t}^{t} \exp\left(+\frac{V(x,t')}{k_B T}\right)}$$

9.8.5 switchingfunction

Functions that measure whether values are less than a certain quantity. Functions that measure whether values are less than a certain quantity.

Switching functions $s(r)$ take a minimum of one input parameter d_0. For $r \leq d_0 \quad s(r) = 1.0$ while for $r > d_0$ the function decays smoothly to 0. The various switching functions available in plumed differ in terms of how this decay is performed.

Where there is an accepted convention in the literature (e.g. COORDINATION) on the form of the switching function we use the convention as the default. However, the flexibility to use different switching functions is always present generally through a single keyword. This keyword generally takes an input with the following form:

```
KEYWORD={TYPE <list of parameters>}
```

The following table contains a list of the various switching functions that are available in plumed 2 together with an example input.

TYPE	FUNCTION	EXAMPLE INPUT	DEFAULT PARAMETERS
RATIONAL	$s(r) = \frac{1-\left(\frac{r-d_0}{r_0}\right)^n}{1-\left(\frac{r-d_0}{r_0}\right)^m}$	{RATIONAL R_0= r_0 D_0= d_0 NN= n MM= m}	$d_0 = 0.0$, $n = 6$, $m = 12$
EXP	$s(r) = \exp\left(-\frac{r-d_0}{r_0}\right)$	{EXP R_0= r_0 D_0= d_0}	$d_0 = 0.0$
GAUSSIAN	$s(r) = \exp\left(-\frac{(r-d_0)^2}{2r_0^2}\right)$	{GAUSSIAN R_0= r_0 D_0= d_0}	$d_0 = 0.0$
SMAP	$s(r) = \left[1+(2^{a/b}-1)\left(\frac{r-d_0}{r_0}\right)\right]^{-b/a}$	{SMAP R_0= r_0 D_0= d_0 A= a B= b}	$d_0 = 0.0$

For all the switching functions in the above table one can also specify a further (optional) parameter using the parameter keyword D_MAX to assert that for $r > d_{\max}$ the switching function can be assumed equal to zero. In this case it is suggested to also use the STRETCH flag, which will bring the switching function smoothly to zero by stretching and shifting it. To be more clear, using

```
KEYWORD={RATIONAL R_0=1 D_MAX=3 STRETCH}
```

the resulting switching function will be $s(r) = \frac{s'(r)-s'(d_{max})}{s'(0)-s'(d_{max})}$ where $s'(r) = \frac{1-r^6}{1-r^{12}}$ Since PLUMED 2.2 this will become the default.

9.8.6 Regular Expressions

When you use a collective variable that has many calculated components and you want to refer to them as arguments you can use regular expressions.

Since version 2.1, plumed takes advantage of a configuration scripts that detects libraries installed on your system. If regex library is found, then you will be able to use regular expressions to refer to collective variables or function names.

Regular expressions are enclosed in round braces and must not contain spaces (the components names have no spaces indeed, so why use them?).

As an example then command

```
d1: DISTANCE ATOMS=1,2 COMPONENTS
PRINT ARG=(d1\.[xy])   STRIDE=100 FILE=colvar FMT=%8.4f
```

will cause both the d1.x and d1.y components of the DISTANCE action to be printed out in the order that they are created by plumed. The "." character must be escaped in order to interpret it as a literal ".". An unescaped dot is a wildcard which is matched by any character, So as an example

```
d1: DISTANCE ATOMS=1,2 COMPONENTS
dxy: DISTANCE ATOMS=1,3

# this will match d1.x,d1.y,dxy
PRINT ARG=(d1.[xy])   STRIDE=100 FILE=colvar FMT=%8.4f

# while this will match d1.x,d1.y only
PRINT ARG=(d1\.[xy])   STRIDE=100 FILE=colvar FMT=%8.4f
```

You can include more than one regular expression by using comma separated regular expressions

```
t1: TORSION ATOMS=5,7,9,15
t2: TORSION ATOMS=7,9,15,17
d1: DISTANCE ATOMS=7,17 COMPONENTS
PRINT ARG=(d1\.[xy]),(t[0-9]) STRIDE=100 FILE=colvar FMT=%8.4f
```

(this selects t1,t2,d1.x and d2.x) Be aware that if you have overlapping selection they will be duplicated so it a better alternative is to use the "or" operator "|".

```
t1: TORSION ATOMS=5,7,9,15
t2: TORSION ATOMS=7,9,15,17
d1: DISTANCE ATOMS=7,17 COMPONENTS
PRINT ARG=(d1\.[xy]|t[0-9]) STRIDE=100 FILE=colvar FMT=%8.4f
```

this selects the same set of arguments as the previous example.

You can check the log to see whether or not your regular expression is picking the set of components you desire.

For more information on regular expressions visit http://www.regular-expressions.info/reference.html.

9.8.7 Files

We tried to design PLUMED in such a manner that input/output is done consistently irrespectively of the file type. Most of the files written or read by PLUMED thus follow the very same conventions discussed below.

9.8.7.1 Restart

Whenever the RESTART option is used, all the files written by PLUMED are appended. This makes it easy to analyze results of simulations performed as a chain of several sub-runs. Notice that most of the PLUMED textual files have a header. The header is repeated at every restart. Additionally, several files have time in the first column. PLUMED just takes the value of the physical time from the MD engine. As such, you could have that time starts again from zero upon restart or not.

An exception from this behavior is given by files which are not growing as the simulation proceeds. For example, grids written with METAD with GRID_WFILE are overwritten by default during the simulation. As such, when restarting, there is no point in appending the file. Internally, PLUMED opens the file in append mode but then rewinds it every time a new grid is dumped.

9.8.7.2 Backup

Whenever the RESTART option is not used, PLUMED tries to write new files. If an old file is found in the way, PLUMED takes a backup named "bck.X.filename" where X is a progressive number. Notice that by default PLUMED

only allows a maximum of 100 backup copies for a file. This behavior can be changed by setting the environment variable PLUMED_MAXBACKUP to the desired number of copies. E.g. export PLUMED_MAXBACKUP=10 will fail after 10 copies. PLUMED_MAXBACKUP=-1 will never fail - be careful since your disk might fill up quickly with this setting.

9.8.7.3 Replica suffix

When running with multiple replicas (e.g., with GROMACS, -multi option) PLUMED adds the replica index as a suffix to all the files. The following command will thus print files named COLVAR.0, COLVAR.1, etc for the different replicas.

```
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=COLVAR
```

(see also DISTANCE and PRINT).

When reading a file, PLUMED will try to add the suffix. If the file is not found, it will fall back to the name without suffix. The most important case is the reading of the plumed input file. If you provide a file for each replica (e.g. plumed.dat.0, plumed.dat.1, etc) you will be able to setup plumed differently on each replica. On the other hand, using a single plumed.dat will make all the replicas read the same file.

9.8.7.3.1 Suffixes and file extension

When PLUMED adds the replica suffix, it recognizes some file extension and add the suffix *before* the extension. The only suffix recognized by PLUMED 2.1 is ".gz". This means that using

```
d: DISTANCE ATOMS=1,2
PRINT ARG=d FILE=COLVAR.gz
```

will write files named COLVAR.0.gz, COLVAR.1.gz, etc. This is useful since the preserved extension makes it easy to process the files later. In future PLUMED versions this behavior might change and more extensions could be recognized.

Chapter 10

Tutorials

The following pages describe how to perform a variety of tasks using PLUMED 2

Belfast tutorial: Analyzing CVs	This tutorial explains how to use plumed to analyze CVs
Belfast tutorial: Adaptive variables I	How to use path CVs
Belfast tutorial: Adaptive variables II	Dimensionality reduction and sketch maps
Belfast tutorial: Umbrella sampling	Umbrella sampling, reweighting, and weighted histogram
Belfast tutorial: Out of equilibrium dynamics	How to run a steered MD simulations and how to estimate the free energy
Belfast tutorial: Metadynamics	How to run a metadynamics simulation
Belfast tutorial: Replica exchange I	Parallel tempering and Metadynamics, Well-Tempered Ensemble
Belfast tutorial: Replica exchange II and Multiple walkers	Bias exchange and multiple walkers
Belfast tutorial: NMR constraints	NMR constraints
Belfast tutorial: Steinhardt Parameters	Steinhardt Parameters
Cambridge tutorial	A short 2 hours tutorial that introduces Well-Tempered Metadynamics, Bias-Exchange Metadynamics and Replica-Average Metadynamics
Moving from PLUMED 1 to PLUMED 2	This tutorial explains how plumed 1 input files can be translated into the new plumed 2 syntax.
Munster tutorial	A short 3 hours tutorial that introduces analysis, well-tempered metadynamics, and multiple-restraints umbrella sampling.

In addition, the following websites contain resources that might be helpful

http://www.youtube.com/watch?v=iDv↩ ZmbWE5ps	A short video introduction to the use of multicolvars in PLUMED 2
http://www.youtube.com/watch?v=PxJ↩ P16qNCYs	A short video introduction to the syntax of the PLUMED 2 input file
http://en.wikipedia.org/wiki/↩ Metadynamics	A wikipedia article on metadynamics

10.1 Belfast tutorial: Analyzing CVs

10.1.1 Aims

The aim of this tutorial is to introduce the users to the plumed syntax. We will go through the writing of simple collective variable and we will use them to analyse a trajectory in terms of probabilty distributions and free energy.

10.1.2 Learning Outcomes

Once this tutorial is completed students will:

- Know how to write a simple plumed input file
- Know how to analyse a trajectory using plumed

10.1.3 Resources

The `tarball` for this project contains the following files:

- trajectory-short.xyz : a (short) trajectory for a 16 residue protein in xyz format. All calculations with plumed driver use this trajectory.
- template.pdb : a single frame from the trajectory that can be used in conjuction with the MOLINFO command

10.1.4 Instructions

PLUMED2 is a library that can be accessed by multiple codes adding a relatively simple and well documented interface. Once PLUMED is installed you can run a plumed executable that can be used for multiple purposes:

```
plumed --help
```

some of the listed options report about the plumed available functionalities, other can be used to tell plumed to do something: from analysing a trajectory to patch the source code of a MD code and so on. All the commands have further options that can be seen using plumed command –help, i.e.:

```
plumed driver --help
```

In the following we are going to see how to write an input file for plumed2 that can be used to analyse a trajectory.

10.1.4.1 A note on units

By default the PLUMED inputs and outputs quantities in the following units:

- Energy - kJ/mol
- Length - nanometers
- Time - picoseconds

If you want to change these units you can do this using the UNITS keyword.

10.1.4.2 Introduction to the PLUMED input file

A typical input file for PLUMED input is composed by specification of one or more CVs, the printout frequency and a termination line. Comments are denoted with a # and the termination of the input for PLUMED is marked with the keyword ENDPLUMED. Whatever it follows is ignored by PLUMED. You can introduce blank lines. They are not interpreted by PLUMED.

In the following input we will analyse the DISTANCE between the two terminal carbons of a 16 residues peptide, and we will PRINT the results in file named COLVAR.

```
#my first plumed input:
DISTANCE ATOMS=2,253 LABEL=e2edist

#printout frequency
PRINT ARG=e2edist STRIDE=1 FILE=COLVAR

#endofinput
ENDPLUMED
here I can write what I want it won't be read.
```

Now we can use this simple input file to analyse the trajectory included in the RESOURCES:

```
plumed driver --plumed plumed.dat --ixyz trajectory-short.xyz --length-units 0.1
```

NOTE: –length-units 0.1, xyz files, as well as pdb files, are in Angstrom.

You should have a file COLVAR, if you look at it (i.e. more COLVAR) the first two lines should be:

```
#! FIELDS time e2edist
 0.000000 2.5613161
```

NOTE: the first line of the file COLVAR tells you what is the content of each column.

In PLUMED2 the commands defined in the input files are executed in the same order in which they are written, this means that the following input file is wrong:

```
#printout frequency
PRINT ARG=cvdist STRIDE=1 FILE=COLVAR
#my first plumed input:
DISTANCE ATOMS=2,253 LABEL=e2edist
#endofinput
ENDPLUMED
here I can write what I want it won't be read.
```

Try to run it.

Sometimes, when calculating a collective variable, you may not want to use the positions of a number of atoms directly. Instead you may wish to use the position of a virtual atom whose position is generated based on the positions of a collection of other atoms. For example you might want to use the center of mass of a group of atoms (COM):

Since PLUMED executes the input in order you need to define the new Virtual Atom before using it:

```
first: COM ATOMS=1,2,3,4,5,6
last: COM ATOMS=251-256

e2edist: DISTANCE ATOMS=2,253
comdist: DISTANCE ATOMS=first,last

PRINT ARG=e2edist,comdist STRIDE=1 FILE=COLVAR

ENDPLUMED
```

NOTE: an action (i.e. COM or DISTANCE here) can be either label using LABEL as we did before or as label: ACTION as we have just done here.

With the above input this is what happen inside PLUMED with a STRIDE=1:

1. calculates the position of the Virtual Atom 'first' as the COM of atoms from 1 to 6;
2. calculates the position of the Virtual Atom 'last' as the COM of atoms from 251 to 256;
3. calculates the distance between atoms 2 and 253 and saves it in 'e2edist';
4. calculates the distance between the two atoms 'first' and 'last' and saves it in 'comdist';

5. print the content of 'e2edist' and 'comdist' in the file COLVAR

In the above input we have used to different ways of writing the atoms used in COM calculation:

1. ATOMS=1,2,3,4,5,6 is the explicit list of the atoms we need
2. ATOMS=251-256 is the range of atoms needed

ranges of atoms can be defined with a stride which can also be negative:

1. ATOMS=from,to:by (i.e.: 251-256:2)
2. ATOMS=to,from:-by (i.e.: 256-251:-2)

Now by plotting the content of the COLVAR file we can compare the behaviour in this trajectory of both the terminal carbons as well as of the centre of masses of the terminal residues.

```
gnuplot
```

What do you expect to see now by looking at the trajectory? Let's have a look at it

```
vmd template.pdb trajectory-short.xyz
```

Virtual atoms can be used in place of standard atoms everywhere an atom can be given as input, they can also be used together with standard atoms. So for example we can analyse the TORSION angle for a set of Virtual and Standard atoms:

```
first: COM ATOMS=1-6
last: COM ATOMS=251-256
cvtor: TORSION ATOMS=first,102,138,last

PRINT ARG=cvtor STRIDE=1 FILE=COLVAR

ENDPLUMED
```

The above CV don't look smart to learn something about the system we are looking at. In principle CV are used to reduce the complexity of a system by looking at a small number of properties that could be enough to rationalise its behaviour.

Now try to write a collective variable that measures the Radius of Gyration of the system: GYRATION.

NOTE: if what you need for one or more variables is a long list of atoms and not a virtual atom one can use the keyword GROUP. A GROUP can be defined using ATOMS in the same way we saw before, in addition it is also possible to define a GROUP by reading a GROMACS index file.

```
ca: GROUP ATOMS=9,16,31,55,69,90,102,114,124,138,160,174,194,208,224,238
```

Now 'ca' is not a virtual atom but a simple list of atoms.

10.1.4.3 MULTICOLVAR

Sometimes it can be useful to calculate properties of many similar collective variables at the same time, for example one can be interested in calculating the properties of the distances between a group of atoms, or properties linked to the distribution of the dihedral angles of a chain and so on. In PLUMED2 this kind of collective variables fall under the name of MULTICOLVAR (cf. MultiColvar Documentation.) Here we are going to analyse the distances between CA carbons along the chain:

```
ca: GROUP ATOMS=9,16,31,55,69,90,102,114,124,138,160,174,194,208,224,238
dd: DISTANCES GROUP=ca MEAN MIN={BETA=50} MAX={BETA=0.02} MOMENTS=2

PRINT ARG=dd.mean,dd.min,dd.max,dd.moment-2 STRIDE=1 FILE=COLVAR

ENDPLUMED
```

The above input tells PLUMED to calculate all the distances between CA carbons and then look for the mean distance, the minimum distance, the maximum distance and the variance. In this way we have defined four collective variables that are calculated using the distances. These four collective variables are stored as components of the defined action 'dd': dd.mean, dd.min, dd.max, dd.moment-2.

The infrastracture of multicolvar has been used to develop many PLUMED2 collective variables as for example the set of Secondary Structure CVs (ANTIBETARMSD, PARABETARMSD and ALPHARMSD).

```
MOLINFO STRUCTURE=template.pdb
abeta: ANTIBETARMSD RESIDUES=all TYPE=DRMSD LESS_THAN={RATIONAL R_0=0.08 NN=8 MM=12} STRANDS_CUTOFF=1

PRINT ARG=abeta.lessthan STRIDE=1 FILE=COLVAR

ENDPLUMED
```

We have now seen how to write the input some of the many CVs available in PLUMED. More complex CVs will be discussed in the next workshop, Belfast tutorial: Adaptive variables I.

10.1.4.4 Analysis of Collective Variables

Collective variables are usually used to visualize the Free Energy of a system. Given a system evolving at fixed temperature, fixed number of particles and fixed volume, it will explore different conformations with a probability

$$P(q) \propto e^{-\frac{U(q)}{kb_BT}}$$

where q are the microscopic coordinates and k_B is the Boltzmann constant.

It is possible to analyse the above probabilty as a function of one or more collective variable $s(q)$:

$$P(s) \propto \int dq e^{-\frac{U(q)}{kb_BT}} \delta(s-s(q))$$

where the δ function means that to for a given value s of the collective variable are counted only those conformations for which the CV is s. The probability can be recast to a free energy by taking its logarithm:

$$F(s) = -k_BT \log P(s)$$

This means that by estimating the probability distribution of a CV it is possible to know the free energy of a system along that CV. Estimating the probability distribution of the conformations of a system is what is called 'sampling'.

In order to estimate a probability distribution one needs to make HISTOGRAM from the calculated CVs. PLUMED2 includes the possibility of histogramming data both on the fly as well as a posteriori as we are going to do now.

```
MOLINFO STRUCTURE=template.pdb
abeta: ANTIBETARMSD RESIDUES=all TYPE=DRMSD LESS_THAN={RATIONAL R_0=0.08 NN=8 MM=12} STRANDS_CUTOFF=1
ca: GROUP ATOMS=9,16,31,55,69,90,102,114,124,138,160,174,194,208,224,238
DISTANCES ...
GROUP=ca MEAN MIN={BETA=50} MAX={BETA=0.02} MOMENTS=2 LABEL=dd
... DISTANCES

PRINT ARG=abeta.lessthan,dd.mean,dd.min,dd.max,dd.moment-2 STRIDE=1 FILE=COLVAR

HISTOGRAM ...
ARG=abeta.lessthan,dd.mean
USE_ALL_DATA
```

```
KERNEL=discrete
GRID_MIN=0,0.8
GRID_MAX=4,1.2
GRID_BIN=40,40
GRID_WFILE=histo
... HISTOGRAM

ENDPLUMED
```

NOTE: HISTOGRAM ... means that what follow is part of the HISTOGRAM function, the same can be done for any action in PLUMED.

The above input tells PLUMED to accumulate the two collective variables on a GRID. In addition the probability can be converted to a free-energy using the flag FREE-ENERGY and setting the temperature using TEMP (i.e. 300K). Histograms can be accumulated in a smoother way by using a KERNEL function, a kernel is a normalised function, for example a normalised gaussian is the default kernel in PLUMED, that is added to the histogram centered in the position of the data. Estimating a probability density using kernels can in principle give more accurate results, on the other hand in addition to the choice of the binning one has to choose a parameter that is the WIDTH of the kernel function. As a rule of thumb: the grid spacing should be smaller (i.e. one half or less) than the BANDWIDTH and the BANDWIDTH should be smaller (i.e. one order of magnitude) than the variance observed/expected for the variable.

```
HISTOGRAM ...
ARG=abeta.lessthan,dd.mean
USE_ALL_DATA
GRID_MIN=0,0.8
GRID_MAX=4,1.2
GRID_SPACING=0.04,0.004
BANDWIDTH=0.08,0.008
GRID_WFILE=histo
... HISTOGRAM

ENDPLUMED
```

If you have time less at the end of the session read the manual and look for alternative collective variables to analyse the trajectory. Furthemore try to play with the HISTOGRAM parameters to see the effect of using KERNEL in analysing data.

10.2 Belfast tutorial: Adaptive variables I

10.2.1 Aim

In this section we want to introduce the concept of adaptive collective variables. These are special variables that are knowledge-based in that are built from a pre-existing notion of the mechanism of the transition under study

10.2.2 Resources

Here is the tarball with the files referenced in the following.

10.2.3 What happens when in a complex reaction?

When you deal with a complex conformational transition that you want to analyze (or bias), very often you cannot just describe it with a single order parameter.

As an example in Figure belfast-2-cdk-fig I consider a large conformational transition like those involved in activating the kinase via open-close transition of the activation loop. In sticks you see the part involved in the large conformational change, the rest is either keeping the structure and just moving a bit or is a badly resolved region in the

X-ray structure. This is a complex transition and it is hard to tell which is the order parameter that best describes the transition.

One could identify a distance that can distinguish open from close but

- the plasicity of the loop is such that the same distance can correspond to an almost closed loop and almost open loop. One would like to completely divide these two situations with something which is discriminating what intuitively one would think as open and closed
- the transition state is an important point where one would like to see a certain crucial interaction forming/breaking so to better explain what is really happening. If you capture then hypothetically you would be able to say what is dictating the rate of this conformational transition. A generic distance is a very hybrid measure that is unlikely to capture a salt-bridge formation and a concerted change of many dihedral change or desolvation contribution which are happening while the transition is happening. All these things are potentially important in such transition but none of them is explaining the whole thing.

So basically in these cases you have to deal with an intrinsic multidimensional collective variable where you would need many dimensions. How would you visualize a 10 dimensional CV where you use many distances, coordinations and dihedrals (ouch, they're periodic too!) ?

Another typical case is the docking of a small molecule in a protein cleft or gorge, which is the mechanism of drug action. This involves specific conformational transition from both the small molecule and the protein as the small molecule approaches the protein cavity. This also might imply a specific desolvation pattern.

Other typical examples are chemical reactions. Nucleophiloic attacks typically happen with a role from the solvent (see some nice paper from Nobel-prize winner Arieh Warshel) and sizeable geometric distortions of the neighboring groups.

10.2.4 Path collective variables

One possibility to describe many different thing that happen in a single reaction is to use a dimensional reduction technique and in plumed the simplest example that may show its usefulness can be considered that of the path collective variables.

In a nutshell, your reaction might be very complex and happening in many degree of freedom but intuitively is a sort of track along which the reaction proceeds. So what we need is a coordinate that, given a conformation, just tells which point along the "reactive track" is closest.

For example, in Fig. belfast-2-ab-fig, you see a typical chemical reaction (hydrolisys of methylphosphate) with the two end-points denoted by A and B. If you are given a third point, just by looking at it, you might find that this is more resemblant to the reactant than the product, so, hypothetically, if you would intuitively give a parameter that would be 1 for a configuration in the A state and 2 for a configuration in the B state, you probably would give it something like 1.3, right?

Path collective variables are the extension to this concept in the case you have many conformation that describe your path, and therefore, instead of an index that goes from 1 to 2 you have an index that goes from 1 to N , where N is the number of conformation that you use in input to describe your path.

From a mathematical point of view, that's rather simple. The progress along the path is calculated with the following equation:

$$S(X) = \frac{\sum_{i=1}^{N} i \, \exp^{-\lambda |X - X_i|}}{\sum_{i=1}^{N} \exp^{-\lambda |X - X_i|}}$$

where in belfast-2-s-eq the $|X - X_i|$ represents a distance between one configuration X which is analyzed and another from the set that compose the path X_i. The parameter λ is a positive value that is tuned in a way explaned later. here are a number of things to note to make you think that this is exactely what you want.

- The negative exponential function is something that is 1 whenever the value at the exponent is zero, and is progressively smaller when the value is larger than zero (trivially, the case with the value at the exponent larger than zero never occurs since lambda is a positive quantity and the distance is by definition positive).

- Whenever you sit exactly on a specific images X_j then all the other terms in the sum disappear (if λ is large enough) and only the value j survives returning exactly $S(X) = j$.

In order to provide a value which is continuous, the parameter λ should be correclty tuned. As a rule of thumb I use the following formula

$$\lambda = \frac{2.3(N-1)}{\sum_{i=1}^{N-1} |X_i - X_{i+1}|}$$

which imply that one should calculate the average distance between consecutive frames composing the path. Note also that this distance should be more or less similar between the frames. Generally I tolerate fluctuation of the order of 10/15 percent tops. If you have larger, then it is better to have a smaller value of λ.

It is important to note that in principle one could even have a specific λ value associated to each frame of the path but this would provide some distortion in the diffucion coefficient which could potentially harm a straightforward interpretation of the free energy landscape.

So, at this point is better to understand what is meant with "distance" since a distance between two conformations can be calculated in very many ways. The way we refer here is by using mean square deviation after optimal alignment. This means that at each step in which the analisys is performed, a number N of optimal alignments is performed. Namely what is calculated is $|X - X_i| = d(X, X_i)^2$ where $d(X, X_i)$ is the RMSD as defined in what you see here RMSD.

Using the MSD instead of RMSD is sometimes more convenient and more stable (you do not have a denominator that gies to zero in the derivatives when biasing.

Anyway this is a matter of choice. Potentially one could equally employ other metrics like a set of coordinations (this was done in the past), and then you would avoid the problem of rototranslations (well, which is not a problem since you have it already in plumed) but for some applications that might become appealing. So in path collective variables (and in all the dimensional reduction based collective variables) the problem is converted from the side of choosing the collective variable in choosing the right way to calculate distances, also called "metrics".

The discussion of this issue is well beyond the topic of this tutorial, so we can move forward in how to tell plumed to calculate the progress along the path whenever the MSD after optimal alignment is used as distance.

```
p1: PATHMSD REFERENCE=all.pdb LAMBDA=50.0
PRINT ARG=p1.sss,p1.zzz STRIDE=100 FILE=colvar FMT=%8.4f
```

Note that reference contains a set of PDB, appended one after the other, with a END field. Note that there is no need to place all the atoms of the system in the PDB reference file you provide. Just put the atoms that you think might be needed. You can leave out big domains, solvent and ions if you think that is not important for your use.

Additionally, note that the measure units of LAMBDA are in the units of the code. In gromacs they are in nm^2 while NAMD is Ang^2. PATHMSD produces two arguments that can be printed or used in other ActionWithArguments. One is the progress along the path of belfast-2-s-eq, the other is the distance from the closest point along the path, which is denoted with the zzz component. This is defined as

$$Z(X) = -\frac{1}{\lambda} \log\left(\sum_{i=1}^{N} \exp^{-\lambda |X - X_i|}\right)$$

It is easy to understand that in case of perfect match of $X = X_i$ this equation gives back the value of $|X - X_i|$ provided that the lambda is adjusted correctly.

So, the two variables, put together can be visualized as This variable is important because whenever your simulation is running close to the path (low Z values), then you know that you are reproducing reliably the path you provided in input but if by chance you find some other path that goes, say, from $S = 1, Z = 0$ to $S = N, Z = 0$ via large Z values, then it might well be that you have just discovered a good alternative pathway. If your path indeed is going from $S = 1, Z = large$ to $S = N, Z = large$ then it might well be that you do not have your reaction accomplished, since your reaction, by definition should go from the reactant which is located at $S = 1, Z = 0$ to the product, which is located at $S = 1, Z = N$ so you should pay attention. This case is exemplified in Fig. belfast-2-ab-sz-nowhere-fig

10.2.5 A note on the path topology

A truly important point is that if you get a trajectory from some form of accelerated dynamics (e.g. simply by heating) this cannot simply be converted into a path. Since it is likely that your trajectory is going stochastically back and forth (not in the case of SMD or US, discussed later), your trajectory might be not topologically suitable. To understand that, suppose you simply collect a reactive trajectory of 100 ps into the reference path you give to the PATHMSD and simply you assign the frame of 1 ps to index 1 (first frame occurring in the reference file provided to PATHMSD), the frame of 2 ps to index 2 and so on : it might be that you have two points which are really similar but one is associated to step, say 5 and the other is associated with frame 12. When you analyse the same trajectory, when you are sitting on any of those points then the calculation of S will be an average like $S(X) = (5+12)/2 = 8.5$ which is an average of the two indexes and is completely misleading sinec it let you think that you are moving between point 8 and 9, which is not the case. So this evidences that your reference frames should be "topologically consecutive". This means that frame 1 should be the closest to frame 2 and all the other frames should be farther apart. Similarly frame 2 should be equally close (in an RMSD sense) to 1 and 3 while all the others should be farther apart. Same for frame 3: this should be closest to frame 2 and 4 and farther apart from all the others and so on. This is equivalent to calculate an "RMSD matrix" which can be conveniently done in vmd (this is a good exercise for a number of reasons) with RMSD Trajectory tools, by choosing different reference system along the set of reference frames.

This is shown in Fig. belfast-2-good-matrix-fig where the matrix has a typical gullwing shape.

On the contrary, whenever you extract the frames from a pdb that you produced via free MD or some biased methods (SMD or Targeted MD for example) then your frame-to-frame distance is rather inhomogeneous and looks something like

Aside from the general shape, which is important to keep the conformation-to-index relation (this, as we will see in the next part is crucial in the multidimensional scaling) the crucial thing is the distance between neighbors.

As a matter of fact, this is not much important in the analysis but is truly crucial in the bias. When biasing a simulation, you generally want to introduce a force that push your system somewhere. In particular, when you add a bias which is based on a path collective variable, most likely you want that your system goes back and forth along your path. The bias is generally applied by an additional term in the hamiltonian, this can be a spring term for Umbrella Sampling, a Gaussian function for Metadynamics or whatever term which is a function of the collective variable s. Therefore the Hamiltonian $H(X)$ where X is the point of in the configurational phase space where your system is takes the following form

$$H'(X) = H(X) + U(S(X))$$

where $U(S(X))$ is the force term which depends on the collective variable that ultimately is a function of the X. Now, when you use biased dynamics you need to evolve according the forces that this term produces (this only holds for MD, while not in MC) and therefore you need

$$F_i = -\frac{dH'(X)}{dx_i} = -\frac{dH'(X)}{dx_i} - \frac{\partial U(S(X))}{\partial S}\frac{\partial S(X)}{\partial x_i}$$

This underlines the fact that, whenever $\frac{\partial S(X)}{\partial x_i}$ is zero, then you have no force on the system. Now the derivative of the progress along the path is

$$\frac{\partial S(X)}{\partial x_i} = \frac{\sum_{i=1}^{N} -\lambda\ i\ \frac{\partial |X-X_i|}{\partial x_i} \exp^{-\lambda|X-X_i|}}{\sum_{i=1}^{N} \exp^{-\lambda|X-X_i|}} - \frac{(\sum_{i=1}^{N} i\ \exp^{-\lambda|X-X_i|})(\sum_{j=1}^{N} -\lambda \frac{\partial |X-X_j|}{\partial x_i} \exp^{-\lambda|X-X_j|})}{(\sum_{i=1}^{N} \exp^{-\lambda|X-X_i|})^2} = \frac{\sum_{i=1}^{N} -\lambda\ i\ \frac{\partial |X-X_i|}{\partial x_i} \exp^{-\lambda|X-X_i|}}{\sum_{i=1}^{N} \exp^{-\lambda|X-X_i|}} - s($$

which can be rewritten as

$$\frac{\partial S(X)}{\partial x_i} = \lambda \frac{\sum_{i=1}^{N} \frac{\partial |X-X_i|}{\partial x_i} \exp^{-\lambda|X-X_i|}[s(X)-i]}{\sum_{i=1}^{N} \exp^{-\lambda|X-X_i|}}$$

It is interesting to note that whenever the λ is too small the force will vanish. Additionally, when λ is too large, then it one single exponential term will dominate over the other on a large part of phase space while the other will vanish. This means that the $S(X)$ will assume almost discrete values that produce zero force. Funny, isn't it?

10.2.6 How many frames do I need?

A very common question that comes is the following: "I have my reaction or a model of it. how many frames do I need to properly define a path collective variable?" This is a very important point that requires a bit of thinking. It all depends on the limiting scale in your reaction. For example, if in your process you have a torsion, as the smallest event that you want to capture with path collective variable, then it is important that you mimick that torsion in the path and that this does not contain simply the initial and final point but also some intermediate. Similarly, if you have a concerted bond breaking, it might be that all takes place in the range of an Angstrom or so. In this case you should have intermediate frames that cover the sub-Angstrom scale. If you have both in the same path, then the smallest dominates and you have to mimick also the torsion with sub-Angstrom accuracy.

10.2.7 Some tricks of the trade: the neighbors list.

If it happens that you have a very complex and detailed path to use, say that it contains 100 frames with 200 atoms each, then the calculation of a 100 alignment is required every time you need the CV. This can be quite expensive but you can use a trick. If your trajectory is continuous and you are sure that your trajectory does not show jumps where your system suddedly move from the reactant to the product, then you can use a so-called neighbor list. The plumed input shown before then becomes

```
p1: PATHMSD REFERENCE=all.pdb LAMBDA=50.0 NEIGH_STRIDE=100 NEIGH_SIZE=10
PRINT ARG=p1.sss,p1.zzz STRIDE=100 FILE=colvar FMT=%8.4f
```

and in this case only the closest 10 frames from the path will be used for the CV. Then the list of the frames to use is updated every 100 steps. If you are using a biased dynamics this may introduce sudden change in the derivatives, therefore it is better to check the stability of the setup before running production-quality calculations.

10.2.8 The molecule of the day: alanine dipeptide

Here and probably in other parts of the tutorial a simple molecule is used as a test case. This is alanine dipeptide in vacuum. This rather simple molecule is useful to make many benchmark that are around for data analysis and free energy methods. It is a rather nice example since it presents two states separated by a high (free) energy barrier.

In Fig. belfast-2-ala-fig its structure is shown.

The two main metastable states are called C_7eq and C_7ax.

Here metastable states are intended as states which have a relatively low free energy compared to adjacent conformation. At this stage it is not really useful to know what is the free energy, just think in term of internal energy. This is almost the same for such a small system whith so few degrees of freedom.

It is conventional use to show the two states in terms of Ramachandran dihedral angles, which are denoted with Φ and Ψ in Fig. belfast-2-transition-fig .

10.2.9 Examples

Now as a simple example, I want to show you that plotting some free dynamics trajectories shoot from the saddle point, you get a different plot in the path collective variables if you use the right path or if you use the wrong path.

In Fig. belfast-2-good-bad-path-fig I show you two example of possible path that join the C_7eq and C_7ax metastable states in alanine dipeptide. You might clearly expect that real (rare) trajectories that move from one basin to the other would rather move along the black line than on the red line.

So, in this example we do a sort of "commmittor analysis" where we start shooting a number of free molecular dynamics from the saddle point located at $\Phi = 0$ and $\Psi = -1$ and we want to see which way do they go. Intuitively, by assigning random velocities every time we should find part of the trajectories that move woward C_7eq and part that move towards C_7ax.

I provided you with two directories, each containing a bash script script.sh whose core (it is a bit more complicated in practice...) consists in:

```
#
# set how many runs you want to do
#
ntests=50
for i in `seq 1 $ntests`
do
        #
        # assign a random velocity at each timestep by initializing the
        #
        sed s/SEED/$RANDOM/ md.mdp >newmd.mdp
        #
        # do the topology: this should write a topol.tpr
        #
        $GROMPP -c start.gro -p topol.top -f newmd.mdp
        $GROMACS_BIN/$MDRUN -plumed plumed.dat
        mv colvar colvar_$i
done
```

This runs 50 short MD runs (few hundreds steps) and just saves the colvar file into a labeled colvar file. In each mdrun plumed is used to plot the collective variables and it is something that reads like the follwing:

```
# Phi
t1: TORSION ATOMS=5,7,9,15
# Psi
t2: TORSION ATOMS=7,9,15,17
# The right path
p1: PATHMSD REFERENCE=right_path.dat LAMBDA=15100.
# The wrong path
p2: PATHMSD REFERENCE=wrong_path.dat  LAMBDA=8244.4
# Just a printout of all the stuff calculated so far
PRINT ARG=* STRIDE=2 FILE=colvar FMT=%12.8f
```

where I just want to plot Φ , Ψ and the two path collective variables. Note that each path has a different LAMBDA parameters. Here the Ramachandran angles are plotted so you can realize which path is the system walking in a more confortable projection. This is of course fine in such a small system but whenever you have to deal with larger systems and control hundreds of CVs at the same time, I think that path collective variables produce a more intuituve description for what you want to do.

If you run the script simply with

```
pd@plumed:~> ./script.sh
```

then after a minute or so, you should have a directory which is full of colvar files. Let's revise together how the colvar file is formatted:

```
#! FIELDS time t1 t2 p1.sss p1.zzz p2.sss p2.zzz
#! SET min_t1 -pi
#! SET max_t1 pi
#! SET min_t2 -pi
#! SET max_t2 pi
 0.000000  -0.17752998  -1.01329788  13.87216908   0.00005492  12.00532256   0.00233905
 0.004000  -0.13370142  -1.10611136  13.87613508   0.00004823  12.03390658   0.00255806
 0.008000  -0.15633049  -1.14298481  13.88290617   0.00004511  12.07203319   0.00273764
 0.012000  -0.23856451  -1.12343958  13.89969608   0.00004267  12.12872544   0.00284883
...
```

In first column you have the time, then t1 (Φ) , then t2 (Ψ) and the path collective variables p1 and p2. Note that the action PATHMSD calculates both the progress along the path (p1.sss) and the distance from it (p1.zzz) . In PLUMED jargon, these are called "components". So a single Action (a line in plumed input) can calculate many components at the same time. This is not always the case: sometimes by default you have one component but specific flags may enable more components to be calculated (see DISTANCE for example). Note that the header (all the part of a colvar file that contains # as first character) is telling already what it inside the file and eventually also tells you if a variable is contained in boundaries (for example torsions, are periodic and their codomain is defined in -Pi and Pi).

At the end of the script, you also have two additional scripts. One is named script_rama.gplt and the other is named script_path.gplt. They contain some gnuplot commands that are very handy to visualize all the colvar files without making you load one by one, that would be a pain.

Now, let's visualize the result from the wrong path directory. In order to do so, after having run the calculation, then do

```
pd@plumed:~>gnuplot
gnuplot> load "script_rama.gplt"
```

what you see is that all the trajectories join the reactant and the product state along the low free energy path depicted before.

Now if you try to load the same bunch of trajectories as a function of the progress along the path and the distance from the path in the case of the wrong path then simply do

```
gnuplot> load "script_path_wrong.gplt"
```

What do you see? A number of trajectories move from the middle towards the right bottom side at low distance from the path. In the middle of the progress along the path, you have higher distance. This is expected since the distance zero corresponds to a unlikely, highly-energetic path which is unlikely to occur. Differently, if you now do

```
gnuplot> load "script_path_right.gplt"
```

You see that the path, compared to what happened before, run much closer to small distance from the path. This means that the provided path is highlt resemblant and representative of what hapens in the reactive path.

Note that going at high distances can be also beneficial. It might help you to explore alternative paths that you have not explored before. But beware, the more you get far from the path, the more states are accessible, in a similar way as the fact that the surface of a sphere increase with its radius. The surface ramps up much faster than the radius therefore you have a lots of states there. This means also high entropy, so many systems actually tend to drift to high distances while, on the contrary, zero distance is never reached in practice (zero entropy system is impossible to reach at finite temperature). So you can see by yourself that this can be a big can of worms. In particular, my experience with path collective variables and biological systems tells me that most of time is hopeless to go to high distances to find new path in many cases (for example, in folding). While this is worth whenever you think that the paths are not too many (alternative routes in chemical reaction or enzymatic catalysis).

10.2.10 How to format my input?

Very often it is asked how to format a set of pdb to be suitably used with path collective variables. Here are some tricks.

- When you dump the files with vmd or (for gromacs users, using trjcat), the pdb you obtain is reindexed from 1. This is also the case when you select a subensemble of atoms of the path (e.g. the heavy atoms only or the backbone atoms). This is rather unfortunate and you have to fix is someway. My preference is to dump the whole pdb but water (when I do not need it) and use some awk script to select the atoms I am interested in.
- Pay attention to the last two column. These are occupancy and beta. With the first (occupancy) you set the atoms which are used to perform the alignment. The atoms which have zero occupancy will not be used in the alignment. The second column is beta and controls which atoms are used for the calculation of the distance after having performed the alignment on the set of atoms which have nonzero occupancy column. In this way you can align all your system by using a part of the system and calculate the distance respect to another set. This is handy in case of protein-ligand. You set the alignment of the protein and you calculate the distance based on the ligand and the part of the protein which is in contact with the protein. This is done for example in this article.
- Plumed-GUI (version $>$ 2.0) provides the *Structure->Build reference structure...* function to generate inputs that conform to the above rules from within VMD.
- Note that all the atoms contained in the REFERENCE must be the same. You cannot have a variable number of atoms in each pdb contained in the reference.

- The reference is composed as a set of concatenated PDBs that are interrupted by a TER/END/ENDMDL card. Both HETATM and ATOM cards denote the atoms of the set.
- Include in the reference frames only the needed atoms. For example, if you have a methyl group involved in a conformational transition, it might be that you do not want to include the hydrogen atoms of the methyl since these rotate fast and probably they do not play ant relevant role.

10.2.11 Fast forward: metadynamics on the path

This section is actually set a bit foward but I included here for completeness now. It is recommended to be read after you have an introduction on Metadynamics and to well-tempered Metadynamics in particular. Here I want to show a couple of concept together.

- Path collective variables can be used for exploring alternative routes. It is effective in highly structure molecules, while it is tricky on complex molecules whenever you have many competing routes
- Path collective variables suffer from problems at the endpoints (as the higly popular coordinates COORDINATION for example) that can be cured with flexible hills and an appropriate reweighting procedure within the well-tempered Metadynamics scheme.

Let's go to the last problem. All comes from the derivative belfast-2-sder-eq. Whenever you have a point of phase space which is similar to one of the endpoint than one of the points in the center then you get a $s(X)$ which is 1 or N (where N is the number of frames composing the path collective variable). In this case that exponential will dominate the others and you are left with a constant (since the derivative of RMSD is a constant since it is linear in space). This means that, no matter what happens here, you have small force. Additionally you have small motion in the CV space. You can move a lot in configuration space but if the closest point is one of the endpoint, your CV value will always be one of the endpoint itself. So, if you use a fixed width of your CV which you retrieve from a middle point in your path, this is not suitable at all at the endpoints where your CV flucutates much less. On the contrary if you pick the hills width by making a free dynamics on the end states you might pick some sigmas that are smaller than what you might use in the middle of the path. This might give a rough free energy profile and definitely more time to converge. A possible solution is to use the adaptive gaussian width scheme. In this scheme you adapt the hills to their fluctuation in time. You find more details in [25] Additionally you also adopt a non spherical shape taking into account variable correlation. So in this scheme you do not have to fix one sigma per variable sigma, but just the time in which you calculate this correlation (another possibility is to calculate it from the compression of phase space but will not be covered here). The input of metadynamics might become something like this

```
t1: TORSION ATOMS=5,7,9,15
t2: TORSION ATOMS=7,9,15,17
p1: PATHMSD REFERENCE=right_path.dat LAMBDA=15100.
p2: PATHMSD REFERENCE=wrong_path.dat  LAMBDA=8244.4
#
# do a metadynamics on the right path, use adaptive hills whose decay time is 125 steps (250 fs)
# and rather standard WT parameters
#
meta: METAD ARG=p1.sss,p1.zzz  ADAPTIVE=DIFF SIGMA=125 HEIGHT=2.4 TEMP=300 BIASFACTOR=12 PACE=125
PRINT ARG=* STRIDE=10 FILE=colvar FMT=%12.8f
```

You can find this example in the directory BIASED_DYNAMICS. After you run for a while it is interesting to have a feeling for the change in shape of the hills. That you can do with

```
pd@plumed:~> gnuplot
gnuplot>  p "<head -400 HILLS" u 2:3:4:5 w xyer
```

that plots the hills in the progress along the path and the distance from the path and add an error bar which reflects the diagonal width of the flexible hills for the first 400 hills (Hey note the funny trick in gnuplot in which you can manipulate the data like in a bash script directly in gnuplot. That's very handy!).

There are a number of things to observe: first that the path explores high distance since the metadynamics is working also in the distance from the path thus accessing the paths that were not explored before, namely the one that goes from the upper left corner of the ramachandran plot and the one that passes through the lower left

corner. So in this way one can also explore other paths. Additionally you can see that the hills are changing size rather considerably. This helps the system to travel faster since at each time you use something that has a nonzero gradient and your forces act on your system in an effective way. Another point is that you can see broad hills covering places which you have not visited in practice. For example you see that hills extend so wide to cover point that have negative progress along the path, which is impossible by using the present definition of the progress along the path. This introduced a problem in calculating the free energy. You actually have to correct for the point that you visited in reality.

You can actually use sum_hills to this purpose in a two-step procedure. First you calculate the negative bias on a given range:

```
pd@plumed:~> plumed sum_hills --hills HILLS --negbias  --outfile negative_bias.dat --bin 100,100 --min -5,-0.0
```

and then calculate the correction. You can use the same hills file for that purpose. The initial transient time should not matter if your simulation is long enough to see many recrossing and, secondly, you should check that the hills heigh in the welltempered are small compared to the beginning.

```
pd@plumed:~> plumed sum_hills --histo HILLS --bin 100,100 --min -5,-0.005 --max 25,0.05 --kt 2.5 --sigma 0.5,0
```

Note that in the correction you should assign a sigma, that is a "trust radius" in which you think that whenever you have a point somewhere, there in the neighborhood you assign a bin (it is done with Gaussian in reality, but this does not matter much). This is the important point that compenstates for the issues you might encounter putting excessive large hills in places that you have not visited. It is nice to have a look to the correction and compare with the hills in the same range.

```
gnuplot> set pm3d
gnuplot> spl "correction.dat" u 1:2:3 w l
gnuplot> set contour
gnuplot> set cntrp lev incremental -20,4.,1000.
gnuplot> set view map
gnuplot> unset clabel
gnuplot> replot
```

You might notice that there are no contour in the unrealistic range, this means that the free energy correction is impossible to calculate since it is too high (see Fig. belfast-2-metadpath-correction-fig).

Now the last thing that one has to do to have the most plausible free energy landscape is to sum both the correction and the negative bias produced. This can be easily done in gnuplot as follows:

```
gnuplot> set pm3d
gnuplot> spl "<paste negative_bias.dat correction.dat " u 1:2:($3+$8) w pm3d
gnuplot> set view map
gnuplot> unset key
gnuplot> set xr [-2:23]
gnuplot> set contour
gnuplot> unset clabel
gnuplot> set cbrange [-140:-30]
gnuplot> set cntrp lev incr -140,6,-30
```

So now we can comment a bit on the free energy surface obtained and note that there is a free energy path that connects the two endpoints and runs very close to zero distance from the path. This means that our input path is actually resemblant of what is really happening in the system. Additionally you can see that there are many comparable routes different from the straight path. Can you make a sense of it just by looking at the free energy on the Ramachandran plot?

10.3 Belfast tutorial: Adaptive variables II

10.3.1 Aims

The aim of this tutorial is to consolidate the material that was covered during Belfast tutorial: Analyzing CVs and Belfast tutorial: Adaptive variables I on analysing trajectories using collective variables and path collective variables.

We will then build on this material by showing how you can use the multidimensional scaling algorithm to automate the process of finding collective variables.

10.3.2 Learning Outcomes

Once this tutorial is completed students will:

- Know how to load colvar data into the GISMO plugin
- Know how to run the multidimensional scaling algorithms on a trajectory
- Be able to explain how we can automate the process of finding collective variables by seeking out an isometry between a high-dimensional and low-dimensional space

10.3.3 Resources

The `tarball` for this project contains the following files:

- trajectory-short.xyz : a (short) trajectory for a 16 residue protein in xyz format. All calculations with plumed driver use this trajectory.
- trajectory-short.pdb : the same trajectory in pdb format, this can be loaded with VMD
- template.pdb : a single frame from the trajectory that can be used in conjuction with the MOLINFO command

10.3.4 Instructions

10.3.4.1 Visualising the trajectory

The aim of this tutorial is to understand the data contained in the trajectory called trajectory-short.pdb. This file contains some frames from a simulation from a 16 residue protein. As a start point then let load this trajectory with vmd and have a look at it. Type the following command into the command line:

```
vmd trajectory-short.pdb
```

Look at it with the various representations that vmd offers. If you at are at the plumed tutorial try typing the letter m on the keyboard - we have made the new cartoon representation will update automatically for each frame of the trajectory - cool huh! What are your impressions about this trajectory based on looking at it with VMD? How many basins in the free energy landscape is this trajectory sampling from? What can we tell from looking at this trajectory that we could perhaps put in a paper?

If your answers to the questions at the end of the above paragraph are I don't know that is good. We can tell very little by just looking at a trajectory. In fact the whole point of today has been to find ways of analyzing trajectories precisely so that we are not put in this position of staring at trajetories mystified!

10.3.4.2 Finding collective variables

Right so lets come up with some CVs to analyse this trajectory with. As some of you may know we can understand the conformation of proteins by looking at the Ramachandran angles. For those of you who don't know here is a Wikkepedia article:

`http://en.wikipedia.org/wiki/Ramachandran_plot`

Our protein has 32 ramachandran angles. We'll come back to that. For the time being pick out a couple that you think might be useful and construct a plumed input to calculate them and print them to a file. You will need to use the TORSION and PRINT commands in order to do this. Once you have created your plumed input use driver to calculate the torsional angles using the following command:

```
plumed driver --plumed plumed.dat --ixyz trajectory.xyz
```

If you have done this correctly you should have an output file containing your torsional angles. We can use vmd+↩ GISMO to visualise the relationship between the ramachandran angles and the atomic configurations. To do this first load the trajectory in VMD:

```
vmd trajectory-short.pdb
```

Then click on Extensions>Analysis>GISMO. A new window should open in this window click on File>Load colvars. You will be asked to select a colvar file. Select the file that was output by the plumed calculation above. Once the file is loaded you should be able to select the labels that you gave to the Ramachandran angles you calculated with plumed. If you do so you will see that this data is plotted in the GISMO window so that you can interact with it and the trajectory.

What can you conclude from this exercise. Do the CV values of the various frames appear in clusters in the plane? Do points in different clusters correspond to structures that look the same or different? Are there similar looking structures clustered together or are they always far apart? What can we conclude about the various basins in the free energy landscape that have been explored in this trajectory? How many are there? Would your estimate be the same if you tried the above estimate with a different pair of ramachandran angles?

10.3.4.3 Dimensionality reduction

What we have done for most of today is seek out a function that takes as input the position of all the atoms in the system - a $3N$ dimensional vector, where N is the number of atoms. This function then outputs a single number - the value of the collective variable - that tells us where in a low dimensional space we should project that configuration. Problems can arise because this collective-variable function is many-to-one. As you have hopefully seen in the previous exercise markedly different confifgurations of the protein can actually have quite similar values of a particular ramachandran angle.

We are going to spend the rest of this session introducing an alternative approach to this bussiness of finding collective variables. In this alternative approach we are going to stop trying to seek out a function that can take any configuration of the atoms (any $3N$-dimensional vector) and find it's low dimensional proejection on the collective variable axis. Instead we are going to take a set of configurations of the atoms (a set of $3N$-dimensional vectors of atom positions) and try to find a sensible set of projections for these configurations. We already touched on this idea earlier when we looked at paths. Our assumption, when we introduced this idea, was that we could find an ordered set of high-dimensional configurations that represented the transtion pathway the system takes as it crossed a barrier and changed between two particularly interesting configurations. Lets say we have a path defined by four reference configurations - this implies that to travel between the configurations at the start and the end of this path you have to pass through configuration 1, then configuration 2, then configuration 3 and then configuration 4. This ordering means that the numbers 1 through 4 constitute sensible projections of these high-dimensional configurations. The numbers 1 through 4 all lie on a single cartesian axis - a low-dimensional space.

The problem when it comes to applying this idea to the data that we have in the trajectory-short trajectory is that we have no information on the "order" of these points. We have not been told that this trajectory represents the transition between two interesting points in phase space and thus we cannot apply the logic of paths. Hence, to seek out a low dimensional representation we are going to try and find a representation of this data we are going to seek out an isometry between the space containing the $3N$-dimensional vectors of atom positions and some lower-dimensional space. This idea is explained in more detail in the following video .

Let's now generate our isometric embedding. You will need to create a plumed input file that contains the following instructions:

```
CLASSICAL_MDS ...
  ATOMS=1-256
  METRIC=OPTIMAL-FAST
  USE_ALL_DATA
  NLOW_DIM=2
  OUTPUT_FILE=rmsd-embed
... CLASSICAL_MDS
```

You should then run this calculation using the following command:

```
plumed driver --ixyz trajectory-short.xyz --plumed plumed.dat
```

This should generate an output file called rmsd-embed. You should now be able to use VMD+GISMO to visualise this output. Do the CV values of the various frames appear in clusters in the plane? Do points in different clusters correspond to structures that look the same or different? Are there similar looking structures clustered together or are they always far apart? What can we conclude about the various basins in the free energy landscape that have been explored in this trajectory? How many are there? Do you think this gives you a fuller picture of the trajectory than the ones you obtained by considering ramachandran angles?

10.3.5 Extensions

As discussed in the previous section this approach to trajectory analysis works by calcalating distances between pairs of atomic configurations. Projections corresponding to these configurations are then generated in the low dimensional space in a way that tries to preserve these pairwise distances. There are, however, an infinite number of ways of calculating the distance between two high-dimensional configurations. In the previous section we used the RMSD distance but you could equally use the DRMSD distance. You could even try calculating a large number of collective variables for each of the high-dimensional points and seeing how much these all changed. You can use these different types of distances with the CLASSICAL_MDS action that was introduced in the previous section. If you have time less at the end of the session read the manual for the CLASSICAL_MDS action and see if you can calculate an MDS projection using an alternative defintion of the distances between configurations. Some suggestions to try in order of increasing difficulty: DRMSD, how much the full set of 32 ramachandran angles change and the change in the contact map

10.3.6 Further Reading

There is a growing community of people using these ideas to analyse trajectory data. Some start points for investigating their work in more detail are:

- http://sketchmap.berlios.de
- http://www.annualreviews.org/doi/abs/10.1146/annurev-physchem-040412-110006

10.4 Belfast tutorial: Umbrella sampling

10.4.1 Aims

In the previous lectures we learned how to compute collective variables (CVs) from atomic positions. We will now learn how one can add a bias potential to enforce the exploration of a particular region of the space. We will also see how it is possible to bias CVs so as to enhance the sampling of events hindered by large free-energy barriers and how to analyze this kind of simulation. This technique is known as "umbrella sampling" and can be used in combination with the weighted-histogram analysis method to compute free-energy landscapes. In this tutorial we will use simple collective variables, but the very same approach can be used with any kind of collective variable.

10.4.2 Summary of theory

10.4.2.1 Biased sampling

A system at temperature T samples conformations from the canonical ensemble:

$$P(q) \propto e^{-\frac{U(q)}{k_B T}}$$

. Here q are the microscopic coordinates and k_B is the Boltzmann constant. Since q is a highly dimensional vector, it is often convenient to analyze it in terms of a few collective variables (see Belfast tutorial: Analyzing CVs , Belfast

tutorial: Adaptive variables I , and Belfast tutorial: Adaptive variables II). The probability distribution for a CV s is

$$P(s) \propto \int dq e^{-\frac{U(q)}{k_B T}} \delta(s - s(q))$$

This probability can be expressed in energy units as a free energy landscape $F(s)$:

$$F(s) = -k_B T \log P(s)$$

.

Now we would like to modify the potential by adding a term that depends on the CV only. That is, instead of using $U(q)$, we use $U(q) + V(s(q))$. There are several reasons why one would like to introduce this potential. One is to avoid that the system samples some un-desired portion of the conformational space. As an example, imagine that you want to study dissociation of a complex of two molecules. If you perform a very long simulation you will be able to see association and dissociation. However, the typical time required for association will depend on the size of the simulation box. It could be thus convenient to limit the exploration to conformations where the distance between the two molecules is lower than a given threshold. This could be done by adding a bias potential on the distance between the two molecules. Another example is the simulation of a portion of a large molecule taken out from its initial context. The fragment alone could be unstable, and one might want to add additional potentials to keep the fragment in place. This could be done by adding a bias potential on some measure of the distance from the experimental structure (e.g. on root-mean-square deviation).

Whatever CV we decide to bias, it is very important to recognize which is the effect of this bias and, if necessary, remove it a posteriori. The biased distribution of the CV will be

$$P'(s) \propto \int dq e^{-\frac{U(q)+V(s(q))}{k_B T}} \delta(s - s(q)) \propto e^{-\frac{V(s(q))}{k_B T}} P(s)$$

and the biased free energy landscape

$$F'(s) = -k_B T \log P'(s) = F(s) + V(s) + C$$

Thus, the effect of a bias potential on the free energy is additive. Also notice the presence of an undetermined constant C. This constant is irrelevant for what concerns free-energy differences and barriers, but will be important later when we will learn the weighted-histogram method. Obviously the last equation can be inverted so as to obtain the original, unbiased free-energy landscape from the biased one just subtracting the bias potential

$$F(s) = F'(s) - V(s) + C$$

Additionally, one might be interested in recovering the distribution of an arbitrary observable. E.g., one could add a bias on the distance between two molecules and be willing to compute the unbiased distribution of some torsional angle. In this case there is no straightforward relationship that can be used, and one has to go back to the relationship between the microscopic probabilities:

$$P(q) \propto P'(q) e^{\frac{V(s(q))}{k_B T}}$$

The consequence of this expression is that one can obtained any kind of unbiased information from a biased simulation just by weighting every sampled conformation with a weight

$$w \propto e^{\frac{V(s(q))}{k_B T}}$$

That is, frames that have been explored in spite of a high (disfavoring) bias potential V will be counted more than frames that has been explored with a less disfavoring bias potential.

10.4.2.2 Umbrella sampling

Often in interesting cases the free-energy landscape has several local minima. If these minima have free-energy differences that are on the order of a few times $k_B T$ they might all be relevant. However, if they are separated by a high saddle point in the free-energy landscape (i.e. a low probability region) than the transition between one and the other will take a lot of time and these minima will correspond to metastable states. The transition between one

minimum and the other could require a time scale which is out of reach for molecular dynamics. In these situations, one could take inspiration from catalysis and try to favor in a controlled manner the conformations corresponding to the transition state.

Imagine that you know since the beginning the shape of the free-energy landscape $F(s)$ as a function of one CV s. If you perform a molecular dynamics simulation using a bias potential which is exactly equal to $-F(s)$, the biased free-energy landscape will be flat and barrierless. This potential acts as an "umbrella" that helps you to safely cross the transition state in spite of its high free energy.

It is however difficult to have an a priori guess of the free-energy landscape. We will see later how adaptive techniques such as metadynamics (Belfast tutorial: Metadynamics) can be used to this aim. Because of this reason, umbrella sampling is often used in a slightly different manner.

Imagine that you do not know the exact height of the free-energy barrier but you have an idea of where the barrier is located. You could try to just favor the sampling of the transition state by adding a harmonic restraint on the CV, e.g. in the form

$$V(s) = \frac{k}{2}(s - s_0)^2$$

. The sampled distribution will be

$$P'(q) \propto P(q) e^{\frac{-k(s(q)-s_0)^2}{2k_B T}}$$

For large values of k, only points close to s_0 will be explored. It is thus clear how one can force the system to explore only a predefined region of the space adding such a restraint. By combining simulations performed with different values of s_0, one could obtain a continuous set of simulations going from one minimum to the other crossing the transition state. In the next section we will see how to combine the information from these simulations.

10.4.2.3 Weighted histogram analysis method

Let's now consider multiple simulations performed with restraints located in different positions. In particular, we will consider the i-th bias potential as V_i. The probability to observe a given value of the collective variable s is:

$$P_i(s) = \frac{P(s) e^{-\frac{V_i(s)}{k_B T}}}{\int ds' P(s') e^{-\frac{V_i(s')}{k_B T}}} = \frac{P(s) e^{-\frac{V_i(s)}{k_B T}}}{Z_i}$$

where

$$Z_i = \sum_q e^{-(U(q)+V_i(q))}$$

The likelyhood for the observation of a sequence of snapshots $q_i(t)$ (where i is the index of the trajectory and t is time) is just the product of the probability of each of the snapshots. We use here the minus-logarithm of the likelihood (so that the product is converted to a sum) that can be written as

$$\mathscr{L} = -\sum_i \int dt \log P_i(s_i(t)) = \sum_i \int dt \left(-\log P(s_i(t)) + \frac{V_i(s_i(t))}{k_B T} + \log Z_i \right)$$

One can then maximize the likelyhood by setting $\frac{\delta \mathscr{L}}{\delta P(\mathbf{s})} = 0$. After some boring algebra the following expression can be obtained

$$0 = \sum_i \int dt \left(-\frac{\delta_{\mathbf{s}_i(t),\mathbf{s}}}{P(\mathbf{s})} + \frac{e^{-\frac{V_i(\mathbf{s})}{k_B T}}}{Z_i} \right)$$

In this equation we aim at finding $P(s)$. However, also the list of normalization factors Z_i is unknown, and they should be found selfconsistently. Thus one can find the solution as

$$P(\mathbf{s}) \propto \frac{N(\mathbf{s})}{\sum_i \int dt \frac{e^{-\frac{V_i(\mathbf{s})}{k_B T}}}{Z_i}}$$

where Z is selfconsistently determined as

$$Z_i \propto \int ds' P(\mathbf{s}') e^{-\frac{V_i(\mathbf{s}')}{k_B T}}$$

These are the WHAM equations that are traditionally solved to derive the unbiased probability $P(s)$ by the combination of multiple restrained simulations. To make a slightly more general implementation, one can compute the weights that should be assigned to each snapshot, that turn out to be:

$$w_i(t) \propto \frac{1}{\sum_j \int dt \frac{e^{-\beta V_j(\mathbf{s}_i(t))}}{Z_j}}$$

The normalization factors can in turn be found from the weights as

$$Z_i \propto \frac{\sum_j \int dt e^{-\beta V_i(\mathbf{s}_j(t))} w_j(t)}{\sum_j \int dt w_j(t)}$$

This allows to straighforwardly compute averages related to other, non-biased degrees of freedom, and it is thus a bit more flexible. It is sufficient to precompute this factors w and use them to weight every single frame in the trajectory.

10.4.3 Learning Outcomes

Once this tutorial is completed students will know how to:

- Setup simulations with restraints.
- Use multiple-restraint umbrella sampling simulations to enhance the transition across a free-energy barrier.
- Analyze the results and compute weighted averages and free-energy profiles.

10.4.4 Resources

The `tarball` for this project contains the following files:

- A gromacs topology (topol.top), configuration (conf.gro), and control file (grompp.mdp). They should not be needed.
- A gromacs binary file (topol.tpr). This is enough for running this system.
- A small C++ program that computes WHAM (wham.cpp) and a script that can be used to feed it (wham.sh)

By working in the directory where the topol.tpr file is stored, one can launch gromacs with the command

```
mdrun_mpi -plumed plumed.dat -nsteps 100000
```

(notice that the -nsteps flag allows the number of steps to be changed).

10.4.5 Instructions

10.4.5.1 The model system

We here use a a model system alanine dipeptide with CHARM27 all atom force field already seen in the previous section.

10.4.5.2 Restrained simulations

The simplest way in which one might influence a CV is by forcing the system to stay close to a chosen value during the simulation. This is achieved with a restraining potential that PLUMED provides via the directive RESTRAINT. In the umbrella sampling method a bias potential is added so as to favor the exploration of some regions of the conformational space and to disfavor the exploration of other regions [33] . A properly chosen bias potential could

allow for example to favor the transition state sampling thus enhancing the transition state for a conformational transition. However, choosing such a potential is not trivial. In a later section we will see how metadynamics can be used to this aim. The simplest way to use umbrella sampling is that to apply harmonic constraints to one or more CVs.

We will now see how to enforce the exploration of a the neighborhood of a selected point the CV space using a RESTRAINT potential.

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Impose an umbrella potential on CV 1 and CV 2
# with a spring constant of 500 kjoule/mol
# at fixed points on the Ramachandran plot
#
restraint-phi: RESTRAINT ARG=phi KAPPA=500 AT=-0.3
restraint-psi: RESTRAINT ARG=psi KAPPA=500 AT=+0.3

# monitor the two variables and the bias potential from the two restraints
PRINT STRIDE=10 ARG=phi,psi,restraint-phi.bias,restraint-psi.bias FILE=COLVAR
```

(see TORSION, RESTRAINT, and PRINT).

The syntax for the command RESTRAINT is rather trivial. The directive is followed by a keyword ARG followed by the label of the CV on which the umbrella potential has to act. The keyword KAPPA determines the hardness of the spring constant and its units are [Energy units]/[Units of the CV]. The additional potential introduced by the UMBRELLA takes the form of a simple Hooke's law:

$$U(s) = \frac{k}{2}(x - x_0)^2$$

.

where x_0 is the value specified following the AT keyword. The choice of AT (x_0) is obviously depending on the specific case. KAPPA (k) is typically chosen not to affect too much the intrinsic fluctuations of the system. A typical recipe is $k \approx \frac{k_B T}{\sigma^2}$, where σ^2 is the variance of the CV in a free simulation). In real applications, one must be careful with values of k larger than $\frac{k_B T}{\sigma^2}$ because they could break down the molecular dynamics integrator.

The CVs as well as the two bias potentials are shown in the COLVAR file. For this specific input the COLVAR file has in first column the time, in the second the value of ϕ, in the third the value of ψ, in the fourth the the additional potential introduced by the restraint on ϕ and in the fifth the additional potential introduced by the restraint on ψ.

It may happen that one wants that a given CV just stays within a given range of values. This is achieved in plumed through the directives UPPER_WALLS and LOWER_WALLS that act on specific collective variables and limit the exploration within given ranges.

10.4.5.3 Reweighting the results

Now consider a simulation performed restraining the variable ϕ:

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
restraint-phi: RESTRAINT ARG=phi KAPPA=10.0 AT=-2
PRINT STRIDE=10 ARG=phi,psi,restraint-phi.bias FILE=COLVAR10
```

and compare the result with the one from a single simulation with no restraint

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
# we use a "dummy" restraint with strength zero here
restraint-phi: RESTRAINT ARG=phi KAPPA=0.0 AT=-2
PRINT STRIDE=10 ARG=phi,psi,restraint-phi.bias FILE=COLVAR0
```

Plot the time dependence of ϕ in the two cases and try to understand the difference.

Now let's try to compute the probability that ψ falls within a given range, say between 1 and 2. This can be done e.g. with this shell script

```
> grep -v \# COLVAR0 | tail -n 80000 |
  awk '{if($3>1 && $3<2)a++; else b++;}END{print a/(a+b)}'
```

Notice that we here considered only the last 80000 frames in the average. Look at the time series for ψ and guess why. Also notice that the script is removing the initial comments. After this trivial preprocessing, the script is just counting how many times the third column (ψ) lies between 1 and 2 and how many times it doesn't. At the end it prints the number of times the variable is between 1 and 2 divided by the total count. The result should be something around 0.40. Now try to do it on trajectories generated with different values of AT. Does the result depend on AT?

We can now try to reweight the result so as to get rid of the bias introduced by the restraint. Since the reweighting factor is just $\exp(\frac{V}{k_BT}$ the script should be modified as

```
> grep -v \# COLVAR10 | tail -n 80000 |
awk '{w=exp($4/2.5); if($3>1 && $3<2)a+=w; else b+=w;}END{print a/(a+b)}'
```

Notice that 2.5 is just k_BT in kj/mol units.

Repeat this calculation for different values of AT. Does the result depend on AT?

10.4.5.4 A free-energy landscape

One can also count the probability of an angle to be in a precise bin. The logarithm of this quantity, in kbT units, is the free-energy associated to that bin. There are several ways to compute histograms, either with PLUMED or with external programs. Here I decided to use awk.

```
grep -v \# COLVAR10 | tail -n 80000 |
awk 'BEGIN{
  min1=-3.14159265358979
  max1=+3.14159265358979
  min2=-3.14159265358979
  max2=+3.14159265358979
  nb1=100;
  nb2=100;
  for(i1=0;i1<nb1;i1++) for(i2=0;i2<nb2;i2++) f[i1,i2]=0.0;
}{
  i1=int(($2-min1)*nb1/(max1-min1));
  i2=int(($3-min2)*nb2/(max2-min2));
# we assume the potential is in the last column, and kbT=2.5 kj/mol
  w=exp($NF/2.5);
  f[i1,i2]+=w;
}
END{
  for(i1=0;i1<nb1;i1++){
  for(i2=0;i2<nb2;i2++) print min1+i1/100.0*(max1-min1), min2+i2/100.0*(max2-min2), -2.5*log(f[i1,i2]);
  print "";
}}' > plotme
```

You can then plot the "plotme" file with

```
gnuplot> set pm3d map
gnuplot> splot "plotme"
```

10.4.5.5 Combining multiple restraints

In the last paragraph you have seen how to reweight simulations done with restraints in different positions to obtain virtually the same result. Let's now see how to combine data from multiple restraint simulations. A possible choice is to download and use the WHAM software here, which is well documented. This is probably the best idea for analyzing a real simulation.

For the sake of learning a bit, we will use a different approach here, namely we will use a short C++ program that implements the weight calculation. Notice that whereas people typically use harmonic restraints in this framework, PLUMED offers a very large variety of bias potentials. For this reason we will keep things as general as possible and use an approach that can be in principle used also to combine simulation with restraint on different variables or with complicated bias potential.

The first step is to generate several simulations with different positions of the restraint, gradually going from say -2 to +2. You can obtain them using e.g. the following script:

```
for AT in -2.0 -1.5 -1.0 -0.5 +0.0 +0.5 +1.0 +1.5 +2.0
do

cat >plumed.dat << EOF
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Impose an umbrella potential on CV 1 and CV 2
# with a spring constant of 500 kjoule/mol
# at fixed points on the Ramachandran plot
#
restraint-phi: RESTRAINT ARG=phi KAPPA=40.0 AT=$AT
# monitor the two variables and the bias potential from the two restraints
PRINT STRIDE=10 ARG=phi,psi,restraint-phi.bias FILE=COLVAR$AT
EOF

mdrun_mpi -plumed plumed.dat -nsteps 100000 -x traj$AT.xtc

done
```

Notice that we are here saving separate trajectories for the separate simulation, as well as separate colvar files. In each simulation the restraint is located in a different position. Have a look at the plot of (phi,psi) for the different simulations to understand what is happening.

An often misunderstood fact about WHAM is that data of the different trajectories can be mixed and it is not necessary to keep track of which restraint was used to produce every single frame. Let's get the concatenated trajectory

```
trjcat -cat -f traj*.xtc -o alltraj.xtc
```

Now we should compute the value of each of the bias potentials on the entire (concatenated) trajectory

```
for AT in -2.0 -1.5 -1.0 -0.5 +0.0 +0.5 +1.0 +1.5 +2.0
do

cat >plumed.dat << EOF
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
restraint-phi: RESTRAINT ARG=phi KAPPA=40.0 AT=$AT

# monitor the two variables and the bias potential from the two restraints
PRINT STRIDE=10 ARG=phi,psi,restraint-phi.bias FILE=ALLCOLVAR$AT
EOF

plumed driver --mf_xtc alltraj.xtc --trajectory-stride=10 --plumed plumed.dat

done
```

It is very important that this script is consistent with the one used to generate the multiple simulations above. Now, single files named ALLCOLVARXX will contain on the fourth column the value of the bias centered in XX computed on the entire concatenated trajectory.

Next step is to compile the C++ program that computes weights self-consistently solving the WHAM equations. This is named wham.cpp and can be compiled with

```
g++ -O3 wham.cpp -o wham.x
```

and can be then used through a wrapper script wham.sh as

```
./wham.sh ALLCOLVAR* > colvar
```

The resulting colvar file will contain 3 columns: time, phi, and psi, plus the weights obtained from WHAM written in logarithmic scale. That is, the file will contain $k_B T \log w$.

Try now to use this file to compute the unbiased free-energy landscape as a function of phi and psi. You can use the script that you used earlier to compute histogram.

10.4.6 Comments

10.4.6.1 How does PLUMED work

The fact that when you add a force on the collective variable PLUMED can force the atoms to do something depends on the fact that the collective variables implemented in PLUMED has analytical derivatives. By biasing the value of a single CV one turns to affect the time evolution of the system itself. Notice that some of the collective variables could be implemented without derivatives (either because the developers were lazy or because the CVs cannot be derived). In this case you might want to have a look at the NUMERICAL_DERIVATIVES option.

10.4.7 Further Reading

Umbrella sampling method is a widely used technique. You can find several resources on the web, e.g.:

- http://en.wikipedia.org/wiki/Umbrella_sampling

10.5 Belfast tutorial: Out of equilibrium dynamics

In plumed you can bring a system in a specific state in a collective variable by means of the MOVINGRESTRAINT directive. This directive is very flexible and allows for a programmed series of draggings and can be used also to sample multiple events within a single simulation. Here I will explain the concepts of it and show some examples

10.5.1 Resources

Here is the tarball with the files referenced in the following .

10.5.2 Steered MD

Steered MD (SMD) is often used to drag the system from an initial configuration to a final one by pulling one or more CVs. Most of time the aim of such simulations is to prepare the system in a particular state or produce nice snapshots for a cool movie. All the CVs present in PLUMED can be used in SMD.

In SMD the Hamiltonian of the system H is modified into H_λ. This new Hamiltonian contains now another new term which now depends on time only via a Harmonic potential centered on a point which moves linear with time

$$\begin{aligned} H_\lambda(X,t) &= H(X)+U_\lambda(X,t) \\ &= H(X)+\frac{k(t)}{2}(s(X)-\lambda(t))^2 \\ &= H(X)+\frac{k(t)}{2}(s(X)-s_0-vt)^2. \end{aligned}$$

This means that if the k is tight enough the system will follow closely the center of the moving harmonic spring. But be careful, if the spring constant is too hard your equations of motion will now keep up since they are tuned to the fastest motion in your system so if you artificially introduce a higher freqeuncy in your system you'll screw up the dynamics. The same is true for the pulling speed v. As a matter of fact I never encountered the case where I had to lower the time step and I could all the time be happy just by making a softer spring constant or a slower steering speed. Generally, integrators of equations of motion like velocity-Verlet are very tolerant. Note that one can also make the spring constant depend on time and this, as we will see later in the examples is particularly useful to prepare your state.

In simulations, it is more convenient to adopt a situation where you specify only the starting point, the final point of cvs and the time in which you want to cover the transition. That's why the plumed input is done in such a way.

For example, let's go back to the alanine dipeptide example encountered in The molecule of the day: alanine dipeptide. Let's say that now we want to steer from C_7eq to C_7ax. If you think, just by dragging along the Φ dihedral angle from a value of -1 to a value 1 should be enough to the state of interest. Additionally, it might be important

to you not to stress the system too much, so you might want first to increase the k first so to lock the system in $\Phi = -1$, then move it gently to $\Phi = 1$ and then release again your spring constant so to end up to an equilibrated and unconstrained state. This you can program in PLUMED like this

```
# set up two variables for Phi and Psi dihedral angles
# drag this
phi: TORSION ATOMS=5,7,9,15
# this is just to monitor that you end up in the interesting state
psi: TORSION ATOMS=7,9,15,17
# the movingrestraint
restraint: ...
        MOVINGRESTRAINT
        ARG=phi
        AT0=-1.0 STEP0=0      KAPPA0=0
        AT1=-1.0 STEP1=2000   KAPPA1=1000
        AT2=1.0  STEP2=4000   KAPPA2=1000
        AT3=1.0  STEP3=6000   KAPPA3=0
...
# monitor the two variables and various restraint outputs
PRINT STRIDE=10 ARG=* FILE=COLVAR
```

Please note the syntax of MOVINGRESTRAINT : You need one (or more) argument(s) and a set of steps denote by ATX, STEPX, KAPPAX where X is a incremental starting from 0 that assign the center and the harness of the spring at step STEPX. What happens in between is a linear interpolation of the AT and KAPPA parameters. If those are identical in two consecutive steps then nothing is happening to that parameter. So if you put the same KAPPA and AT in two different STEPs then this will give you an umbrella potential placed in the same point between the two time intervals defined by STEP. Note that you need to run a bit more than 6000 steps because after this your system has no more restraints so the actual thermalization period starts here.

The COLVAR file produced has the following shape

```
#! FIELDS time phi psi restraint.bias restraint.force2 restraint.phi_cntr restraint.phi_work
#! SET min_phi -pi
#! SET max_phi pi
#! SET min_psi -pi
#! SET max_psi pi
 0.000000 -1.409958 1.246193 0.000000 0.000000 -1.000000 0.000000
 0.020000 -1.432352 1.256545 0.467321 4.673211 -1.000000 0.441499
 0.040000 -1.438652 1.278405 0.962080 19.241592 -1.000000 0.918101
 0.060000 -1.388132 1.283709 1.129846 33.895372 -1.000000 1.344595
 0.080000 -1.360254 1.275045 1.297832 51.913277 -1.000000 1.691475
 ...
```

So we have time, phi, psi and the bias from the moving restraint. Note that at step 0 is zero since we imposed this to start from zero and ramp up in the first 2000 steps up to a value of 2000 kJ/mol/rad^2. It increases immediately since already at step 1 the harmonic potential is going to be increased in bits towards the value of 1000 which is set by KAPPA. The value of restraint.force2 is the squared force (which is a proxy of the force magnitude, despite the direction) on the CV.

$$-\frac{\partial H_\lambda(X,t)}{\partial s} = -(s(X)-s_0-vt)$$

Note that the actual force on an atom of the system involved in a CV is instead

$$\begin{aligned}-\frac{\partial H_\lambda(X,t)}{\partial x_i} &= -\frac{\partial H_\lambda(X,t)}{\partial s}\frac{\partial s}{\partial x_i} \\ &= -(s(X)-s_0-vt)\frac{\partial s}{\partial x_i}\end{aligned}$$

This is important because in CVs that have a derivative that change significantly with space then you might have regions in which no force is exerted while in others you might have an enormous force on it. Typically this is the case of sigmoids that are used in coordination numbers in which, in the tails, they are basically flat as a function of particle positions. Additionally note that this happens on any force-based enhanced-sampling algorithm so keep it

in mind. Very often people miss this aspect and complain either that a CV or a enhanced-sampling method does not work. This is another good reason to use tight spring force so to compensate in the force the lack of derivative from the CV.

The other argument in colvar is restraint.phi_cntr which is the center of the harmonic potential. This is a useful quantity so you may know how close the system is follwing the center of harmonic potential (more on this below). The last parameter is restraint.phi_work. The work is defined as

$$W = \int_0^{t_s} dt \, \frac{\partial H_\lambda(t)}{\partial t}$$

so this is changing only when the Hamiltonian is changing with time. There are two time dependent contributions in this integral: one can come from the fact that $k(t)$ changes with time and another from the fact that the center of the spring potential is changing with time.

$$\begin{aligned} W &= \int_0^{t_s} dt \, \frac{\partial H_\lambda(t)}{\partial t} \\ &= \int_0^{t_s} dt \, \frac{\partial H_\lambda(t)}{\partial \lambda}\frac{\partial \lambda(t)}{\partial t} + \int_0^{t_s} dt \, \frac{\partial H_\lambda(t)}{\partial k}\frac{\partial k}{\partial t} \\ &= \int_0^{t_s} -k(t)(s(X)-\lambda(t))\frac{\partial \lambda(t)}{\partial t}dt + \int_0^{t_s} \frac{(s(X)-\lambda(t))^2}{2}\frac{\partial k}{\partial t}dt \\ &= \int_0^{t_s} -k(t)(s(X)-\lambda(t))v dt + \int_0^{t_s} \frac{(s(X)-\lambda(t))^2}{2}\frac{\partial k}{\partial t}dt \\ &\simeq \sum_i -k(t_i)(s(X(t_i))-\lambda(t_i))\Delta\lambda(t_i) + \sum_i \frac{(s(X(t_i))-\lambda(t_i))^2}{2}\Delta k(t_i) \end{aligned}$$

where we denoted $\Delta\lambda(t_i)$ the difference of the center of the harmonic potential respect to the step before and $\Delta k(t_i)$ is the difference in spring constant respect to the step before. So in the exercised proposed in the first phase you see only the second part of the work since this is the part connected with the spring constant increase. After this phase you see the increase due to the motion of the center and then you later the release of the spring constant.

The work profile as function of time when steering ala dipeptide along the Φ variable.

This you get with gnuplot:

```
pd@plumed:~>gnuplot
gnuplot>  p "COLVAR" u 1:7 w lp
```

Another couple of interesting thing that you can check is

- Is my system finally in the $C7ax$? Plot the two dihedral to have a sense if we are in the right state. You know the target position what should look like, right?
- Is my system moving close to the center of the harmonic potential? This is important and we will see why in a while.

10.5.3 Moving on a more complex path

Very often it is useful to use this movingrestraint to make a fancier schedule by using nice properties of MOVING↩ RESTRAINT. For example you can plan a schedule to drive multiple CVs at the same time in specific point of the phase space and also to stop for a while in specific using a fixed harmonic potential. This can be handy in case of an umbrella sampling run where you might want to explore a 1-dimensional landscape by acquiring some statistics in one point and then moving to the next to acquire more statistics. With MOVINGRESTRAINT you can do it in only one file. To give an example of such capabilities, let's say that we want to move from $C7eq$ vertically toward $\Phi = -1.5; \Psi = -1.3$, stop by for a while (e.g. to acquire a statistics that you might need for an umbrella sampling), then moving toward $\Phi = 1.3; \Psi = -1.3$ which roughly corresponds to $C7ax$.

This can be programmed conveniently with MOVINGRESTRAINT by adopting the following schedule

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
# the movingrestraint
restraint: ...
        MOVINGRESTRAINT
        ARG=phi,psi
        AT0=-1.5,1.3  STEP0=0      KAPPA0=0,0
        AT1=-1.5,1.3  STEP1=2000   KAPPA1=1000,1000
        AT2=-1.5,-1.3 STEP2=4000   KAPPA2=1000,1000
        AT3=-1.5,-1.3 STEP3=4000   KAPPA3=1000,1000
        AT4=1.3,-1.3  STEP4=6000   KAPPA4=1000,1000
        AT5=1.3,-1.3  STEP5=8000   KAPPA5=0,0
...
# monitor the two variables and various restraint outputs
PRINT STRIDE=10 ARG=* FILE=COLVAR
```

Note that by adding two arguments for movingrestraint, now I am allowed to put two values (separated by comma, as usual for multiple values in PLUMED) and correspondingly two KAPPA values. One for each variable. Please note that no space must be used bewtween the arguments! This is a very common fault in writing the inputs.

By plotting the instataneous value of the variables and the value of the center of the harmonic potentials we can inspect the pathways that we make the system walk on the Ramachandran plot. (How to do this? Have a look to the header of COLVAR file to plot the right fields)

.png

Plot of the double steering schedule using MOVINGRESTRAINT

10.5.4 Why work is important?

The work as we have seen is the cumulative change of the hamiltonian in time. So it is connected with the change in energy of your system while you move it around. It has also a more important connection with the free energy via the Jarzynski equation which reads

$$\Delta F = -\beta^{-1} \ln \langle \exp^{-\beta W} \rangle$$

This is important and says that potentially you can calculate the free energy even by driving your system very fast and out of equilibrium between two states of your interest. Unfortunately this in practice not possible since the accurate calculation of the quantity $\langle \exp^{-\beta W} \rangle$ has a huge associated error bar since it involves the average of a noisy quantity (the work) being exponentiated. So, before going wild with SMD, I want to make a small exercise on how tricky that is even for the smallest system.

Now we run, say 30 SMD run and we calculate the free energy difference by using Jarzynski equality and see how this differs from the average. First note that the average $\langle \exp^{-\beta W} \rangle$ is an average over a number of steered MD runs which start from the same value of CV and reach the final value of CV. So it is important to create initially an ensemble of states which are compatible with a given value of CVs. Let's assume that we can do this by using a restrained MD in a point (say at $Phi = -1.5$). In practice the umbrella biases a bit your distribution and the best situation would be to do this with a flat bottom potential and then choosing the snapshot that correspond to the wanted starting value and start from them.

In the directory JARZ/MAKE_ENSEMBLE you find the script to run. After you generate the constrained ensemble this needs to be translated from xtc format to something that GROMACS is able to read in input, typically a more convenient gro file. To do so just to

```
pd@plumed:~> echo 0 | trjconv_mpi-dp-pl -f traj.xtc -s topol.tpr -o all.gro
pd@plumed:~> awk 'BEGIN{i=1}{if($1=="Generated"){outfile=sprintf("start_%d.gro",i);i++}print >>outfile; if(NF=
```

This will generate a set of numbered gro files. Now copy them in the parallel directory MAKE_STEER. There you will find a script (script.sh) where you can set the number of runs that you want to go for. Just try 20 and let it run. Will take short time. The script will also produce a script_rama.gplt that you can use to visualize all the work performed in a single gnuplot session. Just do:

```
pd@plumed:~>gnuplot
gnuplot> load "script_work.gplt"
```

What you see is something like in Fig.

There are a number of interesting fact here. First you see that different starting points may end with very different work values. Not bad, one would say, since Jarzyinsi equality is saying that we can make an average and everything will be ok. So now calculate the average work by using the following bash one-liner:

```
pd@plumed:~> ntest=20; for i in `seq 1 $ntest` ; do tail -1 colvar_$i | awk '{print $7 }' ; done | awk '{g+=ex
```

For my test, what I get is a value of

```
FREE ENERGY ESTIMATE  17.482  STDEV  7.40342
```

and what this is saying is that the only thing that matters is the lowest work that I sampled. This has such an enormous weight over all the other trajectories that will do so that it will be the only other to count, and all the other do not matter much. So it is a kind of a waste of time. Also the standard deviation is rather high and probably it might well be that you obtain a much better result by using a standard umbrella sampling where you can use profitably most of the statistics. Here you waste most of the statistics indeed, since only the lowest work sampled will matter.

Some important point for doing some further exercises:

- How does the work distribution change if you increase the simulation time? Note that you have to increase both the time in the md.mdp file and in the plumed.dat file.
- How the work change if you now use a softer spring constant? And a harder one?
- In particular, what happens when you have softer spring constant, say 10? This does not look like working? Can you guess what is going on there from an analysis of COLVAR files only?
- Have a look of the trajectories in the Ramachandran plot in case of fast simulations and slow simulation. What can you observe? Is there a correlation between steering speed and how often you can go on the low energy path?

10.5.5 Targeted MD

Targeted MD can be seen as a special case of steered MD where the RMSD from a reference structure is used as a collective variable. It can be used for example if one wants to prepare the system so that the coordinates of selected atoms are as close as possible to a target pdb structure.

As an example we can take alanine dipeptide again

```
# set up two variables for Phi and Psi dihedral angles
# these variables will be just monitored to see what happens
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
# creates a CV that measures the RMSD from a reference pdb structure
# the RMSD is measured after OPTIMAL alignment with the target structure
rmsd: RMSD REFERENCE=c7ax.pdb TYPE=OPTIMAL
# the movingrestraint
restraint: ...
        MOVINGRESTRAINT
        ARG=rmsd
        AT0=0.0 STEP0=0      KAPPA0=0
        AT1=0.0 STEP1=5000   KAPPA1=10000
...
# monitor the two variables and various restraint outputs
PRINT STRIDE=10 ARG=* FILE=COLVAR
```

(see TORSION, RMSD, MOVINGRESTRAINT, and PRINT).

Note that RMSD should be provided a reference structure in pdb format and can contain part of the system but the second column (the index) must reflect that of the full pdb so that PLUMED knows specifically which atom to drag where. The MOVINGRESTRAINT bias potential here acts on the rmsd, and the other two variables (phi and psi) are untouched. Notice that whereas the force from the restraint should be applied at every step (thus rmsd is

computed at every step) the two torsions are computed only every 10 steps. PLUMED automatically detect which variables are used at every step, leading to better performance when complicated and computationally expensive variables are monitored - this is not the case here, since the two torsions are very fast to compute. Note that here the work always increase with time and never gets lower which is somewhat surprising if you tink that we are moving in another metastable state. One would expect this to bend and give a signal of approaching a minimum like before. Nevertheless consider what you we are doing: we are constraining the system in one specific conformation and this is completely unnatural for a system at 300 kelvin so, even for this small system adopting a specific conformation in which all the heavy atoms are in a precise position is rather unrealistic. This means that this state is an high free energy state.

10.6 Belfast tutorial: Metadynamics

10.6.1 Aims

The aim of this tutorial is to introduce the users to running a metadynamics simulation with PLUMED. We will set up a simple simulation of alanine dipeptide in vacuum, analyze the output, and estimate free energies from the simulation. We will also learn how to run a well-tempered metadynamics simulation and detect issues related to a bad choice of collective variables.

10.6.2 Summary of theory

In metadynamics, an external history-dependent bias potential is constructed in the space of a few selected degrees of freedom $\vec{s}(q)$, generally called collective variables (CVs) [22]. This potential is built as a sum of Gaussians deposited along the trajectory in the CVs space:

$$V(\vec{s},t) = \sum_{k\tau<t} W(k\tau) \exp\left(-\sum_{i=1}^{d} \frac{(s_i - s_i(q(k\tau)))^2}{2\sigma_i^2}\right).$$

where τ is the Gaussian deposition stride, σ_i the width of the Gaussian for the ith CV, and $W(k\tau)$ the height of the Gaussian. The effect of the metadynamics bias potential is to push the system away from local minima into visiting new regions of the phase space. Furthermore, in the long time limit, the bias potential converges to minus the free energy as a function of the CVs:

$$V(\vec{s},t\to\infty) = -F(\vec{s}) + C.$$

In standard metadynamics, Gaussians of constant height are added for the entire course of a simulation. As a result, the system is eventually pushed to explore high free-energy regions and the estimate of the free energy calculated from the bias potential oscillates around the real value. In well-tempered metadynamics [24], the height of the Gaussian is decreased with simulation time according to:

$$W(k\tau) = W_0 \exp\left(-\frac{V(\vec{s}(q(k\tau)),k\tau)}{k_B \Delta T}\right),$$

where W_0 is an initial Gaussian height, ΔT an input parameter with the dimension of a temperature, and k_B the Boltzmann constant. With this rescaling of the Gaussian height, the bias potential smoothly converges in the long time limit, but it does not fully compensate the underlying free energy:

$$V(\vec{s},t\to\infty) = -\frac{\Delta T}{T+\Delta T} F(\vec{s}) + C.$$

where T is the temperature of the system. In the long time limit, the CVs thus sample an ensemble at a temperature $T+\Delta T$ which is higher than the system temperature T. The parameter ΔT can be chosen to regulate the extent of free-energy exploration: $\Delta T = 0$ corresponds to standard molecular dynamics, $\Delta T \to \infty$ to standard metadynamics.

In well-tempered metadynamics literature and in PLUMED, you will often encounter the term "biasfactor" which is the ratio between the temperature of the CVs ($T+\Delta T$) and the system temperature (T):

$$\gamma = \frac{T+\Delta T}{T}.$$

The biasfactor should thus be carefully chosen in order for the relevant free-energy barriers to be crossed efficiently in the time scale of the simulation.

Additional information can be found in the several review papers on metadynamics [34] [35] [36].

10.6.3 Learning Outcomes

Once this tutorial is completed students will know how to:

- run a metadynamics simulation using PLUMED
- analyze the output of the simulation
- restart a metadynamics simulation
- calculate free energies from a metadynamics simulation
- run a well-tempered metadynamics simulation using PLUMED
- detect issues with the choice of the collective variables

10.6.4 Resources

The `tarball` for this project contains the following directories:

- TOPO: it contains the gromacs topology and configuration files to simulate alanine dipeptide in vacuum
- Exercise_1: run a metadynamics simulation with 2 CVs, dihedrals phi and psi, and analyze the output
- Exercise_2: restart a metadynamics simulation
- Exercise_3: calculate free energies from a metadynamics simulation and monitor convergence
- Exercise_4: run a well-tempered metadynamics simulation with 2 CVs, dihedrals phi and psi
- Exercise_5: run a well-tempered metadynamics simulation with 1 CV, dihedral psi

10.6.5 Instructions

10.6.5.1 The model system

Here we use as model system alanine dipeptide in vacuum with AMBER99SB-ILDN all-atom force field.

10.6.5.2 Exercise 1. Setup and run a metadynamics simulation

In this exercise, we will run a metadynamics simulation on alanine dipeptide in vacuum, using as CVs the two backbone dihedral angles phi and psi. In order to run this simulation we need to prepare the PLUMED input file (plumed.dat) as follows.

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate metadynamics in phi and psi
# depositing a Gaussian every 500 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 0.35 rad for both CVs.
#
metad: METAD ARG=phi,psi PACE=500 HEIGHT=1.2 SIGMA=0.35,0.35 FILE=HILLS

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

(see TORSION, METAD, and PRINT).

The syntax for the command METAD is simple. The directive is followed by a keyword ARG followed by the labels of the CVs on which the metadynamics potential will act. The keyword PACE determines the stride of Gaussian deposition in number of time steps, while the keyword HEIGHT specifies the height of the Gaussian in kJoule/mol. For each CVs, one has to specified the width of the Gaussian by using the keyword SIGMA. Gaussian will be written to the file indicated by the keyword FILE.

Once the PLUMED input file is prepared, one has to run Gromacs with the option to activate PLUMED and read the input file:

```
mdrun_mpi -plumed
```

During the metadynamics simulation, PLUMED will create two files, named COLVAR and HILLS. The COLVAR file contains all the information specified by the PRINT command, in this case the value of the CVs every 10 steps of simulation, along with the current value of the metadynamics bias potential. The HILLS file contains a list of the Gaussians deposited along the simulation. If we give a look at the header of this file, we can find relevant information about its content:

```
#! FIELDS time phi psi sigma_phi sigma_psi height biasf
#! SET multivariate false
#! SET min_phi -pi
#! SET max_phi pi
#! SET min_psi -pi
#! SET max_psi pi
```

The line starting with FIELDS tells us what is displayed in the various columns of the HILLS file: the time of the simulation, the value of phi and psi, the width of the Gaussian in phi and psi, the height of the Gaussian, and the biasfactor. This quantity is relevant only for well-tempered metadynamics simulation (see Exercise 4. Setup and run a well-tempered metadynamics simulation, part I) and it is equal to 1 in standard metadynamics simulations. We will use the HILLS file later to calculate free-energies from the metadynamics simulation and assess its convergence. For the moment, we can plot the behavior of the CVs during the simulation.

By inspecting Figure belfast-6-metad-fig, we can see that the system is initialized in one of the two metastable states of alanine dipeptide. After a while (t=0.3 ns), the system is pushed by the metadynamics bias potential to visit the other local minimum. As the simulation continues, the bias potential fills the underlying free-energy landscape, and the system is able to diffuse in the entire phase space.

If we use the PLUMED input file described above, the expense of a metadynamics simulation increases with the length of the simulation as one has to evaluate the values of a larger and larger number of Gaussians at every step. To avoid this issue you can store the bias on a grid. In order to use grids, we have to add some additional information to the line of the METAD directive, as follows.

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate metadynamics in phi and psi
# depositing a Gaussian every 500 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 0.35 rad for both CVs.
```

```
# The bias potential will be stored on a grid
# with bin size equal to 0.1 rad for both CVs.
# The boundaries of the grid are -pi and pi, for both CVs.
#
METAD ...
LABEL=metad
ARG=phi,psi
PACE=500
HEIGHT=1.2
SIGMA=0.35,0.35
FILE=HILLS
GRID_MIN=-pi,-pi
GRID_MAX=pi,pi
GRID_SPACING=0.1,0.1
... METAD

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

The bias potential will be stored on a grid, whose boundaries are specified by the keywords GRID_MIN and GR↩ ID_MAX. Notice that you should provide either the number of bins for every collective variable (GRID_BIN) or the desired grid spacing (GRID_SPACING). In case you provide both PLUMED will use the most conservative choice (highest number of bins) for each dimension. In case you do not provide any information about bin size (neither GRID_BIN nor GRID_SPACING) and if Gaussian width is fixed PLUMED will use 1/5 of the Gaussian width as grid spacing. This default choice should be reasonable for most applications.

10.6.5.3 Exercise 2. Restart a metadynamics simulation

If we try to run again a metadynamics simulation using the script above in a directory where a COLVAR and HILLS files are already present, PLUMED will create a backup copy of the old files, and run a new simulation. Instead, if we want to restart a previous simulation, we have to add the keyword RESTART to the PLUMED input file (plumed.dat), as follows.

```
# restart previous simulation
RESTART

# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate metadynamics in phi and psi
# depositing a Gaussian every 500 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 0.35 rad for both CVs.
#
metad: METAD ARG=phi,psi PACE=500 HEIGHT=1.2 SIGMA=0.35,0.35 FILE=HILLS

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

(see RESTART, TORSION, METAD, and PRINT).

In this way, PLUMED will read the old Gaussians from the HILLS file and append the new information to both COLVAR and HILLS files.

10.6.5.4 Exercise 3. Calculate free-energies and monitor convergence

One can estimate the free energy as a function of the metadynamics CVs directly from the metadynamics bias potential. In order to do so, the utility sum_hills should be used to sum the Gaussians deposited during the simulation and stored in the HILLS file.

To calculate the two-dimensional free energy as a function of phi and psi, it is sufficient to use the following command line:

```
plumed sum_hills --hills HILLS
```

The command above generates a file called fes.dat in which the free-energy surface as function of phi and psi is calculated on a regular grid. One can modify the default name for the free energy file, as well as the boundaries and bin size of the grid, by using the following options of sum_hills :

```
--outfile - specify the outputfile for sumhills
--min - the lower bounds for the grid
--max - the upper bounds for the grid
--bin - the number of bins for the grid
--spacing - grid spacing, alternative to the number of bins
```

It is also possible to calculate one-dimensional free energies from the two-dimensional metadynamics simulation. For example, if one is interested in the free energy as a function of the phi dihedral alone, the following command line should be used:

```
plumed sum_hills --hills HILLS --idw phi --kt 2.5
```

The result should look like this:

To assess the convergence of a metadynamics simulation, one can calculate the estimate of the free energy as a function of simulation time. At convergence, the reconstructed profiles should be similar, apart from a constant offset. The option –stride should be used to give an estimate of the free energy every N Gaussians deposited, and the option –mintozero can be used to align the profiles by setting the global minimum to zero. If we use the following command line:

```
plumed sum_hills --hills HILLS --idw phi --kt 2.5 --stride 500 --mintozero
```

one free energy is calculated every 500 Gaussians deposited, and the global minimum is set to zero in all profiles. The resulting plot should look like the following:

To assess the convergence of the simulation more quantitatively, we can calculate the free-energy difference between the two local minima in the one-dimensional free energy along phi as a function of simulation time. We can use the bash script analyze_FES.sh to integrate the multiple free-energy profiles in the two basins defined by the following intervals in phi space: basin A, -3<phi<-1, basin B, 0.5<phi<1.5.

```
./analyze_FES.sh NFES -3.0 -1.0 0.5 1.5 KBT
```

where NFES is the number of profiles (free-energy estimates at different times of the simulation) generated by the option –stride of sum_hills, and KBT is the temperature in energy units (in this case KBT=2.5).

This analysis, along with the observation of the diffusive behavior in the CVs space, suggest that the simulation is converged.

10.6.5.5 Exercise 4. Setup and run a well-tempered metadynamics simulation, part I

In this exercise, we will run a well-tempered metadynamics simulation on alanine dipeptide in vacuum, using as CVs the two backbone dihedral angles phi and psi. To activate well-tempered metadynamics, we need to add two keywords to the line of METAD, which specifies the biasfactor and temperature of the simulation. For the first example, we will try a biasfactor equal to 6. Here how the PLUMED input file (plumed.dat) should look like:

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate metadynamics in phi and psi
# depositing a Gaussian every 500 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 0.35 rad for both CVs.
# Well-tempered metadynamics is activated,
# and the biasfactor is set to 6.0
#
metad: METAD ARG=phi,psi PACE=500 HEIGHT=1.2 SIGMA=0.35,0.35 FILE=HILLS BIASFACTOR=6.0 TEMP=300.0

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

(see TORSION, METAD, and PRINT).

After running the simulation using the instruction described above, we can have a look at the HILLS file. At variance with standard metadynamics, the last two columns of the HILLS file report more useful information. The last column contains the value of the biasfactor used, while the last but one the height of the Gaussian, which is rescaled during the simulation following the well-tempered recipe.

```
#! FIELDS time phi psi sigma_phi sigma_psi height biasf
#! SET multivariate false
#! SET min_phi -pi
#! SET max_phi pi
#! SET min_psi -pi
#! SET max_psi pi
      1.0000     -1.3100    0.0525          0.35            0.35      1.4400      6
      2.0000     -1.4054    1.9742          0.35            0.35      1.4400      6
      3.0000     -1.9997    2.5177          0.35            0.35      1.4302      6
      4.0000     -2.2256    2.1929          0.35            0.35      1.3622      6
```

If we carefully look at the height column, we will notice that in the beginning the value reported is higher than the initial height specified in the input file, which should be 1.2 kJoule/mol. In fact, this column reports the height of the Gaussian rescaled by the pre-factor that in well-tempered metadynamics relates the bias potential to the free energy. In this way, when we will use sum_hills, the sum of the Gaussians deposited will directly provide the free-energy, without further rescaling needed.

We can plot the time evolution of the CVs along with the height of the Gaussian.

The system is initialized in one of the local minimum where it starts accumulating bias. As the simulation progresses and the bias added grows, the Gaussian height is progressively reduced. After a while (t=0.8 ns), the system is able to escape the local minimum and explore a new region of the phase space. As soon as this happens, the Gaussian height is restored to the initial value and starts to decrease again. In the long time, the Gaussian height becomes smaller and smaller while the system diffuses in the entire CVs space.

We can now try a different biasfactor and see the effect on the simulation. If we choose a biasfactor equal to 1.5, we can notice a faster decrease of the Gaussian height with simulation time, as expected by the well-tempered recipe. We will also conclude from the plot below that this biasfactor is not large enough to allow for the system to escape from the initial local minimum in the time scale of this simulation.

Following the procedure described for standard metadynamics in the previous example, we can estimate the free energy as a function of time and monitor the convergence of the simulations using the analyze_FES.sh script. We will do this for the simulation in which the biasfactor was set to 6.0. In this case we will notice that the oscillations observed in standard metadynamics are here damped, and the bias potential converges more smoothly to the underlying free-energy landscape, provided that the biasfactor is sufficiently high for the free-energy barriers of the system under study to be crossed.

10.6.5.6 Exercise 5. Setup and run a well-tempered metadynamics simulation, part II

In this exercise, we will study the effect of neglecting a relavant degree of freedom in the choice of metadynamics CVs. We are going to run a well-tempered metadynamics simulation with the psi dihedral alone as CV, using the following PLUMED input file (plumed.dat):

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate metadynamics in psi
# depositing a Gaussian every 500 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 0.35 rad.
# Well-tempered metadynamics is activated,
# and the biasfactor is set to 10.0
#
metad: METAD ARG=psi PACE=500 HEIGHT=1.2 SIGMA=0.35 FILE=HILLS BIASFACTOR=10.0 TEMP=300.0

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

(see TORSION, METAD, and PRINT).

Let's look at the HILLS file, in particular at the time serie of the CV psi and of the Gaussian height.

From this plot, we observe a nice diffusive behavior of the CV psi when the Gaussian height is already quite small. This happens until t=3 ns, when the CV seems to be stuck for a while in a small region of the CV space. This behavior is typical of a situation in which a slow variable is not included in the set of CV. When something happens in this hidden degree of freedom, the biased CVs typically cannot access anymore regions of the phase space previously visited. To understand this behavior, we need to visualize the time evolution of both phi and psi stored in the COLVAR file.

It is clear from the plot above that what happened around t=3 ns is a jump of the neglected, slow degree of freedom phi from one free-energy basin to another. The dynamics of phi is not biased by any potential, so we need to equilibrate this degree of freedom, i.e. to observe multiple transitions from the two basins, before declaring convergence of our simulation. Or alternatively we can add phi to the set of CVs. This example demonstrates how to declare convergence of a well-tempered metadynamics simulation it is necessary but not sufficient to observe: 1) Gaussians with very small height, 2) a diffusive behavior in the CV space (as in the first 3 ns of this example). What we should do is repeating the simulation multiple times starting from different initial conformations. If in all simulations, we observe a diffusive behavior in the biased CV when the Gaussian height is very small, and we obtain very similar free-energy surfaces, then we can be quite confident that our simulations are converged to the right value. If this is not the case, a manual inspection of the runs can help us identifying the missing slow degrees of freedom to add to the set of biased CVs.

10.7 Belfast tutorial: Replica exchange I

10.7.1 Aims

The aim of this tutorial is to introduce the users to running a parallel tempering (PT) simulation using PLUMED. We will set up a simple simulation of alanine dipeptide in vacuum, analyze the output, calculate free energies from the simulation, and detect problems. We will also learn how to run a combined PT-metadynamics simulation (PTMetaD) and introduce the users to the Well-Tempered Ensemble (WTE).

10.7.2 Summary of theory

In Replica Exchange Methods [37] (REM), sampling is accelerated by modifying the original Hamiltonian of the system. This goal is achieved by simulating N non-interacting replicas of the system, each evolving in parallel according to a different Hamiltonian. At fixed intervals, an exchange of configurations between two replicas is attempted. One popular case of REM is PT, in which replicas are simulated using the same potential energy function, but different temperatures. By accessing high temperatures, replicas are prevented from being trapped in local minima. In PT, exchanges are usually attempted between adjacent temperatures with the following acceptance probability:

$$p(i \to j) = min\{1, \Delta_{i,j}^{PT}\},$$

with

$$\Delta_{i,j}^{PT} = \left(\frac{1}{k_B T_i} - \frac{1}{k_B T_j}\right)(U(R_i) - U(R_j)),$$

where R_i and R_j are the configurations at temperature T_i and T_j, respectively. The equation above suggests that the acceptance probability is ultimately determined by the overlap between the energy distributions of two replicas. The efficiency of the algorithm depends on the benefits provided by sampling at high-temperature. Therefore, an efficient diffusion in temperature space is required and configurational sampling is still limited by entropic barriers. Finally, PT scales poorly with system size. In fact, a sufficient overlap between the potential energy distributions of neighboring temperatures is required in order to obtain a significant diffusion in temperature. Therefore, the number of temperatures needed to cover a given temperature range scales as the square root of the number of degrees of freedom, making this approach prohibitively expensive for large systems.

PT can be easily combined with metadynamics [38]. In the resulting PTMetaD algorithm (16), N replicas performed in parallel a metadynamics simulation at different temperatures, using the same set of CVs. The PT acceptance probability must be modified in order to account for the presence of a bias potential:

$$\Delta_{i,j}^{PTMetaD} = \Delta_{i,j}^{PT} + \frac{1}{k_B T_i}\left[V_G^i(s(R_i),t) - V_G^i(s(R_j),t)\right] + \frac{1}{k_B T_j}\left[V_G^j(s(R_j),t) - V_G^j(s(R_i),t)\right],$$

where V_G^i and V_G^j are the bias potentials acting on the i-th and j-th replicas, respectively.

PTMetaD is particularly effective because it compensates for some of the weaknesses of each method alone. The effect of neglecting a slow degree of freedom in the choice of the metadynamics CVs is alleviated by PT, which allows the system to cross moderately high free-energy barriers on all degrees of freedom. On the other hand, the metadynamics bias potential allows crossing higher barriers on a few selected CVs, in such a way that the sampling efficiency of PTMetaD is greater than that of PT alone.

PTMetaD still suffers from the poor scaling of computational resources with system size. This issue may be circumvented by including the potential energy of the system among the set of well-tempered metadynamics CVs. The well-tempered metadynamics bias leads to the sampling of a well-defined distribution called Well-Tempered Ensemble (WTE) [12]. In this ensemble, the average energy remains close to the canonical value but its fluctuations are enhanced in a tunable way, thus improving sampling. In the so-called PTMetaD-WTE scheme [39], each replica diffusion in temperature space is enhanced by the increased energy fluctuations at all temperatures.

10.7.3 Learning Outcomes

Once this tutorial is completed students will know how to:

- run a PT simulation
- analyze the output of the PT simulation and detect problems
- run a PTMetaD simulation
- run a PT and PTMetaD in the WTE

10.7.4 Resources

The `tarball` for this project contains the following directories:

- Exercise_1: run a PT simulation using 2 replicas and analyze the output
- Exercise_2: run a PT simulation using 4 replicas and analyze the output
- Exercise_3: run a PTMetaD simulation
- Exercise_4: run a PT, PT-WTE and PTMetaD-WTE simulations

Each directory contains a TOPO subdirectory where topology and configuration files for Gromacs are stored.

10.7.5 Instructions

10.7.5.1 The model system

Here we use as model systems alanine dipeptide in vacuum and water with AMBER99SB-ILDN all-atom force field.

10.7.5.2 Exercise 1. Setup and run a PT simulation, part I

In this exercise, we will run a PT simulation of alanine dipeptide in vacuum, using only two replicas, one at 300K, the other at 305K. During this simulation, we will monitor the time evolution of the two dihedral angles phi and psi. In order to do that, we need the following PLUMED input file (plumed.dat).

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17

# monitor the two variables
PRINT STRIDE=10 ARG=phi,psi FILE=COLVAR
```

(see TORSION, and PRINT).

To submit this simulation with Gromacs, we need the following command line.

```
mpirun -np 2 mdrun_mpi -s TOPO/topol -plumed -multi 2 -replex 100
```

This command will execute two MPI processess in parallel, using the topology files topol0.tpr and topol1.tpr stored in the TOPO subdirectory. These two binary files have been created using the usual Gromacs procedure (see Gromacs manual for further details) and setting the temperature of the two simulations at 300K and 305K in the configuration files. An exchange between the configurations of the two simulations will be attempted every 100 steps.

When Gromacs is executed using the -multi option and PLUMED is activated, the output files produced by PLUMED will be renamed and a suffix indicating the replica id will be appended. We will start inspecting the output file COLVAR.0, which reports the time evolution of the CVs at 300K.

The plot above suggests that during the PT simulation the system is capable to access both the relevant basins in the free-energy surface. This seems to suggest that our simulation is converged. We can use the COLVAR.0 and COLVAR.1 along with the tool sum_hills to estimate the free energy as a function of the CV phi. We will do this a function of simulation time to assess convergence more quantitatively, using the following command line:

```
plumed sum_hills --histo COLVAR.0 --idw phi --sigma 0.2 --kt 2.5 --outhisto fes_ --stride 1000
```

As we did in our previous tutorial, we can now use the script analyze_FES.sh to calculate the free-energy difference between basin A (-3.0<phi<-1.0) and basin B (0.5<phi<1.5), as a function of simulation time.

The estimate of the free-energy difference between these two basins seems to be converged. This consideration, along with the observation that the system is exploring all the relevant free-energy basins, might lead us to declare convergence and to state that the difference in free energy between basin A and B is roughly 0 kJoule/mol. Unfortunately, in doing so we would make a big mistake.

In PT simulations we have to be a little bit more careful, and examine the time evolution of each replica diffusing in temperature space, before concluding that our simulation is converged. In order to do that, we need to reconstruct the continuos trajectories of the replicas in temperature from the two files (typically traj0.trr and traj1.trr) which contain the discontinuous trajectories of the system at 300K and 305K. To demux the trajectories, we need to use the following command line:

```
demux.pl md0.log
```

which will create two files, called replica_temp.xvg and replica_index.xvg. We can plot the first file, which reports the temperature index of each configuration at a given time of the simulation. This file is extremely useful, because it allows us monitoring the replicas diffusion in temperature. As we discussed in Summary of theory, in order for the PT algorithm to be effective, we need an efficient diffusion of the replicas in temperature space. In this case, both replicas are rapidly accessing the highest temperature, so there seems not to be any problem on this side.

We can use the second file to reconstruct the continuous trajectories of each replica in temperature:

```
trjcat_mpi -f traj0.trr traj1.trr -demux replica_index.xvg
```

and the following PLUMED input file (plumed_demux.dat) to recalculate the value of the CVs on the demuxed trajectories, typically called 0_trajout.xtc and 1_trajout.xtc.

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17

# monitor the two variables
PRINT STRIDE=1 ARG=phi,psi FILE=COLVAR_DEMUX
```

(see TORSION, and PRINT).

For the analysis of the demuxed trajectories, we can use the -rerun option of Gromacs, as follows:

```
# rerun Gromacs on replica 0 trajectory
mdrun_mpi -s TOPO/topol0.tpr -plumed plumed_demux.dat -rerun 0_trajout.xtc
# rename the output
mv COLVAR_DEMUX COLVAR_DEMUX.0
# rerun Gromacs on replica 1 trajectory
mdrun_mpi -s TOPO/topol1.tpr -plumed plumed_demux.dat -rerun 1_trajout.xtc
# rename the output
mv COLVAR_DEMUX COLVAR_DEMUX.1
```

We can now plot the time evolution of the two CVs in the two demuxed trajectories.

This plot shows clearly that each replica is sampling only one of the two basins of the free-energy landscape, and it is never able to cross the high barrier that separates them. This means that what we considered an exhaustive exploration of the free energy landscape at 300K (Figure belfast-7-pt-fig), was instead caused by an efficient exchange of configurations between replicas that were trapped in different free-energy basins. The results of the present simulation were then influenced by the initial conformations of the two replicas. If we had initialized both of them in the same basin, we would have never observed "transitions" to the other basin at 300K.

To declare convergence of a PT simulation, it is thus mandatory to examine the behavior of the replicas diffusing in temperature and check that these are exploring all the relevant basins. Another good practice is repeating the PT simulation starting from different initial conformations, and check that the results are consistent.

10.7.5.3 Exercise 2. Setup and run a PT simulation, part II

We will now repeat the previous exercise, but with a different setup. The problem with the previous exercise was that replicas were never able to interconvert between the two metastable states of alanine dipeptide in vacuum. The reason was that the highest temperature used (305K) was too low to accelerate barrier crossing in the time scale of the simulation. We will now use 4 replicas at the following temperatures: 300K, 400K, 600K, and 1000K.

We can use the same PLUMED input file described above (plumed.dat), and execute Gromacs using the following command line:

```
mpirun -np 4 mdrun_mpi -s TOPO/topol -plumed -multi 4 -replex 100
```

At the end of the simulation, we first monitor the diffusion in temperature space of each replica. We need to create the replica_temp.xvg and replica_index.xvg:

```
demux.pl md0.log
```

and plot the content of replica_temp.xvg. Here how it should look for Replica 0:

From this analysis, we can conclude that replicas are diffusing effectively in temperature. Now, we need to monitor the space sampled by each replica while diffusing in temperature space and verify that they are interconverting between the different basins of the free-energy landscape. The demux is carried out as in the previous example:

```
trjcat_mpi -f traj0.trr traj1.trr traj2.trr traj3.trr -demux replica_index.xvg
```

and so is the analysis of the demuxed trajectories using the -rerun option of Gromacs and the plumed_demux.dat input file. Here is the space sampled by two of the four replicas:

It is clear that in this case replicas are able to interconvert between the two metastable states, while efficiently diffusing in temperature. We can then calculate the free-energy difference between basin A and B as a function of simulation time at 300K:

and conclude that in this case the PT simulation is converged.

10.7.5.4 Exercise 3. Setup and run a PTMetaD simulation

In this exercise we will learn how to combine PT with metadynamics. We will use the setup of the previous exercise, and run a PT simulations with 4 replicas at the following temperatures: 300K, 400K, 600K, and 1000K. Each simulation will perform a well-tempered metadynamics calculation, using the dihedral psi alone as CV and a biasfactor equal to 10 (see Exercise 5. Setup and run a well-tempered metadynamics simulation, part II).

Previously, we prepared a single PLUMED input file to run a PT simulation. This was enough, since in that case the same task was performed at all temperatures. Here instead we need to have a slightly different PLUMED input file for each simulation, since we need to use the keyword TEMP to specify the temperature on the line of the METAD directory. We will thus prepare 4 input files, called plumed.dat.0, plumed.dat.1, plumed.dat.2, and plumed.dat.3, with a different value for the keyword TEMP. Here how plumed.dat.3 should look like:

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate metadynamics in psi
# depositing a Gaussian every 500 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 0.35 rad.
# Well-tempered metadynamics is activated,
# and the biasfactor is set to 10.0
#
metad: METAD ARG=psi PACE=500 HEIGHT=1.2 SIGMA=0.35 FILE=HILLS BIASFACTOR=10.0 TEMP=1000.0

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

(see TORSION, METAD, and PRINT).

The PTMetaD simulation is executed in the same way as the PT:

```
mpirun -np 4 mdrun_mpi -s TOPO/topol -plumed -multi 4 -replex 100
```

and it will produce one COLVAR and HILLS file per temperature (COLVAR.0, HILLS.0, ...). The analysis of the results requires what we have learned in the previous exercise for the PT case (analysis of the replica diffusion in temperature and demuxing of each replica trajectory), and the post-processing of a well-tempered metadynamics simulation (FES calculation using sum_hills and convergence analysis).

Since in the previous tutorial we performed the same well-tempered metadynamics simulation without the use of PT (see Exercise 5. Setup and run a well-tempered metadynamics simulation, part II), here we can focus on the differences with the PTMetaD simulation. Let's compare the behavior of the biased variable psi in the two simulations:

In well-tempered metadynamics (left panel), the biased variable psi looked stuck early in the simulation (t=3 ns). The reason was the transition of the other hidden degree of freedom phi from one free-energy basin to the other. In the PTMetaD case (right panel), this seems not to happen. To better appreciate the difference, we can plot the time evolution of the hidden degree of freedom phi in the two cases.

Thanks to the excursions at high temperature, in the PTMetaD simulation the transition of the CV phi between the two basins is accelerate. As a result, the convergence of the reconstructed free energy in psi will be accelerated. This simple exercise demonstrates how PTMetaD can be used to cure a bad choice of metadynamics CVs and the neglecting of slow degrees of freedom.

10.7.5.5 Exercise 4. The Well-Tempered Ensemble

In this exercise we will learn how to run a PT-WTE and PTMetaD-WTE simulations of alanine dipeptide in water. We will start by running a short PT simulation using 4 replicas in the temperature range between 300K and 400K. We will use a geometric distribution of temperatures, which is valid under the assumption that the specific heat of the system is constant across temperatures. Replicas will thus be simulated at T=300, 330.2, 363.4, and 400K. In this simulation, we will just monitor the two dihedral angles and the total energy of the system, by preparing the following PLUMED input file (plumed_PT.dat):

```
# set up three variables for Phi and Psi dihedral angles
# and total energy
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
ene: ENERGY

# monitor the three variables
PRINT STRIDE=10 ARG=phi,psi,ene FILE=COLVAR_PT
```

(see TORSION, ENERGY, and PRINT).

As usual, the simulation is run for 400ps using the following command:

```
 mpirun -np 4 mdrun_mpi -s TOPO/topol -plumed plumed_PT.dat -multi 4 -replex 100
```

At the end of the run, we want to analyze the acceptance rate between exchanges. This quantity is reported at the end of the Gromacs output file, typically called md.log, and it can be extracted using the following bash command line:

```
grep -A2 "Repl  average probabilities" md0.log
```

From the line above, we will find out that none of the attempted exchanges has been accepted. The reason is that the current setup (4 replicas to cover the temperature range 300-400K) resulted in a poor overlap of the energy distributions at different temperatures. We can easily realize this by plotting the time series of the total energy in the different replicas:

To improve the overlap of the potential energy distributions at different temperatures, we enlarge the fluctuations of the energy by sampling the Well-Tempered Ensemble (WTE). In order to do that, we need to setup a well-tempered metadynamics simulation using energy as CV. In WTE, fluctuations - the standard deviation of the energy time serie measured above - will be enhanced by a factor equal to the square root of the biasfactor. In this exercise, we will enhance fluctuations of a factor of 4, thus we will set the biasfactor equal to 16. We need to prepare 4 PLUM↩ ED input files (plumed_PTWTE.dat.0, plumed_PTWTE.dat.1,...), which will be identical to the following but for the temperature specified in the line of the METAD directive:

```
# set up three variables for Phi and Psi dihedral angles
# and total energy
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
ene: ENERGY

# Activate metadynamics in ene
# depositing a Gaussian every 250 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 140 kJoule/mol.
# Well-tempered metadynamics is activated,
# and the biasfactor is set to 16.0
#
wte: METAD ARG=ene PACE=250 HEIGHT=1.2 SIGMA=140.0 FILE=HILLS_PTWTE BIASFACTOR=16.0 TEMP=300.0

# monitor the three variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,ene,wte.bias FILE=COLVAR_PTWTE
```

(see TORSION, ENERGY, METAD, and PRINT).

Here, we use a Gaussian width larger than usual, and of the order of the fluctuations of the potential energy at 300K, as calculated from the preliminary PT run.

We run the simulation following the usual procedure:

```
mpirun -np 4 mdrun_mpi -s TOPO/topol -plumed plumed_PTWTE.dat -multi 4 -replex 100
```

If we analyze the average acceptance probability in this run:

```
grep -A2 "Repl  average probabilities" md0.log
```

we will notice that now on average 18% of the exchanges are accepted. To monitor the diffusion of each replica in temperature, we can examine the file replica_temp.xvg created by the following command line:

```
demux.pl md0.log
```

This analysis assures us that the system is efficiently diffusing in the entire temperature range and no bottlenecks are present.

Finally, as done in the previous run, we can visualize the time serie of the energy CV at all temperatures:

If we compare this plot with the one obtained in the PT run, we can notice that now the enlarged fluctuations caused by the use of WTE lead to a good overlap between energy distributions at different temperatures, thus increasing the exchange acceptance probability. At this point, we can extend our PT-WTE simulation and for example converge the free energy as a function of the dihedral angles phi and psi. Alternatively, we can accelerate sampling of the phi and psi dihedrals by combining PT-WTE with a metadynamics simulation using phi and psi as CVs (PTMetaD-WTE [39]). This can be achieved by preparing 4 PLUMED input files (plumed_PTMetaDWTE.dat.0, plumed_PTMeta↩ DWTE.dat.1,...), which will be identical to the following but for the temperature specified in the two lines containing the METAD directives:

```
# reload WTE bias
RESTART

# set up three variables for Phi and Psi dihedral angles
# and total energy
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
ene: ENERGY

# Activate metadynamics in ene
# Old Gaussians will be reloaded to perform
# the second metadynamics run in WTE.
#
wte: METAD ARG=ene PACE=99999999 HEIGHT=1.2 SIGMA=140.0 FILE=HILLS_PTWTE BIASFACTOR=16.0 TEMP=300.0

# Activate metadynamics in phi and psi
# depositing a Gaussian every 500 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 0.35 rad for both CVs.
# Well-tempered metadynamics is activated,
# and the biasfactor is set to 6.0
#
metad: METAD ARG=phi,psi PACE=500 HEIGHT=1.2 SIGMA=0.35,0.35 FILE=HILLS_PTMetaDWTE BIASFACTOR=6.0 TEMP=300.0

# monitor the three variables, the wte and metadynamics bias potentials
PRINT STRIDE=10 ARG=phi,psi,ene,wte.bias,metad.bias FILE=COLVAR_PTMetaDWTE
```

(see TORSION, ENERGY, METAD, and PRINT).

These scripts activate two metadynamics simulations. One will use the energy as CV and will reload the Gaussians deposited during the preliminary PT-WTE run. No additional Gaussians on this variable will be deposited during the PTMetaD-WTE simulation, due to the large deposition stride. A second metadynamics simulation will be activated on the dihedral angles. Please note the different parameters and biasfactors in the two metadynamics runs.

The simulation is carried out using the usual procedure:

```
mpirun -np 4 mdrun_mpi -s TOPO/topol -plumed plumed_PTMetaDWTE.dat -multi 4 -replex 100
```

10.8 Belfast tutorial: Replica exchange II and Multiple walkers

10.8.1 Aims

The aim of this tutorial is to introduce the users to the use of Bias-Exchange Metadynamics. We will go through the writing of the input files for BEMETA for a simple case of three peptide and we will use METAGUI to to analyse them. We will compare the results of WT-BEMETA and STANDARD-BEMETA with four independent runs on the four Collective Variables. Finally we will use a simplified version of BEMETA that is Multiple Walkers Metadynamics.

10.8.1.1 Learning Outcomes

Once this tutorial is completed students will:

- Know how to run a Bias-Exchange simulation using PLUMED and GROMACS
- Know how to analyse the results of BEMETA with the help of METAGUI
- Know how to run a Multiple Walker simulation

10.8.2 Resources

The `tarball` for this project contains the following files:

- system folder: a starting structure for Val-Ile-Leu system
- WTBX: a run of Well-Tempered Bias-Exchange Metadynamics ready for the analysis

10.8.3 Instructions

10.8.3.1 Bias-Exchange Metadynamics

In all variants of metadynamics the free-energy landscape of the system is reconstructed by gradually filling the local minima with gaussian hills. The dimensionality of the landscape is equal to the number of CVs which are biased, and typically a number of CVs smaller than three is employed. The reason for this is that qualitatively, if the CVs are not correlated among them, the simulation time required to fill the free-energy landscape grows exponentially with the number of CVs. This limitation can be severe when studying complex transformations or reactions in which more than say three relevant CVs can be identified.

A possible technique to overcome this limitation is parallel-tempering metadynamics, Belfast tutorial: Replica exchange I. A different solution is performing a bias-exchange simulation: in this approach a relatively large number N of CVs is chosen to describe the possible transformations of the system (e.g., to study the conformations of a peptide one may consider all the dihedral angles between amino acids). Then, N metadynamics simulations (replicas) are run on the same system at the same temperature, biasing a different CV in each replica.

Normally, in these conditions, each bias profile would converge very slowly to the equilibrium free-energy, due to hysteresis. Instead, in the bias-exchange approach every fixed number of steps (say 10,000) an exchange is attempted between a randomly selected pair of replicas a and b. The probability to accept the exchange is given by a Metropolis rule:

$$\min\left(1,\exp\left[\beta(V_G^a(x^a,t)+V_G^b(x^b,t)-V_G^a(x^b,t)-V_G^b(x^a,t))\right]\right)$$

where x^a and x^b are the coordinates of replicas a and b and $V_G^{a(b)}(x,t)$ is the metadynamics potential acting on the replica a(b). Each trajectory evolves through the high dimensional free energy landscape in the space of the CVs sequentially biased by different metadynamics potentials acting on one CV at each time. The results of the simulation are N one-dimensional projections of the free energy.

In the following example, a bias-exchange simulation is performed on a VIL peptide (zwitterionic form, in vacuum with $\varepsilon = 80$, force field amber03), using the four backbone dihedral angles as CVs.

Four replicas of the system are employed, each one biased on a different CV, thus four similar Plumed input files are prepared as follows:

- a common input file in which all the collective variables are defined:

```
MOLINFO STRUCTURE=VIL.pdb
RANDOM_EXCHANGES

cv1: TORSION ATOMS=@psi-1
cv2: TORSION ATOMS=@phi-2
cv3: TORSION ATOMS=@psi-2
cv4: TORSION ATOMS=@phi-3
```

NOTE:

1. By using MOLINFO we can use shortcut to select atoms for dihedral angles (currently @phi, @psi, @omega and @chi1 are available).
2. We use cv# as labels in order to make the output compatible with METAGUI.
3. RANDOM_EXCHANGES generates random exchanges list that are sent back to GROMACS.

- four additional input files that INCLUDE the common input and define the four METAD along the four CVs, respectively.

```
INCLUDE FILE=plumed-common.dat
be: METAD ARG=cv1 HEIGHT=0.2 SIGMA=0.2 PACE=100 GRID_MIN=-pi GRID_MAX=pi GRID_BIN=200
PRINT ARG=cv1,cv2,cv3,cv4 STRIDE=1000 FILE=COLVAR
```

NOTE:

1. in COLVAR we PRINT only the four collective variables, always in the same order in such a way that COL↩ VARs are compatible with METAGUI
2. if you want to print additional information, like the METAD bias it is possibile to use additional PRINT keyword

```
PRINT ARG=cv1,be.bias STRIDE=xxx FILE=BIAS
```

The four replicas start from the same GROMACS topology file replicated four times: topol0.tpr, topol1.tpr, topol2.tpr, topol3.tpr. Finally, GROMACS is launched as a parallel run on 4 cores, with one replica per core, with the command

```
mpirun -np 4 mdrun_mpi -s topol -plumed plumed -multi 4 -replex 2000 >& log &
```

where -replex 2000 indicates that every 2000 molecular-dynamics steps all replicas are randomly paired (e.g. 0-2 and 1-3) and exchanges are attempted between each pair (as printed in the GROMACS $*$.log files).

The same simulation can be run using WELLTEMPERED metadynamics.

10.8.3.2 Convergence of the Simulations

In the resources for this tutorial you can find the results for a 40ns long Well-Tempered Bias Exchange simulation. First of all we can try to assess the convergence of the simulations by looking at the profiles. In the "convergence" folder there is a script that calculates the free energy from the HILLS.0 file at incresing simulation lengths (i.e. every more 0.8 ns of simulation). The scripts also generate two measures of the evolution of the profiles in time:

1. time-diff.HILLS.0: where it is stored the average deviation between two successive profiles

2. KL.HILLS.0: where it is stored the average deviation between profiles correctly weigheted for the free energy of the profiles themselves (Symmetrized Kullback-Lieber divergence)

From both plots one can deduce that after 8 ns the profiles don't change significantly thus suggesting that averaging over the range 8-40ns should result in a accurate profile (we will test this using metagui). Another test is that of looking at the fluctuations of the profiles in a time window instead of looking at successive profiles:

10.8.3.3 Bias-Exchange Analysis with METAGUI

In principle Bias-Exchange Metadynamics can give as a results only N 1D free energy profiles. But the information contained in all the replicas can be used to recover multidensional free energy surfaces in $>$=N dimensions. A simple way to perform this analysis is to use METAGUI. METAGUI performs the following operations:

1. Clusterizes the trajectories on a multidimensional GRID defined by at least the biased coordinates.
2. By using the 1D free energy profiles and the clustering assigns a free energy to the cluster using a WHAM procedure.
3. Lets the user visualize the clusters.
4. Approximates the kinetics among clusters.

METAGUI (Biarnes et. al) is a plugin for VMD that implements the approch developed by Marinelli et. al 2009. It can be downloaded from the PLUMED website.

In order for the colvar and hills file to be compatible with METAGUI their header must be formatted as following:
COLVAR.#:

```
#! FIELDS time cv1 cv2 cv3 cv4
#! ACTIVE 1 1 A
#! ..
...
```

NOTE:

1. the COLVAR.# files should contain ALL the collective variables employed (all those biased in at least one replica plus those additionaly analysed). They MUST be named cv1 ... cvN.
2. the COLVAR.# files must be synchronised with the trajectories, this means that for each frame in the trajectory at time t there must be a line in each colvar at time t and viceversa. The best option is usually to analyse the trajectories a posteriori using plumed driver.
3. a keyword #! ACTIVE NBIASEDCV BIASEDCV LABEL is needed, where NBIASEDCV is the number of biased cv in that replica (not overall), BIASEDCV is the index of the biased cv in that replica (i.e. 1 for the first replica and so on); LABEL is a letter that identify the replica (usually is simply A for the first, B for the second and so on) this is usufull if two replicas are biasing the same collective variable:

```
COLVAR.0:
#! FIELDS time cv1 cv2 cv3
#! ACTIVE 1 1 A
#! ..
...
COLVAR.1:
#! FIELDS time cv1 cv2 cv3
#! ACTIVE 1 2 B
#! ..
...
COLVAR.2:
#! FIELDS time cv1 cv2 cv3
#! ACTIVE 1 2 C
#! ..
...
```

```
COLVAR.3:
#! FIELDS time cv1 cv2 cv3
#! ACTIVE 0
#! ..
...
```

In the above case Replica 0 biases cv1; replicas 1 and 2 biases cv2 while replica 3 is a neutral (unbiased) replica. cv3 is unbiased in all the replicas.

The ACTIVE keyword must be the FIRST LINE in the HILLS.# files:

HILLS.#:

```
#! ACTIVE 1 1 A
#! FIELDS time cv1 sigma_cv1 height biasf
#! ..
...
```

The above notes hold for the HILLS files as well. In the folder metagui the script check_for_metagui.sh checks if the header of your file is compatible with METAGUI, but remember that this is not enough! Synchronisation of COLVAR and trajectory files is also needed. HILLS files can be written with a different frequency but times must be consistent.

NOTE: It is important to copy HILLS files in the metagui folder.

```
./check_for_metagui.sh ../COLVAR.0
```

will tell you that the ACTIVE keyword is missing, you need to modify all the header BEFORE proceeding with the tutorial!!

In the metagui folder there is a metagui.input file:

```
WHAM_EXE        wham_bemeta.x
BASINS_EXE      kinetic_basins.x
KT 2.4900
HILLS_FILE   HILLS.0
HILLS_FILE   HILLS.1
HILLS_FILE   HILLS.2
HILLS_FILE   HILLS.3
GRO_FILE     VIL.pdb
COLVAR_FILE COLVAR.0 ../traj0.xtc "psi-1"
COLVAR_FILE COLVAR.1 ../traj1.xtc "phi-2"
COLVAR_FILE COLVAR.2 ../traj2.xtc "psi-2"
COLVAR_FILE COLVAR.3 ../traj3.xtc "phi-3"
TRAJ_SKIP 10
CVGRID 1  -3.1415 3.1415 15 PERIODIC
CVGRID 2  -3.1415 3.1415 15 PERIODIC
CVGRID 3  -3.1415 3.1415 15 PERIODIC
CVGRID 4  -3.1415 3.1415 15 PERIODIC
ACTIVE 4 1 2 3 4
T_CLUSTER 0.
T_FILL    8000.
DELTA 4
GCORR 1
TR_N_EXP 5
```

where are defined the temperature in energy units, the place where to find COLVAR, HILLS and trajectory files. A reference gro or pdb file is needed to load the trajectories. The definition of the ranges and the number of bins for the available collective variables.

Now let's start with the analysis:

1. run VMD and load metagui
2. in metagui load the metagui.input file belfast-8-mg1-fig
3. In the left section of the interface "load all" the trajectories

4. Find the Microstates

In order to visualise the microstate it is convenient to align all the structures using the VMD RMSD Trajectory tool that can be found in Extensions->Analysis.

One or more microstates can be visualised by selecting them and clicking show.

You can sort the microstates using the column name tabs, for example by clicking on size the microstates will be ordered from the larger to the smaller. If you look at the largest one it is possible to observe that by using the four selected collective variables the backbone conformation of the peptide is well defined while the sidechains can populate different rotameric states.

The equilibrium time in the analysis panel should be such that by averaging over the two halves of the remind of the simulation the profiles are the same (i.e the profile averaged between Teq and Teq+(Ttot-Teq)/2 should be the same of that averaged from Teq+(Ttot-Teq)/2 and Ttot). By clicking on COMPUTE FREE ENERGIES, the program will first generate the 1D free energy profiles from the HILLS files and then run the WHAM analysis on the microstates. Once the analysis is done it is possible to visually check the convergence of the 1D profiles one by one by clicking on the K bottons next to the HILLS.# files. The BLUE and the RED profiles are the two profiles just defined, while the GREEN is the average of the two. Now it is possible for example to sort the microstates as a function of the free energy and save them by dumping the structures for further analysis.

If you look in the metagui folder you will see a lot of files, some of them can be very usefull:

metagui/MICROSTATES: is the content of the microstates list table metagui/WHAM_RUN/VG_HILLS.#: are the opposite of the free energies calculated from the hills files metagui/WHAM_RUN/$*$.gnp: are gnuplot input files to plot the VG_HILLS.# files (i.e. gnuplot -> load "convergence..") metagui/WHAM_RUN/FES: is the result of the WHAM, for each cluster there is its free energy and the error estimate from WHAM

```
gnuplot> plot [0:40]'FES' u 2:3
```

plots the microstate error in the free energy estimate as a function of the microstates free energy. Finally in the folder metagui/FES there is script to integrate the multidimensional free energy contained in the MICROSTATES files to a 2D FES as a function of two of the used CV. To use it is enough to copy the MICROSTATES file in FES:

```
cp MICROSTATES FES/FES.4D
```

and edit the script to select the two columns of MICROSTATES on which show the integrated FES.

10.8.3.4 Multiple Walker Metadynamics

Multiple Walker metadynamics is the simplest way to parallelise a MetaD calculation: multiple simulation of the same system are run in parallel using metadynamics on the same set of collective variables. The deposited bias is shared among the replicas in such a way that the history dependent potential depends on the whole history.

We can use the same common input file defined above and then we can define four metadynamics bias in a similar way of what was done above for bias-exchange but now all the biases are defined on the same collective variables:

```
plumed.dat.#
INCLUDE FILE=plumed-common.dat

METAD ...
LABEL=mw
ARG=cv2,cv3
SIGMA=0.3,0.3
HEIGHT=0.2
PACE=100
BIASFACTOR=8
TEMP=300
GRID_MIN=-pi,-pi
GRID_MAX=pi,pi
GRID_BIN=200,200
WALKERS_MPI
... METAD

PRINT ARG=cv1,cv2,cv3,cv4 STRIDE=1000 FILE=COLVAR
```

and the simulation can be run in a similar way without doing exchanges:

```
mpirun -np 4 mdrun_mpi -s topol -plumed plumed -multi 4  >& log &
```

alternatively Multiple Walkers can be run as independent simulations sharing via the file system the biasing potential, this is usefull because it provides a parallelisation that does not need a parallel code. In this case the walkers read with a given frequency the gaussians deposited by the others and add them to their own METAD.

10.8.4 Reference

This tutorial is freely inspired to the work of Biarnes et al.

More materials can be found in

1. Marinelli, F., Pietrucci, F., Laio, A. & Piana, S. A kinetic model of trp-cage folding from multiple biased molecular dynamics simulations. PLoS Comput. Biol. 5, e1000452 (2009).
2. Biarnés, X., Pietrucci, F., Marinelli, F. & Laio, A. METAGUI. A VMD interface for analyzing metadynamics and molecular dynamics simulations. Comput. Phys. Commun. 183, 203–211 (2012).
3. Baftizadeh, F., Cossio, P., Pietrucci, F. & Laio, A. Protein folding and ligand-enzyme binding from bias-exchange metadynamics simulations. Current Physical Chemistry 2, 79–91 (2012).
4. Granata, D., Camilloni, C., Vendruscolo, M. & Laio, A. Characterization of the free-energy landscapes of proteins by NMR-guided metadynamics. Proc. Natl. Acad. Sci. U.S.A. 110, 6817–6822 (2013).
5. Raiteri, P., Laio, A., Gervasio, F. L., Micheletti, C. & Parrinello, M. Efficient reconstruction of complex free energy landscapes by multiple walkers metadynamics. J. Phys. Chem. B 110, 3533–3539 (2006).

10.9 Belfast tutorial: NMR constraints

10.9.1 Aims

This tutorial is about the use of experimental data, in particular NMR data, either as collective variables or as replica-averaged restraints in MD simulations. While the first is a just a simple extension of what we have been already doing in previous tutorials, the latter is an approach that can be used to increase the quality of a force-field in describing the properties of a specific system.

10.9.1.1 Learning Outcomes

Once this tutorial is completed students will:

- know why and how to use experimental data to define a collective variable
- know why and how to use experimental data as replica-averaged restraints in MD simulations

10.9.2 Resources

The `tarball` for this project contains the following:

- system: the files use to generate the topol?.tpr files of the first and second example
- first: an example on the use of chemical shifts as a collective variable
- second: an example on the use of chemical shifts as replica-averaged restraints
- third: an example on the use of RDCs (calculated with the theta-method) as replica-averaged restrains

10.9.3 Instructions

10.9.3.1 Experimental data as Collective Variables

In the former tutorials it has been often discussed the possibility of measuring a distance with respect to a structure representing some kind of state for a system, i.e. Belfast tutorial: Out of equilibrium dynamics. An alternative possibility is to use as a reference a set of experimental data that represent a state and measure the current deviation from the set. In plumed there are currently implemented the following NMR experimental observables: Chemical Shifts (only for proteins) CS2BACKBONE and CH3SHIFTS, NOE distances and Residual Dipolar couplings RDC. In addition NOE collective variable can be also used for PRE distances and 3J Couplings can be implemented using TORSION and MATHEVAL. Among the above listed collective variables those based on chemical shifts make use of an external library, ALMOST, that must be downloaded and compiled separately. In addition plumed must be configured in such a way to link ALMOST. Detailed instructions on how to compile PLUMED with ALMOST can be found in CS2BACKBONE.

In the following we will write the CS2BACKBONE collective variable that has been used in Gratana et al. (2013).

```
prot: GROUP ATOMS=1-862
WHOLEMOLECULES ENTITY0=prot

cs: CS2BACKBONE ATOMS=prot DATA=data FF=a03_gromacs.mdb NRES=56 FLAT=1.0 WRITE_CS=50

PRINT ARG=cs FILE=COLVAR STRIDE=100

ENDPLUMED
```

In this case the chemical shifts are those measured for the native state of the protein and can be used, together with other CVs and Bias-Exchange Metadynamics, to guide the system back and forth from the native structure. The experimental chemical shifts are in six files inside the "data/" folder (see first example in the resources tarball), one file for each nucleus. A 0 chemical shift is used where a chemical shift doesn't exist (i.e. CB of GLY) or where it has not been assigned. Additionally the data folder contains:

- camshift.db: this file is a parameter file for camshift, it is a standard file needed to calculate the chemical shifts from a structure
- a03_gromacs.mdb: this is a Amber force field in ALMOST format and it is used to map the atom names from plumed and almost (in this case we are using amber for our simulation)
- template.pdb: this is a pdb file for the protein we are simulating (i.e. editconf -f conf.gro -o template.pdb) where atoms are ordered in the same way in which are included in the main code and again it is used to map the atom in plumed with those in almost.

This example can be executed as

```
mdrun_mpi -s topol -plumed plumed
```

10.9.3.2 Replica-Averaged Restrained Simulations

NMR data, as all the equilibrium experimental data, are the result of a measure over an ensemble of structures and over time. In principle a "perfect" molecular dynamics simulations, that is a simulations with a perfect force-field and a perfect sampling can predict the outcome of an experiments in a quantitative way. Actually in most of the cases obtaining a qualitative agreement is already a fortunate outcome. In order to increase the accuracy of a force field in a system dependent manner it is possible to add to the force-field an additional term based on the agreement with a set of experimental data. This agreement is not enforced as a simple restraint because this would mean to ask the system to be always in agreement with all the experimental data at the same time, instead the restraint is applied over an AVERAGED COLLECTIVE VARIABLE where the average is performed over multiple identical simulations. In this way the is not a single replica that must be in agreement with the experimental data but they should be in agreement on average. It has been shown that this approach is equivalent in solving the problem of finding a modified version of the force field that will reproduce the provided set of experimental data withouth any additional assumption on the data themselves.

Currently ENSEMBLE AVERAGING of a collective variable can be performed only using the NMR variables (CS2BACKBONE, CH3SHIFTS, NOE and RDC).

The second example included in the resources show how the amber force field can be improved in the case of protein domain GB3 using the native state chemical shifts a replica-averaged restraint. By the fact that replica-averaging needs the use of multiple replica simulated in parallel in the same conditions it is easily complemented with BIAS-EXCHANGE or MULTIPLE WALKER metadynamics to enhance the sampling.

```
prot: GROUP ATOMS=1-862
WHOLEMOLECULES ENTITY0=prot

cs: CS2BACKBONE ATOMS=prot DATA=data FF=a03_gromacs.mdb NRES=56 FLAT=0.0 WRITE_CS=500 ENSEMBLE

cse: RESTRAINT ARG=cs AT=0. KAPPA=0. SLOPE=24

PRINT ARG=cs FILE=COLVAR STRIDE=10

ENDPLUMED
```

with respect to the case in which chemical shifts are used to define a standard collective variable, in this case the keyword ENSEMBLE tells plumed to calculate all the chemical shifts from the replicas (i.e. 4 replicas) average them and only after the averaging calculate the difference with respect to the experimental ones. On this difference that is the AVERAGED Collective Variable it is possible to apply a linear RESTRAINT (because the variable is already a sum of squared differences) that is the new term we are adding to the underlying force field.

This example can be executed as

```
mpiexec -np 4 mdrun_mpi -s topol -plumed plumed -multi 4
```

The third example show how RDC (calculated with the theta-methods) can be employed in the same way, in this case to describe the native state of Ubiquitin. In particular it is possible to observe how the RDC averaged restraint applied on the correlation between the calculated and experimental N-H and CA-HA RDCs result in the increase of the correlation of the RDCs for other bonds already on a very short time scale.

```
RDC ...
ENSEMBLE
CORRELATION
GYROM=-72.5388
SCALE=0.001060
ATOMS1=20,21 COUPLING1=8.17
ATOMS2=37,38 COUPLING2=-8.271
ATOMS3=56,57 COUPLING3=-10.489
ATOMS4=76,77 COUPLING4=-9.871
#continue....
```

In this input the first four N-H RDCs are defined.

This example can be executed as

```
mpiexec -np 8 mdrun_mpi -s topol -plumed plumed -multi 8
```

10.9.4 Reference

1. Granata, D., Camilloni, C., Vendruscolo, M. & Laio, A. Characterization of the free-energy landscapes of proteins by NMR-guided metadynamics. Proc. Natl. Acad. Sci. U.S.A. 110, 6817–6822 (2013).
2. Cavalli, A., Camilloni, C. & Vendruscolo, M. Molecular dynamics simulations with replica-averaged structural restraints generate structural ensembles according to the maximum entropy principle. J. Chem. Phys. 138, 094112 (2013).
3. Camilloni, C., Cavalli, A. & Vendruscolo, M. Replica-Averaged Metadynamics. Journal of Chemical Theory ... 9, 5610–5617 (2013).
4. Roux, B. & Weare, J. On the statistical equivalence of restrained-ensemble simulations with the maximum entropy method. J. Chem. Phys. 138, 084107 (2013).

5. Boomsma, W., Lindorff-Larsen, K. & Ferkinghoff-Borg, J. Combining Experiments and Simulations Using the Maximum Entropy Principle. PLoS Comput. Biol. 10, e1003406 (2014).

6. Camilloni, C. & Vendruscolo M. A Tensor-Free Method for the Structural and Dynamical Refinement of Proteins using Residual Dipolar Couplings. J. PHYS. CHEM. B XXX (2014).

10.10 Belfast tutorial: Steinhardt Parameters

10.10.1 Aims

This tutorial is concerned with the collective variables that we use to study order/disorder transitions such as the transition between the solid and liquid phase. In this tutorial we will look at the transition from solid to liquid as this is easier to study using simulation. The CVs we will intorduce can, and have, been used to study the transition from liquid to solid. More information can be found in the further reading section.

10.10.1.1 Learning Outcomes

Once this tutorial is completed students will:

- Know of the existence of the Steinhardt Parameters and know how to create plumed input files for calculating them.
- Appreciate that the Steinhardt Parameter for a particular atom is a vector quantity and that you can thus measure local order in a material by taking dot products of the Steinhardt Parameter vectors of adjacent atoms. Students will know how to calculate such quantities using plumed.

10.10.2 Resources

The `tarball` for this project contains the following files:

- in : An input file for the simplemd code that forms part of plumed
- input.xyz : A configuration file for Lennard-Jones solid with an fcc solid structure

10.10.3 Instructions

10.10.3.1 Simplemd

For this tutorial we will be using the MD code that is part of plumed - simplemd. This code allows us to do the simulations of `Lennard-Jones` that we require here but not much else. We will thus start this tutorial by doing some simulations with this code. You should have two files from the tarball, the first is called input.xyz and is basically an xyz file containing the posisitions of the atoms. The second meanwhile is called in and is the input to simplemd. If you open the file the contents should look something like this:

```
inputfile input.xyz
outputfile output.xyz
temperature 0.2
tstep 0.005
friction 1
forcecutoff 2.5
listcutoff  3.0
nstep 50000
nconfig 100 trajectory.xyz
nstat   10 energies.dat
```

Having run some molecular dynamics simulations in the past it should be pretty obvious what each line of the file does. One thing that might be a little strange is the units. Simplemd works with Lennard-Jones units so energy is in units of ε and length is in units of σ. This means that temperature is in units of $\frac{k_B T}{\varepsilon}$, which is why the temperature in the above file is apparently so low.

Use simplemd to run 50000 step calculations at 0.2, 0.8 and 1.2 $\frac{k_B T}{\varepsilon}$. (N.B. You will need an empty file called plumed.dat in order to run these calculations.) Visualise the trajectory output by each of your simulations using VMD. Please be aware that simplemd does not wrap the atoms into the cell box automatically. If you are at the tutorial we have resolved this problem by making it so that if you press w when you load all the atoms they will be wrapped into the box. At what temperatures did the simulations melt? What then is the melting point of the Lennard-Jones potential at this density?

10.10.3.2 Coordination Numbers

At the end of the previous section you were able to make very sophisticated judegements about whether or not the arrangment of atoms in your system was solid-like or liquid-like by simply looking at the configuration. The aim in the rest of this tutorial is to see if we can derive collective variables that are able to make an equally sophisticated judgement. For our first attempt lets use a CV which we have encoutered elsewhere, the COORDINATIONNUMBER.

Create a plumed input file that calculates the average COORDINATIONNUMBER of the atoms in your system and outputs it to a file every 100 steps. You will need to use the COORDINATIONNUMBER and PRINT actions. The switching function that tells plumed whether or not two atoms are bonded should be of type RATIONAL and should have its d_0 parameter equal to 1.3 its r_0 parameter equal to 0.2 and its n and m parameters set equal to 6 and 12 repsectively.

Rerun the simpled simulations described at the end of the previous section. Is the average coordination number good at differentiaing between solid and liquid configurations? Given your knowledge of physics/chemistry is this result surprising?

10.10.3.3 Steinhard parameter

The solid and liquid phases of a material are both relatively dense so the result at the end of the last section - the fact that the coordination number is not particularly good at differentiating between them - should not be that much of a surprise. As you will have learnt early on in your scientific education when solids melt the atoms rearrange themselves in a much less ordered fashion. The bonds between them do not necessarily break it is just that, whereas in a the solid the bonds were all at pretty well defined angles to each other, in a liquid the spread of bond angles is considerably larger. To detect the transition from solid to liquid what we need then is a coordinate that is able to recognise whether or not the geometry in the coordination spheres around each of the atoms in the system are the same or different. If these geometries are the same the system is in a solid-like configuration, whereas if they are different the system is liquid-like. The Steinhardt parameters Q3, Q4 and Q6 are coordinates that allow us to examine the coordination spheres of atoms in precisely this way. The way in which these coordinates are able to do this is explained in the `video` .

Repeat the calculations from the end of the previous section but edit the plumed.dat file so that the average Q6 parameter is calculated and printed as well as the average COORDINATIONNUMBER. In the Steinhardt parameter implementation in plumed the set of atoms in the coordination sphere of a particular atom are defined using a continuous switching function. For this calculation you should use a RATIONAL switching funciton with parameters d_0 equal to 1.3, r_0 equal to 0.2 and n and m set equal to 6 and 12 repsectively. Is the average Q6 parameter able to differentiate between the solid and liquid states?

10.10.3.4 Local versus Global

At the end of the previous section you showed that the average Q6 parameter for a system of atoms is able to tell you whether or not that collection of atoms is in a solid-like or liquid-like configuration. In this section we will now ask the question - can the Steinhardt parameter always, unambiously tell us whether particular tagged atoms are in a solid-like region of the material or whether they are in a liquid-like region of the material? This is an important question that might come up if we are looking at nucleation of a solid from the melt. Our question in this context

reads - how do we unambigously identify those atoms that are in the crystalline nucleus? A similar question would also come up when studying an interface between the solid and liquid phases. In this guise we would be asking about the extent of interface; namely, how far from the interface do we have to go before we can start thinking of the atoms as just another atom in the solid/liquid phase?

With these questions in mind our aim is to look at the distribution of values for the Q6 parameters of atoms in our system of Lennard-Jones have. If Q6 is able to unambigously tell us whether or not an atom is in a solid-like pose or a liquid-like pose then there should be no overlap in the distributions obtained for the solid and liquid phases. If there is overlap, however, then we cannot use these coordinates for the applications described in the previous paragraph. To calculate these distributions you will need to run two simulations - one at 0.2 $\frac{k_BT}{\epsilon}$ and one at 1.2 $\frac{k_BT}{\epsilon}$. For these simulation you will need plumed.dat files that calculate and output the distribution of Steinhardt parameters. In writing the plumed input for these calcualtions you will need to use the PRINT command and the Q6 command. In your Q6 instructions you will need to use the HISTOGRAM keyword - my recommendation would be to calculate a histogram consisting of 20 bins over a range starting at 0.0 and finishing at 1.0. Set the SMEAR parameter equal to 0.1. Do the distributions of Q6 parameters for the solid and liquid phase overlap? N.B. You can load the output from these simulations using GISMO and the all cvs button from the bar at the bottom.

10.10.3.5 Local Steinhardt parameters

Hopefully you saw that the distribution of Q6 parameters for the solid and liquid phase overlap at the end of the previous section. Again this is not so surprising - as you go from solid to liquid the distribution of the geometries of the coordination spheres widens. The system is now able to arrange the atoms in the coordination spheres in a much wider variety of different poses. Importantly, however, the fact that an atom is in a liquid does not preclude it from having a very-ordered, solid-like coordination environment. As such in the liquid state we will find the occasional atom with a high value for the Q6 parameter. Consequently, an ordred coordination environment does not always differentiate solid-like atoms from liquid-like atoms. The major difference is the liquid the ordered atoms will always be isolated. That is to say in the liquid atoms with an ordered coordination will always be surrounded by atoms with a disordered coordination sphere. By contrast in the solid each ordered atom will be surrounded by further ordered atoms. This observation forms the basis of the local Steinhardt parameters and the locally averaged Steinhardt parameters that are explained in this video .

Lets use plumed to calculate the distribution of LOCAL_Q6 parameters in the solid and liquid phases. Repeat the calculations from the previous section but now use the HISTOGRAM keyword to calculate the distribution of LOCAL_Q6 parameters. For the switching function in the LOCAL_Q6 parameter instruction use a RATIONAL function with d_0 equal to 1.3, r_0 equal to 0.2 and n and m set equal to 6 and 12 repsectively. For the HISTOGRAM use 20 bins starting at 0.0 and finishing at 1.0. Se the SMEAR paraemter equal to 0.1. Do the distributions of LOCAL_Q6 parameter for the solid and liquid phases overlap?

Lectner and Dellago have designed an alternative to the LOCAL_Q6 parameter that is based on taking the LOCAL_AVERAGE of the Q6 parameter arround a central atom. This quantity can be calcualted using plumed. If you have time try to use the manual to work out how.

10.10.4 Further Reading

There is a substantial literature on simulation of nucleation. Some papers are listed below but this list is far from exhaustive.

- F. Trudu, D. Donadio and M. Parrinello Freezing of a Lennard-Jones Fluid: From Nucleation to Spinodal Regime , Phys. Rev. Lett. 97 105701 (2006)
- D. Quigley and P. M. Rodger A metadynamics-based approach to sampling crystallization events , Mol. Simul. 2009
- W. Lechner and C. Dellago Accurate determination of crystal structures based on averaged local bond order parameters J. Chem. Phys 129 114707 (2008)

10.11 Cambridge tutorial

Authors

Max Bonomi and Carlo Camilloni, part of the tutorial is based on other tutorials.

Date

March 4, 2015

This document describes the PLUMED tutorial held for the Computational Biology MPhil, University of Cambridge, March 2015. The aim of this tutorial is to learn how to use PLUMED to run well-tempered and bias-exchange simulations and how to run replica-averaged metadynamics to incorporate experimental data into your simulation. Although the presented input files are correct, the users are invited to refer to the literature to understand how the parameters of enhanced sampling methods should be chosen in a real application.

Users are also encouraged to follow the links to the full PLUMED reference documentation and to wander around in the manual to discover the many available features and to do the other, more complete, tutorials. Here we are going to present only a very narrow selection of methods.

We here use PLUMED 2.1, simulations are made using GROMACS 4.6.7. However, these examples could be easily converted to other MD software.

Users are expected to write PLUMED input files based on the instructions below.

10.11.1 Alanine dipeptide: our toy model

In this tutorial we will play with alanine dipeptide (see Fig. cambridge-1-ala-fig). This rather simple molecule is useful to make many benchmark that are around for data analysis and free energy methods. It is a rather nice example since it presents two metastable states separated by a high (free) energy barrier. Here metastable states are intended as states which have a relatively low free energy compared to adjacent conformation. It is conventional use to show the two states in terms of Ramachandran dihedral angles, which are denoted with Φ and Ψ in Fig. cambridge-1-transition-fig .

10.11.2 Exercise 1. Metadynamics

10.11.2.1 Resources

The `tarball` for this project contains all the files needed to run this exercise.

10.11.2.2 Summary of theory

In metadynamics, an external history-dependent bias potential is constructed in the space of a few selected degrees of freedom $\vec{s}(q)$, generally called collective variables (CVs) [22]. This potential is built as a sum of Gaussians deposited along the trajectory in the CVs space:

$$V(\vec{s},t) = \sum_{k\tau<t} W(k\tau) \exp\left(-\sum_{i=1}^{d} \frac{(s_i - s_i(q(k\tau)))^2}{2\sigma_i^2} \right).$$

where τ is the Gaussian deposition stride, σ_i the width of the Gaussian for the ith CV, and $W(k\tau)$ the height of the Gaussian. The effect of the metadynamics bias potential is to push the system away from local minima into visiting new regions of the phase space. Furthermore, in the long time limit, the bias potential converges to minus the free energy as a function of the CVs:

$$V(\vec{s},t \to \infty) = -F(\vec{s}) + C.$$

In standard metadynamics, Gaussians of constant height are added for the entire course of a simulation. As a result, the system is eventually pushed to explore high free-energy regions and the estimate of the free energy

calculated from the bias potential oscillates around the real value. In well-tempered metadynamics [24], the height of the Gaussian is decreased with simulation time according to:

$$W(k\tau) = W_0 \exp\left(-\frac{V(\vec{s}(q(k\tau)),k\tau)}{k_B \Delta T}\right),$$

where W_0 is an initial Gaussian height, ΔT an input parameter with the dimension of a temperature, and k_B the Boltzmann constant. With this rescaling of the Gaussian height, the bias potential smoothly converges in the long time limit, but it does not fully compensate the underlying free energy:

$$V(\vec{s}, t \to \infty) = -\frac{\Delta T}{T + \Delta T} F(\vec{s}) + C.$$

where T is the temperature of the system. In the long time limit, the CVs thus sample an ensemble at a temperature $T + \Delta T$ which is higher than the system temperature T. The parameter ΔT can be chosen to regulate the extent of free-energy exploration: $\Delta T = 0$ corresponds to standard molecular dynamics, $\Delta T \to \infty$ to standard metadynamics. In well-tempered metadynamics literature and in PLUMED, you will often encounter the term "biasfactor" which is the ratio between the temperature of the CVs ($T + \Delta T$) and the system temperature (T):

$$\gamma = \frac{T + \Delta T}{T}.$$

The biasfactor should thus be carefully chosen in order for the relevant free-energy barriers to be crossed efficiently in the time scale of the simulation.

Additional information can be found in the several review papers on metadynamics [34] [35] [36].

10.11.2.3 Setup, run, and analyse a well-tempered metadynamics simulation

In this exercise, we will run a well-tempered metadynamics simulation on alanine dipeptide in vacuum, using as CVs the two backbone dihedral angles phi and psi.

In order to run this simulation we need to prepare the PLUMED input file (plumed.dat) as follows.

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate metadynamics in phi and psi
# depositing a Gaussian every 500 time steps,
# with height equal to 1.2 kJoule/mol,
# and width 0.35 rad for both CVs.
# Well-tempered metadynamics is activated,
# and the biasfactor is set to 6.0
#
metad: METAD ARG=phi,psi PACE=500 HEIGHT=1.2 SIGMA=0.35,0.35 FILE=HILLS BIASFACTOR=6.0 TEMP=300.0

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

(see TORSION, METAD, and PRINT).

The syntax for the command METAD is simple. The directive is followed by a keyword ARG followed by the labels of the CVs on which the metadynamics potential will act. The keyword PACE determines the stride of Gaussian deposition in number of time steps, while the keyword HEIGHT specifies the height of the Gaussian in kJoule/mol. To run well-tempered metadynamics, we need two additional keywords BIASFACTOR and TEMP, which specify how fast the bias potential is decreasing with time and the temperature of the simulation, respectively. For each CVs, one has to specify the width of the Gaussian by using the keyword SIGMA. Gaussian will be written to the file indicated by the keyword FILE.

Once the PLUMED input file is prepared, one has to run Gromacs with the option to activate PLUMED and read the input file:

```
mdrun_mpi -plumed
```

During the metadynamics simulation, PLUMED will create two files, named COLVAR and HILLS. The COLVAR file contains all the information specified by the PRINT command, in this case the value of the CVs every 10 steps of simulation, along with the current value of the metadynamics bias potential. The HILLS file contains a list of the Gaussians deposited along the simulation. If we give a look at the header of this file, we can find relevant information about its content. The last column contains the value of the biasfactor used, while the last but one the height of the Gaussian, which is rescaled during the simulation following the well-tempered recipe.

```
#! FIELDS time phi psi sigma_phi sigma_psi height biasf
#! SET multivariate false
#! SET min_phi -pi
#! SET max_phi pi
#! SET min_psi -pi
#! SET max_psi pi
      1.0000     -1.3100    0.0525          0.35            0.35      1.4400      6
      2.0000     -1.4054    1.9742          0.35            0.35      1.4400      6
      3.0000     -1.9997    2.5177          0.35            0.35      1.4302      6
      4.0000     -2.2256    2.1929          0.35            0.35      1.3622      6
```

If we carefully look at the height column, we will notice that in the beginning the value reported is higher than the initial height specified in the input file, which should be 1.2 kJoule/mol. In fact, this column reports the height of the Gaussian rescaled by the pre-factor that in well-tempered metadynamics relates the bias potential to the free energy. In this way, when we will use sum_hills, the sum of the Gaussians deposited will directly provide the free-energy, without further rescaling needed.

We can plot the time evolution of the CVs along with the height of the Gaussian.

The system is initialized in one of the local minimum where it starts accumulating bias. As the simulation progresses and the bias added grows, the Gaussian height is progressively reduced. After a while (t=0.8 ns), the system is able to escape the local minimum and explore a new region of the phase space. As soon as this happens, the Gaussian height is restored to the initial value and starts to decrease again. In the long time, the Gaussian height becomes smaller and smaller while the system diffuses in the entire CVs space.

10.11.2.4 Calculate free-energies and monitor convergence

One can estimate the free energy as a function of the metadynamics CVs directly from the metadynamics bias potential. In order to do so, the utility sum_hills should be used to sum the Gaussians deposited during the simulation and stored in the HILLS file.

To calculate the two-dimensional free energy as a function of phi and psi, it is sufficient to use the following command line:

```
plumed sum_hills --hills HILLS
```

The command above generates a file called fes.dat in which the free-energy surface as function of phi and psi is calculated on a regular grid. One can modify the default name for the free energy file, as well as the boundaries and bin size of the grid, by using the following options of sum_hills :

```
--outfile - specify the outputfile for sumhills
--min - the lower bounds for the grid
--max - the upper bounds for the grid
--bin - the number of bins for the grid
--spacing - grid spacing, alternative to the number of bins
```

It is also possible to calculate one-dimensional free energies from the two-dimensional metadynamics simulation. For example, if one is interested in the free energy as a function of the phi dihedral alone, the following command line should be used:

```
plumed sum_hills --hills HILLS --idw phi --kt 2.5
```

The result should look like this:

To assess the convergence of a metadynamics simulation, one can calculate the estimate of the free energy as a function of simulation time. At convergence, the reconstructed profiles should be similar. The option –stride should be used to give an estimate of the free energy every N Gaussians deposited, and the option –mintozero can be used to align the profiles by setting the global minimum to zero. If we use the following command line:

```
plumed sum_hills --hills HILLS --idw phi --kt 2.5 --stride 500 --mintozero
```

one free energy is calculated every 500 Gaussians deposited, and the global minimum is set to zero in all profiles. Try to visualize the different estimates of the free energy as a function of time.

To assess the convergence of the simulation more quantitatively, we can calculate the free-energy difference between the two local minima in the one-dimensional free energy along phi as a function of simulation time. We can use the bash script analyze_FES.sh to integrate the multiple free-energy profiles in the two basins defined by the following intervals in phi space: basin A, -3<phi<-1, basin B, 0.5<phi<1.5.

```
./analyze_FES.sh NFES -3.0 -1.0 0.5 1.5 KBT
```

where NFES is the number of profiles (free-energy estimates at different times of the simulation) generated by the option –stride of sum_hills, and KBT is the temperature in energy units (in this case KBT=2.5).

This analysis, along with the observation of the diffusive behavior in the CVs space, suggest that the simulation is converged.

10.11.3 Exercise 2. Bias-Exchange Metadynamics

10.11.3.1 Resources

The `tarball` for this project contains all the files needed to run this exercise.

10.11.3.2 Summary of theory

In well-tempered metadynamics the free-energy landscape of the system is reconstructed by gradually filling the local minima with gaussian hills. The dimensionality of the landscape is equal to the number of CVs which are biased, and typically a number of CVs smaller than four is employed. The reason for this is that qualitatively, if the CVs are not correlated among them, the simulation time required to fill the free-energy landscape grows exponentially with the number of CVs. This limitation can be severe when studying complex transformations or reactions in which more than say three relevant CVs can be identified.

Alternative solutions employ metadynamics together with other enhanced sampling techniques (i.e. Parallel Tempering or more generally Hamiltionan Replica Exchange). In particular in Bias-Exchange metadynamics the problem of sampling a multi-dimensional free-energy surface is recast in that of sampling many one-dimensional free-energies. In this approach a relatively large number N of CVs is chosen to describe the possible transformations of the system (e.g., to study the conformations of a peptide one may consider all the dihedral angles between amino acids). Then, N metadynamics simulations (replicas) are run on the same system at the same temperature, biasing a different CV in each replica. Normally, in these conditions, each bias profile would converge very slowly to the equilibrium free-energy, due to hysteresis.

In order to tackle this problem, in the bias-exchange approach every fixed number of steps (say 10,000) an exchange is attempted between a randomly selected pair of replicas a and b. The exchanges allow the replicas to gain from the sampling of the other replicas and enable the system to explore the conformational space along a large number of different directions.

The probability to accept the exchange is given by a Metropolis rule:

$$\min\left(1,\exp\left[\beta(V_G^a(x^a,t)+V_G^b(x^b,t)-V_G^a(x^b,t)-V_G^b(x^a,t))\right]\right)$$

where x^a and x^b are the coordinates of replicas a and b and $V_G^{a(b)}(x,t)$ is the metadynamics potential acting on the replica a(b). Each trajectory evolves through the high dimensional free energy landscape in the space of the CVs sequentially biased by different metadynamics potentials acting on one CV at each time. The results of the simulation are N one-dimensional projections of the free energy.

10.11.3.3 Setup, run, and analyse a well-tempered bias-exchange metadynamics simulation

In the following example, a bias-exchange simulation is performed on Alanine di-peptide. A typical input file for BE-WTMetaD is the following:

- a common input file in which all the collective variables are defined:

```
plumed-common.dat:

# read a pdb file to get topology info
MOLINFO STRUCTURE=ALAD.pdb

# tell PLUMED to generate random exchange trials
RANDOM_EXCHANGES

# set up four variables for Phi, Psi, Theta, Xi dihedral angles
phi: TORSION ATOMS=@phi-2
psi: TORSION ATOMS=@psi-2
theta: TORSION ATOMS=6,5,7,9
xi: TORSION ATOMS=16,15,17,19

PRINT ARG=phi,psi,theta,xi STRIDE=250 FILE=COLVAR
```

(see MOLINFO and RANDOM_EXCHANGES).

- additional input files that INCLUDE the common input and define the METAD along the selected CVs:

```
plumed.dat.0:

INCLUDE FILE=plumed-common.dat
METAD ARG=phi HEIGHT=1.2 SIGMA=0.25 PACE=100 GRID_MIN=-pi GRID_MAX=pi BIASFACTOR=10.0
ENDPLUMED


plumed.dat.1:

INCLUDE FILE=plumed-common.dat
METAD ARG=psi HEIGHT=1.2 SIGMA=0.25 PACE=100 GRID_MIN=-pi GRID_MAX=pi BIASFACTOR=10.0
ENDPLUMED


plumed.dat.2:

INCLUDE FILE=plumed-common.dat
METAD ARG=theta HEIGHT=1.2 SIGMA=0.50 PACE=100 GRID_MIN=-pi GRID_MAX=pi BIASFACTOR=10.0
ENDPLUMED


plumed.dat.3:

INCLUDE FILE=plumed-common.dat
METAD ARG=xi HEIGHT=1.2 SIGMA=0.50 PACE=100 GRID_MIN=-pi GRID_MAX=pi BIASFACTOR=10.0
ENDPLUMED
```

The, in this case, two replicas start from the same GROMACS topology file replicated twice: topol0.tpr, topol1.tpr, topol2.tpr and topol3.tpr. Finally, GROMACS is launched as a parallel run on 4 cores, with one replica per core, with the command

```
mpirun -np 4 mdrun_mpi -s topol -plumed plumed -multi 4 -replex 10000 >& log &
```

where -replex 10000 indicates that every 10000 molecular-dynamics steps exchanges are attempted (as printed in the GROMACS $*$.log files).

In order to have an idea of how Bias-Exchange Metadynamics works we can compare the sampling of the four replicas along the first two collective variables with respect to a free energy surface obtained by sampling in two dimensions.

The main features are that all the replicas can sample all the relevant free energy minima even if their collective variable is not meant to sample them, only the replicas with a collective variable meant to pass a barrier can sample that transition, so high energy regions are sampled only locally. This can seem a weakness in the case of alanine dipeptide but is the real strength of BE-MetaD, indeed by not knowing anything of a system it is possible to choose as many collective variables as possible and even if most of them are not particularly usefull a posteriori, they all gain from the good ones and nothing is lost.

10.11.3.4 Calculate free-energies and monitor convergence

As for the former case, one can estimate the free energy as a function of the metadynamics CVs directly from the metadynamics bias potential. This approach can only be used to recover one-dimensional free-energy profiles:

```
plumed sum_hills --hills HILLS.0 --mintozero
```

The command above generates a file called fes.dat in which the free-energy surface as function of phi is calculated on a regular grid.

It is also possible to calculate one-dimensional free energies from the two-dimensional metadynamics simulation. For example, if one is interested in the free energy as a function of the phi dihedral alone, the following command line should be used:

The result should look like this (compared with the one obtained before):

As above we can assess the convergence of a metadynamics simulation by calculating the estimate of the free energy as a function of simulation time. At convergence, the reconstructed profiles should be similar. The option –stride should be used to give an estimate of the free energy every N Gaussians deposited, and the option –mintozero can be used to align the profiles by setting the global minimum to zero. If we use the following command line:

```
plumed sum_hills --hills HILLS.0 --stride 500 --mintozero
```

Try to visualize the different estimates of the free energy as a function of time.

To assess the convergence of the simulation more quantitatively, we can calculate the free-energy difference between the two local minima in the one-dimensional free energy along phi as a function of simulation time. We can use the bash script analyze_FES.sh to integrate the multiple free-energy profiles in the two basins defined by the following intervals in phi space: basin A, -3<phi<-1, basin B, 0.5<phi<1.5.

```
./analyze_FES.sh NFES -3.0 -1.0 0.5 1.5 KBT
```

where NFES is the number of profiles (free-energy estimates at different times of the simulation) generated by the option –stride of sum_hills, and KBT is the temperature in energy units (in this case KBT=2.5).

In addition to check for the convergence of the one dimesional profiles, it is important to verify the rate of exchange among the replicas. In fact if the rate is too low or if it is not enough homegenues then the replicas can still suffer from histeris problems (ie with one replica trapped somewhere in the conformational space). Generally speaking a good probability of exchange is between 10 and 30%.

It is possible to check the exchange statistics and the detailed diffusion of the replicas among the different biases using a script provided:

```
./demux_m.pl md0.log
```

which will create two files, called replica_temp.xvg and replica_index.xvg. We can plot the first file, which reports the bias index of each configuration at a given time of the simulation. This file is extremely useful, because it allows us monitoring the replicas diffusion in bias.

10.11.4 Exercise 3. Replica-Average Metadynamics

10.11.4.1 Resources

The `tarball` for this project contains all the files needed to run this exercise.

10.11.4.2 Summary of theory

Equilibrium experimental data are the result of a measure over an ensemble of structures and over time. In principle a "perfect" molecular dynamics simulations, that is a simulations with a perfect force-field and a perfect sampling can predict the outcome of an experiment in a quantitative way. Actually in most of the cases obtaining a qualitative agreement is already a fortunate outcome. In order to increase the accuracy of a force field in a system dependent manner it is possible to add to the force-field an additional term based on the agreement with a set of experimental data. This agreement is not enforced as a simple restraint because this would mean to ask the system to be always in agreement with all the experimental data at the same time, instead the restraint is applied over an AVERAGED COLLECTIVE VARIABLE where the average is performed over multiple identical simulations (replicas). In this way there is not a single replica that must be in agreement with the experimental data but they should be in agreement on average. It has been shown that this approach is equivalent to solve the problem of finding a modified version of the force field that reproduces the provided set of experimental data without any additional assumption on the data themselves [40] [41] [42] [43] .

10.11.4.3 The system: Chignolin

In this exercise we will model the equilibrium ensemble of Chignolin by combining an implicit solvent force-field (Amber99sb) with synthetic experimental data derived from an experimentally determined ensemble of structures.

The experimental data are the following distances among CA carbons that have been averaged over the whole ensemble.

```
#RES  RES  DISTANCE(nm)
 1    3    0.656214
 1    4    0.963703
 1    5    1.197160
 1    6    1.219970
 1    7    1.206760
 1    8    1.060110
 1    9    0.858911
 1   10    0.872988
 2    4    0.656913
 2    5    0.931405
 2    6    0.953046
 2    7    0.901493
 2    8    0.809623
 2    9    0.716444
 2   10    0.882070
 3    5    0.588322
 3    6    0.660023
 3    7    0.704165
 3    8    0.720063
 3    9    0.799961
 3   10    0.981838
 4    6    0.550374
 4    7    0.572299
 4    8    0.764540
 4    9    0.939752
 4   10    1.186040
 5    7    0.613088
 5    8    0.847734
 5    9    1.084080
 5   10    1.296630
 6    8    0.591571
 6    9    0.888388
 6   10    1.102920
 7    9    0.701717
```

```
 7  10   0.987368
 8  10   0.654310
```

10.11.4.4 Setup, run and analysis

Here we will give partial information on how to setup the simulations. Users should refer to the manual and the literature cited on how to complete the information provided and thus successfully perform the exercise.

Step 1: Add the collective variables that describe the distances averaged over the ensemble to the template input files provided.

For each of the above experimental data one should define:

```
#this is the distance between CA atoms 5 and 33 belonging to residues 1 and 3, respectively.
d1: DISTANCE ATOMS=5,33
# this is the averaging of the distance among the replicas
ed1: ENSEMBLE ARG=d1
# this is the restraint applied on the averaged distance to match the experimental one
RESTRAINT ARG=ed1.d1 KAPPA=200000 AT=0.656214
```

(see DISTANCE, ENSEMBLE, and RESTRAINT).

Step 2: Setup the Bias-Exchange simulations

We will run Bias-Exchange simulations using four CVs, two of them have already been chosen (see plumed-common.dat) while the others should be selected by you. Additionally Metadynamics parameters METAD for the two selected cvs should be added in the plumed.dat.0 and plumed.dat.1 files. In particular while the PACE, HEIGHT and BIASFACTOR can be kept the same of those defined for the preselected CVs, the SIGMA should be half of the fluctuations of the chosen CV in an unbiased run ($>$1 ns).

Step 3: Run

The simulation should be run until convergence of the four monodimensional free energies, this will tipically take more than 200 ns per replica.

Step 4: Analysis

The user should check the diffusion of the replicas among the biases (see above) and send the converged free energy profiles.

Step 5: Only for the brave!

The same simulation without experimental restraints could be performed in such a way to compare the free energy profiles obtained with and without the inclusion of experimental data.

10.12 Moving from PLUMED 1 to PLUMED 2

Syntax in PLUMED 2 has been completely redesigned based on our experience using PLUMED 1, hopefully makeing it clearer, more flexible, and less error prone. The main difference is that whereas in PLUMED 1 lines could be inserted in any order, in PLUMED 2 the order of the lines matters. This is due to a major change in the internal architecture of PLUMED. In version 2, commands (or "actions") are executed in the order they are found in the input file. Because of this, you must e.g. first compute a collective variable and then print it later. More information can be found in the Section about Getting started.

Other important changes are in the way groups and units are used, as discussed below. Finally, many features appear under a different name in the new version.

10.12.1 New syntax

We know that changing the input syntax requires a lot of work from the user side to update their input files. However, we believe that the new syntax is easier to read and that it allows for more flexibility. As an example, something that in PLUMED 1.3 was:

```
HILLS HEIGHT 0.4 W_STRIDE 600
WELLTEMPERED SIMTEMP 300  BIASFACTOR 15
RGYR LIST <all>
all->
1   5  6  7   9  11  15  16  17 19
all<-
TORSION LIST 5 7 9 15 SIGMA  0.1
TORSION LIST 7 9 15 17
PRINT W_STRIDE 100
```

in PLUMED 2.x becomes:

```
all:  GROUP     ATOMS=1,5,6,7,9,11,15,16,17,19
rg:   GYRATION  ATOMS=all
t1:   TORSION ATOMS=5,7,9,15
METAD ...
  LABEL=meta
  ARG=t1 SIGMA=0.1 HEIGHT=0.4 PACE=600
  BIASFACTOR=15 TEMP=300
... METAD
t2:   TORSION ATOMS=7,9,15,17
PRINT ARG=t1,t2,meta.bias  STRIDE=100   FILE=COLVAR
```

Giving all quantities an explicit name makes the input easier to interpret. Additionally, all the parameters related to METAD are placed on a single line (actually, this is done here exploiting continuation lines). Also notice that one can customize the name of the COLVAR file. By specifying to PRINT which collective variables should be printed, one can easily decide what to print exactly and using which stride. By repeating the PRINT line one can also monitor very expensive variables with a larger stride, just putting the result on a separate file.

You might have noticed that ideas that were very difficult to implement in PLUMED 1.3 now become immediately available. As an example, one can now apply concurrently several METAD potentials [28] .

10.12.2 Groups

In PLUMED 1 groups (lists) were used for two tasks:

- To provide centers of masses to collective variables such as distances, angles, etc. This is now done by defining virtual atoms using either CENTER or COM
- To provide lists of atoms to collective variables such as coordination, gyration radius, etc. This is now done directly in the line that defines the collective variable.

If you would still like to use groups you can use the GROUP commands. Whenever the label for a GROUP action appears in the input it is replaced by the list of atoms that were specified in the GROUP.

A restraint on the distance between centers of mass in PLUMED 1 was something like:

```
DISTANCE LIST <g1> <g2>
g1->
17 20 22 30
g1<-
g2->
LOOP 37 40
g2<-
UMBRELLA CV 1 KAPPA 200 AT 1.0
PRINT W_STRIDE 100
```

The same in PLUMED 2.x reads:

```
g1: COM ATOMS=17,20,22,30
g2: COM ATOMS=37-40
d:  DISTANCE ATOMS=g1,g2
r:  RESTRAINT ARG=d KAPPA=200 AT=1.0
PRINT STRIDE=100 FILE=COLVAR ARG=d,r.*
```

Notice that virtual atoms are very powerful tools in PLUMED 2. Actually, they can be used in any collective variable where normal atoms can be used, just by calling them by name. This allows to straightforwardly define variables such as coordination between centers of mass, which would have required an ad hoc implementation in PLUMED 1.

In the example above you can also appreciate the advantage of calling collective variables by name. It is obvious here that RESTRAINT is acting on distance `d`, whereas in PLUMED 1 one had to keep track of the number of the collective variables. This was easy for a single collective variable, but could become cumbersome for complex input files.

10.12.3 Names in output files

Another advantage of having names is that when PLUMED produces an output file it can insert explicit names in the file. Consider for example this PLUMED 1 input

```
DISTANCE LIST 1 2
ANGLE    LIST 3 4 5
PRINT W_STRIDE 100
```

The first line of the COLVAR file was then

```
#!  FIELDS time cv1 cv2
```

The equivalent input file in PLUMED 2 is

```
d: DISTANCE ATOMS=1,2
a: ANGLE    ATOMS=3,4,5
PRINT ARG=d,a FILE=COLVAR STRIDE=100
```

The first line of the COLVAR file now is

```
#! FIELDS time d a
```

This makes it easy to remember what's the meaning of each column of the COLVAR file without the need to always go back to the PLUMED input to check in which order the variables were declared.

10.12.4 Units

In PLUMED 1 the input file of PLUMED was expected to be written using the same units of the MD code. This choice was made to allow users to adopt the same units they were used to. However, we realized later that this choice was not allowing the PLUMED input files to be ported between different MD code. Let's say that one was using this keyword with GROMACS (kj/mol - nm)

```
UMBRELLA CV 1 KAPPA 200 AT 1.0
```

Let's assume the variable CV 1 was a distance. The same keyword in NAMD (kcal/mol - A) should have been converted to

```
UMBRELLA CV 1 KAPPA .4780 AT 10.0
```

The conversion of `AT` is straightforward (1nm=10A), but the conversion on `KAPPA` is more error prone. This is because `KAPPA` is measured in units of energy divided by distance squared. Notice that a different factor should have been used if the CV was an angle.

Learning from this, we designed PLUMED 2 in a way that units in the PLUMED input are independent of the MD code. Technically, this is achieved by doing the conversion when coordinates and forces are passed back and forth. In this way, the same PLUMED input could be used with GROMACS and NAMD.

We decided to use as standard units in PLUMED kj/mol, nm, and ps. Perhaps this is because most of the developers are using GROMACS, which also adopts these units. However, we still allow the personalization of units by means of the UNITS keyword. Since this keyword is included directly in the PLUMED input file, even when using personalized units the input files remains perfectly portable across different MD engines.

10.12.5 Directives

What follows is a list of all the documented directives of PLUMED 1 together with their plumed 2 equivalents. Be aware that the input syntaxes for these directives are not totally equivalent. You should read the documentation for the PLUMED 2 Action.

HILLS	METAD
WELLTEMPERED	METAD with BIASFACTOR
GRID	METAD with GRID_MIN, GRID_MAX, and GRID_BIN
WRITE_GRID	METAD with GRID_WFILE, GRID_WSTRIDE
READ_GRID	**currently missing**
MULTIPLE_WALKERS	METAD with options WALKERS_ID, WALKERS_N, WALKERS_DIR, and WALKERS_RSTRIDE
NOHILLS	not needed (collective variables are not biased by default)
INTERVAL	METAD with INTERVAL
INVERT	**currently missing**
PTMETAD	not needed (replica exchange detected from MD engine)
BIASXMD	not needed (replica exchange detected from MD engine); one should anyway use RANDOM_EXCHANGES to get the normal behavior
UMBRELLA	RESTRAINT
STEER	MOVINGRESTRAINT
STEERPLAN	MOVINGRESTRAINT
ABMD	ABMD
UWALL	UPPER_WALLS
LWALL	LOWER_WALLS
EXTERNAL	EXTERNAL
COMMITMENT	COMMITTOR
PROJ_GRAD	DUMPPROJECTIONS
DAFED	**currently missing**
DISTANCE	DISTANCE
POSITION	POSITION
MINDIST	DISTANCES with keyword MIN
ANGLE	ANGLE
TORSION	TORSION
COORD	COORDINATION
HBOND	**currently missing** , can be emulated with COORDINATION
WATERBRIDGE	BRIDGE
RGYR	GYRATION
DIPOLE	DIPOLE
DIHCOR	DIHCOR
ALPHABETA	ALPHABETA
ALPHARMSD	ALPHARMSD
ANTIBETARMSD	ANTIBETARMSD
PARABETARMSD	PARABETARMSD
ELSTPOT	**currently missing**

PUCKERING	**currently missing**
S_PATH	PATHMSD, s component
Z_PATH	PATHMSD, z component
TARGETED	RMSD
ENERGY	ENERGY
HELIX	**currently missing**
PCA	**currently missing**
SPRINT	SPRINT
RDF	DISTANCES, used in combination with HISTOGRAM / BETWEEN keyword
ADF	ANGLES, used in combination with HISTOGRAM / BETWEEN keyword
POLY	COMBINE
FUNCTION	MATHEVAL
ALIGN_ATOMS	WHOLEMOLECULES

10.13 Munster tutorial

Authors

Max Bonomi and Giovanni Bussi, stealing a lot of material from other tutorials. Richard Cunha is acknowledged for beta-testing this tutorial.

Date

March 11, 2015

This document describes the PLUMED tutorial held in Munster, March 2015. The aim of this tutorial is to learn how to use PLUMED to analyze molecular dynamics simulations on the fly, to analyze existing trajectories, and to perform enhanced sampling. Although the presented input files are correct, the users are invited to **refer to the literature to understand how the parameters of enhanced sampling methods should be chosen in a real application.**

Users are also encouraged to follow the links to the full PLUMED reference documentation and to wander around in the manual to discover the many available features and to do the other, more complete, tutorials. Here we are going to present only a very narrow selection of methods.

We here use PLUMED 2.1 syntax and we explicitly note if some syntax is expected to change in PLUMED 2.2, which will be released later in 2015. All the tests here are performed on a toy system, alanine dipeptide, simulated using the AMBER99SB force field. We provide both a setup that includes explicit water, which is more realistic but slower, and a setup in gas phase, which is much faster. Simulations are made using GROMACS 4.6.7, which is here assumed to be already patched with PLUMED and properly installed. However, these examples could be easily converted to other MD software.

All the gromacs input files and analsys scripts are provided in this `tarball` .

Users are expected to write PLUMED input files based on the instructions below.

10.13.1 Alanine dipeptide: our toy model

In this tutorial we will play with alanine dipeptide (see Fig. munster-1-ala-fig). This rather simple molecule is useful to make benchmark that are around for data analysis and free energy methods. It is a nice example since it presents two metastable states separated by a high (free) energy barrier. Here metastable states are intended as states which have a relatively low free energy compared to adjacent conformations. It is conventional use to show the two states in terms of Ramachandran dihedral angles, which are denoted with Φ and Ψ in Fig. munster-1-transition-fig .

10.13.2 Monitoring collective variables

The main goal of PLUMED is to compute collective variables, which are complex descriptors than can be used to analyze a conformational change or a chemical reaction. This can be done either on the fly, that is during molecular dynamics, or a posteriori, using PLUMED as a post-processing tool. In both cases one should create an input file with a specific PLUMED syntax. A sample input file is below:

```
# compute distance between atoms 1 and 10
d: DISTANCE ATOMS=1,10
# create a virtual atom in the center between atoms 20 and 30
center: CENTER ATOMS=20,30
# compute torsional angle between atoms 1,10,20 and center
phi: TORSION ATOMS=1,10,20,center
# compute some function of previously computed variables
d2: MATHEVAL ARG=phi FUNC=cos(x) PERIODIC=NO
# print both of them every 10 step
PRINT ARG=d,phi,d2 STRIDE=10
```

(see DISTANCE, CENTER, TORSION, MATHEVAL, and PRINT)

PLUMED works using kJ/nm/ps as energy/length/time units. This can be personalized using UNITS. Notice that variables should be given a name (in the example above, `d`, `phi`, and `d2`), which is then used to refer to these variables. Lists of atoms should be provided as comma separated numbers, with no space. Virtual atoms can be created and assigned a name for later use. You can find more information on the PLUMED syntax at Getting started page of the manual. The complete documentation for all the supported collective variables can be found at the Collective variables page.

10.13.2.1 Analyze on the fly

Here we will run a plain MD on alanine dipeptide and compute two torsional angles on the fly. GROMACS needs a .tpr file, which is a binary file containing initial positions as well as force-field parameters. We also provide .gro, .mdp, and .top files, that can be modified and used to generate a new .tpr file. For this tutorial, it is sufficient to use the provided .tpr files. You will find several tpr files, namely:

- topolAwat.tpr - setup in water, initialized in state A
- topolBwat.tpr - setup in water, initialized in state B
- topolA.tpr - setup in vacuum, initialized in state A
- topolB.tpr - setup in vacuum, initialized in state B

Gromacs md can be run using on the command line:

```
> mdrun_mpi -s topolA.tpr -nsteps 10000
```

The nsteps flags can be used to change the number of timesteps and topolA.tpr is the name of the tpr file. While running, gromacs will produce an md.log file, with log information, and a traj.xtc file, with a binary trajectory. The trajectory can be visualized with VMD using a command such as

```
> vmd confout.gro tra.xtc
```

To run a simulation with gromacs+plumed you just need to add a -plumed flag

```
> mdrun_mpi -s topolA.tpr -nsteps 10000 -plumed plumed.dat
```

Here plumed.dat is the name of the plumed input file. Notice that PLUMED will write information in the md.log that could be useful to verify if the simulation has been set up properly.

10.13.2.1.1 Exercise 0

In this exercise, we will run a plain molecular dynamics simulation and monitor the Φ and Ψ dihedral angles on the fly. Using the following PLUMED input file you can monitor Φ and Ψ angles during the MD simulation

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
PRINT ARG=phi,psi STRIDE=100 FILE=colvar
```

(see TORSION and PRINT)

Notice that PLUMED is going to compute the collective variables only when necessary, that is, in this case, every 100 steps. This is not very relevant for simple variables such as torsional angles, but provides a significant speedup when using expensive collective variables.

PLUMED will write a textual file named `colvar` containing three columns: physical time, Φ and Ψ. Results can be plotted using gnuplot:

```
> gnuplot
# this shows phi as a function of time
gnuplot> plot "colvar" u 2
# this shows psi as a function of time
gnuplot> plot "colvar" u 3
# this shows psi as a function of phi
gnuplot> plot "colvar" u 2:3
```

Now try to do the same using the two different initial configurations that we provided (`topolA.tpr` and `topolB.tpr`). You can try both setup (water and vacuum). Results from 200ps (100000 steps) trajectories in vacuum are shown in Figure munster-ala-traj.

Notice that the result depends heavily on the starting structure. For the simulation in vacuum, the two free-energy minima are separated by a large barrier and, in such a short simulation, the system cannot cross it. In water the barrier is smaller and you might see some crossing. Also notice that the two clouds are well separated, indicating that these two collective variables are good enough to properly distinguish among the two minima.

As a final comment, notice that if you run twice the same calculation in the same directory, you might overwrite the resulting files. GROMACS takes automatic backup of the output files, and PLUMED does it as well. In case you are restarting a simulation, you can add the keyword RESTART at the beginning of the PLUMED input file. This will tell PLUMED to *append* files instead of taking a backup copy.

10.13.2.2 Analyze using the driver

Imagine you already made a simulation, with or without PLUMED. You might want to compute the collective variables a posteriori, from the trajectory file. You can do this by using the plumed executable on the command line. Type

```
> plumed driver --help
```

to have an idea of the possible options. See driver for the full documentation.

Here we will use the driver the compute Φ and Ψ on the already generated trajectory. Let's assume the trajectory is named `traj.xtc`. You should prepare an PLUMED input file named `analysis.dat` as:

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
PRINT ARG=phi,psi FILE=analysis
```

(see TORSION and PRINT) Notice that typically when using the driver we do not provide a STRIDE keyword to PRINT. This implies "print at every step" which, analyzing a trajectory, means "print for all the available snapshots". Then, you can use the following command:

```
> plumed driver --mf_xtc traj.xtc --plumed analysis.dat
```

Notice that PLUMED has no way to now the value of physical time from the trajectory. If you want physical time to be printed in the `analysis` file you should give more information to the driver, e.g.:

```
> plumed driver --mf_xtc traj.xtc --plumed analysis.dat --timestep 0.002 --trajectory-stride 1000
```

(see driver)

In this case we inform the driver that the `traj.xtc` file was produced in a run with a timestep of 0.002 ps and saving a snapshop every 1000 timesteps.

You might want to analyze a different collective variable, such as the gyration radius. The gyration radius tells how extended is the molecules in space. You can do it with the following plumed input file

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17

heavy: GROUP ATOMS=1,5,6,7,9,11,15,16,17,19
gyr: GYRATION ATOMS=heavy

# the same could have been achieved with
# gyr: GYRATION ATOMS=1,5,6,7,9,11,15,16,17,19

PRINT ARG=phi,psi,gyr FILE=analyze
```

(see TORSION, GYRATION, GROUP, and PRINT)

Now try to compute the time series of the gyration radius.

10.13.2.3 Periodic boundaries and explicit water

In case you are running the simulation in water, you might see that at some point this variable shows some crazy jump. The reason is that the trajectory containes coordinates where molecules are broken across periodic-boundary conditions This happens with GROMACS and some other MD code. These codes typically have tools to process trajecories and restore whole molecules. This trick is ok for the a-posteriori analysis we are trying now, but cannot used when one needs to compute a collective variable on-the-fly or, as we will see later, one wants to add a bias to that collective variable. For this reason, we implemented a workaround in PLUMED, that is the molecule should be made whole using the WHOLEMOLECULES command. What this command is doing is making a loop over all the atoms (in the order they are provided) and set the coordinates of each of them in the periodic image which is as close as possible to the coordinates of the preceeding atom. In most cases it is sufficient to list all atoms of a molecule in order. Look in the WHOLEMOLECULES page to get more information. Here this will be enough:

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17

# notice the 1-22 syntax, a shortcut for a list 1,2,3,...,22
WHOLEMOLECULES ENTITY0=1-22

heavy: GROUP ATOMS=1,5,6,7,9,11,15,16,17,19
gyr: GYRATION ATOMS=heavy

PRINT ARG=phi,psi,gyr FILE=analyze
```

(see TORSION, WHOLEMOLECULES, GROUP, GYRATION, and PRINT)

This is a very important issue that should be kept in mind when using PLUMED. Notice that starting with version 2.2 PLUMED will make molecules used in GYRATION (as well as in other variables) whole automatically, so that this extra command will not be necessary.

Notice that you can instruct PLUMED to dump on a file not only the collective variables (as we are doing with PRINT) but also the atomic positions. This is a very good way to understand what WHOLEMOLECULES is actually doing. Try the following input

```
MOLINFO STRUCTURE=../TOPO/reference.pdb
DUMPATOMS FILE=test1.gro ATOMS=1-22
WHOLEMOLECULES ENTITY0=1-22
DUMPATOMS FILE=test2.gro ATOMS=1-22
```

(see MOLINFO, DUMPATOMS, and WHOLEMOLECULES).

DUMPATOMS writes on a gro file the coordinates of the alanine dipeptide atoms. Here PLUMED will produce two files, one with coordinates *before* the application of WHOLEMOLECULES, and one with coordinates after∗ the

application of WHOLEMOLECULES. You can load both trajectories in VMD to see the difference. The MOLINFO command is here used to provide atom names to PLUMED so that the resulting gro file looks nicer in VMD.

Notice that PLUMED has several commands that manipulate atomic coordinates. One example is WHOLEMOLECULES, that fixes problems with periodic boundary conditions. COM and CENTER add new atoms, and FIT_TO_TEMPLATE can actually move atoms from their original position to align them on a template. DUMPATOMS is this very useful to check what these commands are doing and for using the PLUMED driver to manipulate MD trajectories.

10.13.2.4 Other analysis tools

PLUMED also allows you to make some analyis on the collective variables you are calculating. For example, you can compute a histogram with an input like this one

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
heavy: GROUP ATOMS=1,5,6,7,9,11,15,16,17,19
gyr: GYRATION ATOMS=heavy
PRINT ARG=phi,psi,gyr FILE=analyze
HISTOGRAM ...
  ARG=gyr
  USE_ALL_DATA
  KERNEL=discrete
  GRID_MIN=0
  GRID_MAX=1
  GRID_BIN=50
  GRID_WFILE=histogram
... HISTOGRAM
```

(see TORSION, WHOLEMOLECULES, GROUP, GYRATION, PRINT, and HISTOGRAM)

An histogram with 50 bins will be performed on the gyration radius. Try to compute the histogram for the Φ and Ψ angles.

PLUMED can do much more than a histogram, more information on analysis can be found at the page Analysis

Notice that the plumed driver can also be used directly from VMD taking advantage of the PLUMED collective variable tool developed by Toni Giorgino (http://multiscalelab.org/utilities/PlumedGUI). Just open a recent version of VMD and go to Extensions/Analysis/Collective Variable Analsys (PLUMED). This graphical interface can also be used to quickly build PLUMED input files based on template lines.

10.13.3 Biasing collective variables

We have seen that PLUMED can be used to compute collective variables. However, PLUMED is most often use to add forces on the collective variables. To this aim, we have implemented a variety of possible biases acting on collective variables. A bias works in a manner conceptually similar to the PRINT command, taking as argument one or more collective variables. However, here the STRIDE is usually omitted (that is equivalent to setting it to 1), which means that forces are applied at every timestep. In PLUMED 2.2 you will be able to change the STRIDE also for bias potentials, but that's another story. In the following we will see how to apply harmonic restraints and how to build an adaptive bias potential with metadynamics. The complete documentation for all the biasing methods available in PLUMED can be found at the Bias page.

10.13.3.1 Metadynamics

10.13.3.1.1 Summary of theory

In metadynamics, an external history-dependent bias potential is constructed in the space of a few selected degrees of freedom $\vec{s}(q)$, generally called collective variables (CVs) [22]. This potential is built as a sum of Gaussians deposited along the trajectory in the CVs space:

$$V(\vec{s},t) = \sum_{k\tau<t} W(k\tau) \exp\left(-\sum_{i=1}^{d} \frac{(s_i - s_i(q(k\tau)))^2}{2\sigma_i^2}\right).$$

where τ is the Gaussian deposition stride, σ_i the width of the Gaussian for the ith CV, and $W(k\tau)$ the height of the Gaussian. The effect of the metadynamics bias potential is to push the system away from local minima into visiting new regions of the phase space. Furthermore, in the long time limit, the bias potential converges to minus the free energy as a function of the CVs:

$$V(\vec{s},t\to\infty) = -F(\vec{s}) + C.$$

In standard metadynamics, Gaussians of constant height are added for the entire course of a simulation. As a result, the system is eventually pushed to explore high free-energy regions and the estimate of the free energy calculated from the bias potential oscillates around the real value. In well-tempered metadynamics [24], the height of the Gaussian is decreased with simulation time according to:

$$W(k\tau) = W_0 \exp\left(-\frac{V(\vec{s}(q(k\tau)),k\tau)}{k_B\Delta T}\right),$$

where W_0 is an initial Gaussian height, ΔT an input parameter with the dimension of a temperature, and k_B the Boltzmann constant. With this rescaling of the Gaussian height, the bias potential smoothly converges in the long time limit, but it does not fully compensate the underlying free energy:

$$V(\vec{s},t\to\infty) = -\frac{\Delta T}{T+\Delta T}F(\vec{s}) + C.$$

where T is the temperature of the system. In the long time limit, the CVs thus sample an ensemble at a temperature $T+\Delta T$ which is higher than the system temperature T. The parameter ΔT can be chosen to regulate the extent of free-energy exploration: $\Delta T = 0$ corresponds to standard molecular dynamics, $\Delta T \to \infty$ to standard metadynamics. In well-tempered metadynamics literature and in PLUMED, you will often encounter the term "biasfactor" which is the ratio between the temperature of the CVs ($T+\Delta T$) and the system temperature (T):

$$\gamma = \frac{T+\Delta T}{T}.$$

The biasfactor should thus be carefully chosen in order for the relevant free-energy barriers to be crossed efficiently in the time scale of the simulation.

Additional information can be found in the several review papers on metadynamics [34] [35] [36].

If you do not know exactly where you would like your collective variables to go, and just know (or suspect) that some variables have large free-energy barriers that hinder some conformational rearrangement or some chemical reaction, you can bias them using metadynamics. In this way, a time dependent, adaptive potential will be constructed that tends to disfavor visited configurations in the collective-variable space. The bias is usually built as a sum of Gaussian deposited in the already visited states.

10.13.3.1.2 Exercise 1

Now run a metadynamics simulation with the following input

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
METAD ARG=phi,psi HEIGHT=1.0 BIASFACTOR=10 SIGMA=0.35,0.35 PACE=100 GRID_MIN=-pi,-pi GRID_MAX=pi,pi
```

(see TORSION and METAD) Thus, a single METAD line will contain all the metadynamics related options, such as Gaussian height (`HEIGHT`, here in kJ/mol), stride (`PACE`, here in number of time steps), bias factor (`BIASFACTOR`, here indicates that we are going to effectively boost the temperature of the collective variables by a factor 10), and width (`SIGMA`, an array with same size as the number of collective variables).

There are two additional keywords that are optional, namely GRID_MIN and GRID_MAX. These keywords sets the range of the collective variables and tell PLUMED to keep the bias potential stored on a grid. This affects speed but, in principle, not the accuracy of the calculation. You can try to remove those keywords and see the difference.

Now, run a metadynamics simulations and check the explored collective variable space. Results from a 200ps (100000 steps) trajectory in vacuum are shown in Figure munster-ala-traj-metad.

As you can see, exploration is greatly enhanced. Notice that the explored ensemble can be tuned using the biasfactor γ. Larger γ implies that the system will explore states with higher free energy. As a rule of thumb, if you expect a barrier of the order of ΔG^*, a reasonable choice for the biasfactor is $\gamma \approx \frac{\Delta G}{2k_BT}$.

Finally, notice that METAD potential depends on the previously visited trajectories. As such, when you restart a previous simulation, it should read the previously deposited HILLS file. This is automatically triggered by the RESTART keyword.

10.13.3.1.3 Exercise 2

In this exercise, we will run a well-tempered metadynamics simulation on alanine dipeptide in vacuum, using as CV the backbone dihedral angle phi. In order to run this simulation we need to prepare the PLUMED input file (plumed.dat) as follows.

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate well-tempered metadynamics in phi depositing
# a Gaussian every 500 time steps, with initial height equal
# to 1.2 kJoule/mol, biasfactor equal to 10.0, and width to 0.35 rad

METAD ...
LABEL=metad
ARG=phi
PACE=500
HEIGHT=1.2
SIGMA=0.35
FILE=HILLS
BIASFACTOR=10.0
TEMP=300.0
GRID_MIN=-pi
GRID_MAX=pi
GRID_SPACING=0.1
... METAD

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

(see TORSION, METAD, and PRINT).

The syntax for the command METAD is simple. The directive is followed by a keyword ARG followed by the labels of the CVs on which the metadynamics potential will act. The keyword PACE determines the stride of Gaussian deposition in number of time steps, while the keyword HEIGHT specifies the height of the Gaussian in kJoule/mol. For each CVs, one has to specified the width of the Gaussian by using the keyword SIGMA. Gaussian will be written to the file indicated by the keyword FILE.

The bias potential will be stored on a grid, whose boundaries are specified by the keywords GRID_MIN and GR↩ ID_MAX. Notice that you should provide either the number of bins for every collective variable (GRID_BIN) or the desired grid spacing (GRID_SPACING). In case you provide both PLUMED will use the most conservative choice (highest number of bins) for each dimension. In case you do not provide any information about bin size (neither GRID_BIN nor GRID_SPACING) and if Gaussian width is fixed PLUMED will use 1/5 of the Gaussian width as grid spacing. This default choice should be reasonable for most applications.

Once the PLUMED input file is prepared, one has to run Gromacs with the option to activate PLUMED and read the input file:

```
mdrun_mpi -s ../TOPO/topolA.tpr -plumed plumed.dat -nsteps 5000000
```

During the metadynamics simulation, PLUMED will create two files, named COLVAR and HILLS. The COLVAR file contains all the information specified by the PRINT command, in this case the value of the CVs every 10 steps of simulation, along with the current value of the metadynamics bias potential. We can use COLVAR to visualize the

behavior of the CV during the simulation:

By inspecting Figure munster-metad-phi-fig, we can see that the system is initialized in one of the two metastable states of alanine dipeptide. After a while (t=0.1 ns), the system is pushed by the metadynamics bias potential to visit the other local minimum. As the simulation continues, the bias potential fills the underlying free-energy landscape, and the system is able to diffuse in the entire phase space.

The HILLS file contains a list of the Gaussians deposited along the simulation. If we give a look at the header of this file, we can find relevant information about its content:

```
#! FIELDS time phi psi sigma_phi sigma_psi height biasf
#! SET multivariate false
#! SET min_phi -pi
#! SET max_phi pi
#! SET min_psi -pi
#! SET max_psi pi
```

The line starting with FIELDS tells us what is displayed in the various columns of the HILLS file: the time of the simulation, the value of phi and psi, the width of the Gaussian in phi and psi, the height of the Gaussian, and the biasfactor. We can use the HILLS file to visualize the decrease of the Gaussian height during the simulation, according to the well-tempered recipe:

If we look carefully at the scale of the y-axis, we will notice that in the beginning the value of the Gaussian height is higher than the initial height specified in the input file, which should be 1.2 kJoule/mol. In fact, this column reports the height of the Gaussian rescaled by the pre-factor that in well-tempered metadynamics relates the bias potential to the free energy. In this way, when we will use sum_hills, the sum of the Gaussians deposited will directly provide the free-energy, without further rescaling needed (see below).

One can estimate the free energy as a function of the metadynamics CVs directly from the metadynamics bias potential. In order to do so, the utility sum_hills should be used to sum the Gaussians deposited during the simulation and stored in the HILLS file. To calculate the free energy as a function of phi, it is sufficient to use the following command line:

```
plumed sum_hills --hills HILLS
```

The command above generates a file called fes.dat in which the free-energy surface as function of phi is calculated on a regular grid. One can modify the default name for the free energy file, as well as the boundaries and bin size of the grid, by using the following options of sum_hills :

```
--outfile - specify the outputfile for sumhills
--min - the lower bounds for the grid
--max - the upper bounds for the grid
--bin - the number of bins for the grid
--spacing - grid spacing, alternative to the number of bins
```

The result should look like this:

To assess the convergence of a metadynamics simulation, one can calculate the estimate of the free energy as a function of simulation time. At convergence, the reconstructed profiles should be similar. The option –stride should be used to give an estimate of the free energy every N Gaussians deposited, and the option –mintozero can be used to align the profiles by setting the global minimum to zero. If we use the following command line:

```
plumed sum_hills --hills HILLS --stride 100 --mintozero
```

one free energy is calculated every 100 Gaussians deposited, and the global minimum is set to zero in all profiles. The resulting plot should look like the following:

To assess the convergence of the simulation more quantitatively, we can calculate the free-energy difference between the two local minima of the free energy along phi as a function of simulation time. We can use following script to integrate the multiple free-energy profiles in the two basins defined by the following intervals in phi space: basin A, -3<phi<-1, basin B, 0.5<phi<1.5.

```
# number of free-energy profiles
nfes= # put here the number of profiles
# minimum of basin A
minA=-3
# maximum of basin A
maxA=1
# minimum of basin B
minB=0.5
# maximum of basin B
maxB=1.5
# temperature in energy units
kbt=2.5

for((i=0;i<nfes;i++))
do
 # calculate free-energy of basin A
 A=`awk 'BEGIN{tot=0.0}{if($1!="#!" && $1>min && $1<max)tot+=exp(-$2/kbt)}END{print -kbt*log(tot)}' min=${minA
 # and basin B
 B=`awk 'BEGIN{tot=0.0}{if($1!="#!" && $1>min && $1<max)tot+=exp(-$2/kbt)}END{print -kbt*log(tot)}' min=${minB
 # calculate difference
 Delta=$(echo "${A} - ${B}" | bc -l)
 # print it
 echo $i $Delta
done
```

notice that nfes should be set to the number of profiles (free-energy estimates at different times of the simulation) generated by the option –stride of sum_hills.

This analysis, along with the observation of the diffusive behavior in the CVs space, suggest that the simulation is converged.

10.13.3.1.4 Exercise 3

In this exercise, we will run a well-tempered metadynamics simulation on alanine dipeptide in vacuum, using as CV the backbone dihedral angle psi. In order to run this simulation we need to prepare the PLUMED input file (plumed.dat) as follows.

```
# set up two variables for Phi and Psi dihedral angles
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Activate well-tempered metadynamics in psi depositing
# a Gaussian every 500 time steps, with initial height equal
# to 1.2 kJoule/mol, biasfactor equal to 10.0, and width to 0.35 rad

METAD ...
LABEL=metad
ARG=psi
PACE=500
HEIGHT=1.2
SIGMA=0.35
FILE=HILLS
BIASFACTOR=10.0
TEMP=300.0
GRID_MIN=-pi
GRID_MAX=pi
GRID_SPACING=0.1
... METAD

# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=10 ARG=phi,psi,metad.bias FILE=COLVAR
```

(see TORSION, METAD, and PRINT).

Once the PLUMED input file is prepared, one has to run Gromacs with the option to activate PLUMED and read the input file:

```
mdrun_mpi -s ../TOPO/topolA.tpr -plumed plumed.dat -nsteps 5000000
```

As we did in the previous exercise, we can use COLVAR to visualize the behavior of the CV during the simulation. Here we will plot at the same time the evolution of the metadynamics CV psi and of the other dihedral phi:

By inspecting Figure munster-metad-psi-phi-fig, we notice that something different happened compared to the previous exercise. At first the behavior of psi looks diffusive in the entire CV space. However, around t=1 ns, psi seems trapped in a region of the CV space in which it was previously diffusing without problems. The reason is that the non-biased CV phi after a while has jumped into a different local minima. Since phi is not directly biased, one has to wait for this (slow) degree of freedom to equilibrate before the free energy along psi can converge. Try to repeat the analysis done in the previous exercise (calculate the estimate of the free energy as a function of time and monitor the free-energy difference between basins) to assess the convergence of this metadynamics simulation.

10.13.3.2 Restraints

10.13.3.2.1 Biased sampling theory

A system at temperature T samples conformations from the canonical ensemble:

$$P(q) \propto e^{-\frac{U(q)}{k_B T}}$$

. Here q are the microscopic coordinates and k_B is the Boltzmann constant. Since q is a highly dimensional vector, it is often convenient to analyze it in terms of a few collective variables (see Belfast tutorial: Analyzing CVs , Belfast tutorial: Adaptive variables I , and Belfast tutorial: Adaptive variables II). The probability distribution for a CV s is

$$P(s) \propto \int dq e^{-\frac{U(q)}{k_B T}} \delta(s - s(q))$$

This probability can be expressed in energy units as a free energy landscape $F(s)$:

$$F(s) = -k_B T \log P(s)$$

.

Now we would like to modify the potential by adding a term that depends on the CV only. That is, instead of using $U(q)$, we use $U(q) + V(s(q))$. There are several reasons why one would like to introduce this potential. One is to avoid that the system samples some un-desired portion of the conformational space. As an example, imagine that you want to study dissociation of a complex of two molecules. If you perform a very long simulation you will be able to see association and dissociation. However, the typical time required for association will depend on the size of the simulation box. It could be thus convenient to limit the exploration to conformations where the distance between the two molecules is lower than a given threshold. This could be done by adding a bias potential on the distance between the two molecules. Another example is the simulation of a portion of a large molecule taken out from its initial context. The fragment alone could be unstable, and one might want to add additional potentials to keep the fragment in place. This could be done by adding a bias potential on some measure of the distance from the experimental structure (e.g. on root-mean-square deviation).

Whatever CV we decide to bias, it is very important to recognize which is the effect of this bias and, if necessary, remove it a posteriori. The biased distribution of the CV will be

$$P'(s) \propto \int dq e^{-\frac{U(q)+V(s(q))}{k_B T}} \delta(s - s(q)) \propto e^{-\frac{V(s(q))}{k_B T}} P(s)$$

and the biased free energy landscape

$$F'(s) = -k_B T \log P'(s) = F(s) + V(s) + C$$

Thus, the effect of a bias potential on the free energy is additive. Also notice the presence of an undetermined constant C. This constant is irrelevant for what concerns free-energy differences and barriers, but will be important later when we will learn the weighted-histogram method. Obviously the last equation can be inverted so as to obtain the original, unbiased free-energy landscape from the biased one just subtracting the bias potential

$$F(s) = F'(s) - V(s) + C$$

Additionally, one might be interested in recovering the distribution of an arbitrary observable. E.g., one could add a bias on the distance between two molecules and be willing to compute the unbiased distribution of some

torsional angle. In this case there is no straightforward relationship that can be used, and one has to go back to the relationship between the microscopic probabilities:

$$P(q) \propto P'(q) e^{\frac{V(s(q))}{k_B T}}$$

The consequence of this expression is that one can obtained any kind of unbiased information from a biased simulation just by weighting every sampled conformation with a weight

$$w \propto e^{\frac{V(s(q))}{k_B T}}$$

That is, frames that have been explored in spite of a high (disfavoring) bias potential V will be counted more than frames that has been explored with a less disfavoring bias potential.

10.13.3.2.2 Umbrella sampling theory

Often in interesting cases the free-energy landscape has several local minima. If these minima have free-energy differences that are on the order of a few times $k_B T$ they might all be relevant. However, if they are separated by a high saddle point in the free-energy landscape (i.e. a low probability region) than the transition between one and the other will take a lot of time and these minima will correspond to metastable states. The transition between one minimum and the other could require a time scale which is out of reach for molecular dynamics. In these situations, one could take inspiration from catalysis and try to favor in a controlled manner the conformations corresponding to the transition state.

Imagine that you know since the beginning the shape of the free-energy landscape $F(s)$ as a function of one CV s. If you perform a molecular dynamics simulation using a bias potential which is exactly equal to $-F(s)$, the biased free-energy landscape will be flat and barrierless. This potential acts as an "umbrella" that helps you to safely cross the transition state in spite of its high free energy.

It is however difficult to have an a priori guess of the free-energy landscape. We will see later how adaptive techniques such as metadynamics (Belfast tutorial: Metadynamics) can be used to this aim. Because of this reason, umbrella sampling is often used in a slightly different manner.

Imagine that you do not know the exact height of the free-energy barrier but you have an idea of where the barrier is located. You could try to just favor the sampling of the transition state by adding a harmonic restraint on the CV, e.g. in the form

$$V(s) = \frac{k}{2}(s - s_0)^2$$

. The sampled distribution will be

$$P'(q) \propto P(q) e^{\frac{-k(s(q)-s_0)^2}{2k_B T}}$$

For large values of k, only points close to s_0 will be explored. It is thus clear how one can force the system to explore only a predefined region of the space adding such a restraint. By combining simulations performed with different values of s_0, one could obtain a continuous set of simulations going from one minimum to the other crossing the transition state. In the next section we will see how to combine the information from these simulations.

If you want to just bring a collective variables to a specific value, you can use a simple restraint. Let's imagine that we want to force the Φ angle to visit a region close to $\Phi = \pi/2$. We can do it adding a restraint in Φ, with the following input

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
res: RESTRAINT ARG=phi AT=0.5pi KAPPA=5
PRINT ARG=phi,psi,res.bias
```

(see TORSION, RESTRAINT, and PRINT).

Notice that here we are printing a quantity named `res.bias`. We do this because RESTRAINT does not define a single value (that here would be theoretically named `res`) but a structure with several components. All biasing methods (including METAD) do so, as well as many collective variables (see e.g. DISTANCE used with COMP↩ ONENTS keyword). Printing the bias allows one to know how much a given snapshop was penalized. Also notice that PLUMED understands numbers in the for `{number}pi`. This is convenient when using torsions, since they are expressed in radians.

Now you can plot your trajectory with gnuplot and see the effect of KAPPA. You can also try different values of KAPPA. The stiffer the restraint, the less the collective variable will fluctuate. However, notice that a too large kappa could make the MD integrator unstable.

10.13.3.3 Moving restraints

A restraint can also be modified as a function of time. For example, if you want to bring the system from one minimum to the other, you can use a moving restraint on Φ:

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
# notice that a long line can be splitted with this syntax
MOVINGRESTRAINT ...
# also notice that a LABEL keyword can be used and is equivalent
# to adding the name at the beginning of the line with colon, as we did so far
  LABEL=res
  ARG=phi
  STEP0=0 AT0=-0.5pi KAPPA0=5
  STEP1=10000 AT0=0.5pi
...
PRINT ARG=phi,psi,res.work,res.phi_cntr FILE=colvar
```

(see TORSION, MOVINGRESTRAINT, and PRINT).

Notice that here we are plotting a few new components, namely `work` and `phi_cntr`. The former gives the work performed in pulling the restraint, and the latter the position of the restraint. Notice that if pulling is slow enough one can compute free energy profile from the work. You can plot the putative free-energy landscape with

```
> gnuplot
# column 5 is res.phi_cntr
# column 4 is res.work
gnuplot> p "colvar" u 5:4
```

10.13.3.4 Using multiple replicas

Warning

> Notice that multireplica simulations with PLUMED are fully supported with GROMACS, but only partly supported with other MD engines.

Some free-energy methods are intrinsically parallel and requires running several simultaneous simulations. This can be done with gromacs using the multi replica framework. That is, if you have 4 tpr files named topol0.tpr, topol1.tpr, topol2.tpr, topol3.tpr you can run 4 simultaneous simulations.

```
> mpirun -np 4 mdrun_mpi -s topol.tpr -plumed plumed.dat -multi 4 -nsteps 500000
```

Each of the 4 replicas will open a different topol file, and GROMACS will take care of adding the replica number before the .tpr suffix. PLUMED deals with the extra number in a slightly different way. In this case, for example, PLUMED first look for a file named `plumed.dat.X`, where X is the number of the replica. In case the file is not found, then PLUMED looks for `plumed.dat`. If also this is not found, PLUMED will complain. As a consequence, if all the replicas should use the same input file it is sufficient to put a single `plumed.dat` file, but one has also the flexibility of using separate files named `plumed.dat.0`, `plumed.dat.1` etc. Finally, notice that the way PLUMED adds suffixes will change in version 2.2, and names will be `plumed.0.dat` etc.

Also notice that providing the flag `-replex` one can instruct gromacs to perform a replica exchange simulation. Namely, from time to time gromacs will try to swap coordinates among neighboring replicas and accept of reject the exchange with a Monte Carlo procedure which also takes into account the bias potentials acting on the replicas, even if different bias potentials are used in different replicas. That is, PLUMED allows to easily implement many forms of Hamiltonian replica exchange.

10.13.3.5 Using multiple restraints with replica exchange

10.13.3.5.1 Weighted histogram analysis method theory

Let's now consider multiple simulations performed with restraints located in different positions. In particular, we will consider the i-th bias potential as V_i. The probability to observe a given value of the collective variable s is:

$$P_i(s) = \frac{P(s)e^{-\frac{V_i(s)}{k_BT}}}{\int ds' P(s')e^{-\frac{V_i(s')}{k_BT}}} = \frac{P(s)e^{-\frac{V_i(s)}{k_BT}}}{Z_i}$$

where

$$Z_i = \sum_q e^{-(U(q)+V_i(q))}$$

The likelyhood for the observation of a sequence of snapshots $q_i(t)$ (where i is the index of the trajectory and t is time) is just the product of the probability of each of the snapshots. We use here the minus-logarithm of the likelihood (so that the product is converted to a sum) that can be written as

$$\mathscr{L} = -\sum_i \int dt \log P_i(s_i(t)) = \sum_i \int dt \left(-\log P(s_i(t)) + \frac{V_i(s_i(t))}{k_BT} + \log Z_i \right)$$

One can then maximize the likelyhood by setting $\frac{\delta \mathscr{L}}{\delta P(\mathbf{s})} = 0$. After some boring algebra the following expression can be obtained

$$0 = \sum_i \int dt \left(-\frac{\delta_{\mathbf{s}_i(t),\mathbf{s}}}{P(\mathbf{s})} + \frac{e^{-\frac{V_i(\mathbf{s})}{k_BT}}}{Z_i} \right)$$

In this equation we aim at finding $P(s)$. However, also the list of normalization factors Z_i is unknown, and they should be found selfconsistently. Thus one can find the solution as

$$P(\mathbf{s}) \propto \frac{N(\mathbf{s})}{\sum_i \int dt \frac{e^{-\frac{V_i(\mathbf{s})}{k_BT}}}{Z_i}}$$

where Z is selfconsistently determined as

$$Z_i \propto \int ds' P(\mathbf{s}') e^{-\frac{V_i(\mathbf{s}')}{k_BT}}$$

These are the WHAM equations that are traditionally solved to derive the unbiased probability $P(s)$ by the combination of multiple restrained simulations. To make a slightly more general implementation, one can compute the weights that should be assigned to each snapshot, that turn out to be:

$$w_i(t) \propto \frac{1}{\sum_j \int dt \frac{e^{-\beta V_j(\mathbf{s}_i(t))}}{Z_j}}$$

The normalization factors can in turn be found from the weights as

$$Z_i \propto \frac{\sum_j \int dt e^{-\beta V_i(\mathbf{s}_j(t))} w_j(t)}{\sum_j \int dt w_j(t)}$$

This allows to straighforwardly compute averages related to other, non-biased degrees of freedom, and it is thus a bit more flexible. It is sufficient to precompute this factors w and use them to weight every single frame in the trajectory.

10.13.3.5.2 Exercise 4

In this exercise we will run multiple restraint simulations and learn how to reweight and combine data with WHAM to obtain free-energy profiles. We start with running in a replica-exchange scheme 32 simulations with a restraint on phi in different positions, ranging from -3 to 3. We will instruct gromacs to attempt an exchange between different simulations every 1000 steps.

```
nrep=32
dx=`echo "6.0 / ( $nrep - 1 )" | bc -l`

for((i=0;i<nrep;i++))
do
# center of the restraint
AT=`echo "$i * $dx - 3.0" | bc -l`

cat >plumed.dat.$i << EOF
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
#
# Impose an umbrella potential on phi
# with a spring constant of 200 kjoule/mol
# and centered in phi=AT
#
restraint-phi: RESTRAINT ARG=phi KAPPA=200.0 AT=$AT
# monitor the two variables and the bias potential
PRINT STRIDE=100 ARG=phi,psi,restraint-phi.bias FILE=COLVAR
EOF

# we initialize some replicas in A and some in B:
if((i%2==0)); then
  cp ../TOPO/topolA.tpr topol$i.tpr
else
  cp ../TOPO/topolB.tpr topol$i.tpr
fi
done

# run REM
mpirun -np $nrep mdrun_mpi -plumed plumed.dat -s topol.tpr -multi $nrep -replex 1000 -nsteps 500000
```

To be able to combine data from all the simulations, it is necessary to have an overlap between statistics collected in two adjacent umbrellas. Have a look at the plot of (phi,psi) for the different simulations to understand what is happening.

An often misunderstood fact about WHAM is that data of the different trajectories can be mixed and it is not necessary to keep track of which restraint was used to produce every single frame. Let's get the concatenated trajectory

```
trjcat_mpi -cat -f traj*.xtc -o alltraj.xtc
```

Now we should compute the value of each of the bias potentials on the entire (concatenated) trajectory.

```
nrep=32
dx=`echo "6.0 / ( $nrep - 1 )" | bc -l`

for i in `seq 0 $(( $nrep - 1 ))`
do
# center of the restraint
AT=`echo "$i * $dx - 3.0" | bc -l`

cat >plumed.dat << EOF
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
restraint-phi: RESTRAINT ARG=phi KAPPA=200.0 AT=$AT

# monitor the two variables and the bias potential
PRINT STRIDE=100 ARG=phi,psi,restraint-phi.bias FILE=ALLCOLVAR.$i
EOF

plumed driver --mf_xtc alltraj.xtc --trajectory-stride=10 --plumed plumed.dat

done
```

It is very important that this script is consistent with the one used to generate the multiple simulations above. Now, single files named ALLCOLVAR.XX will contain on the fourth column the value of the bias centered in a given position, computed on the entire concatenated trajectory.

Next step is to compute the weights self-consistently solving the WHAM equations, using the python script "wham.↩ py" contained in the SCRIPTS directory. To use this code:

```
../SCRIPTS/wham.sh ALLCOLVAR.*
```

This script will produce several files. Let's visualize "phi_fes.dat", which contains the free energy as a function of phi, and compare this with the result previously obtained with metadynamics.

10.13.3.5.3 Exercise 5

In the previous exercise, we use multiple restraint simulations to calculate the free energy as a function of the dihedral phi. The resulting free energy was in excellent agreement with our previous metadynamics simulation. In this exercise we will repeat the same procedure for the dihedral psi. At the end of the steps defined above, we can plot the free energy "psi_fes.dat" and compare it with the reference profile calculated from a metadynamics simulations using both phi and psi as CVs.

We can easily spot from the plot above that something went wrong in this multiple restraint simulations, despite we used the very same approach we adopted for the phi dihedral. The problem here is that psi is a "bad" collective variable, and the system is not able to equilibrate the missing slow degree of freedom phi in the short time scale of the umbrella simulation (1 ns). In the metadynamics exercise in which we biased only psi, we detect problems by observing the behavior of the CV as a function of simulation time. How can we detect problems in multiple restraint simulations? This is slightly more complicated, but running this kind of simulation in a replica-exchange scheme offers a convenient way to detect problems.

The first thing we need to do is to demux the replica-exchange trajectories and reconstruct the continous trajectories of the replicas across the different restraint potentials. In order to do so, we can use the following script:

```
demux.pl md0.log
trjcat_mpi -f traj*.xtc  -demux replica_index.xvg
```

This commands will generate 32 continous trajectories, named XX_trajout.xtc. We will use the driver to calculate the value of the CVs phi and psi on these trajectories.

```
nrep=32

for i in `seq 0 $(( $nrep - 1 ))`
do

cat >plumed.dat << EOF
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17

# monitor the two variables
PRINT STRIDE=100 ARG=phi,psi FILE=COLVARDEMUX.$i
EOF

plumed driver --mf_xtc ${i}_trajout.xtc --trajectory-stride=10 --plumed plumed.dat

done
```

The COLVARDEMUX.XX files will contain the value of the CVs on the demuxed trajectory. If we visualize these files we will notice that replicas sample the CVs space differently. In order for each umbrella to equilibrate the slow degrees of freedom phi, the continuous replicas must be ergodic and thus sample the same distribution in phi and psi.

Chapter 11

Index of Actions

The following page contains an alphabetically ordered list of all the Actions and command line tools that are available in PLUMED 2. For lists of Actions classified in accordance with the particular tasks that are being performed see:

- Collective variables tells you about the ways that you can calculate functions of the positions of the atoms.
- Analysis tells you about the various forms of analysis you can run on trajectories using PLUMED.
- Bias tells you about the methods that you can use to bias molecular dynamics simulations with PLUMED.

11.1 Full list of actions

ABMD	BIAS	Adds a ratchet-and-pawl like restraint on one or more variables.
ALPHABETA	COLVAR	Measures a distance including pbc between the instantaneous values of a set of torsional angles and set of reference values.
ALPHARMSD	COLVAR	Probe the alpha helical content of a protein structure.
ANGLES	MCOLVAR	Calculate functions of the distribution of angles .
ANGLE	COLVAR	Calculate an angle.
ANTIBETARMSD	COLVAR	Probe the antiparallel beta sheet content of your protein structure.
AROUND	MCOLVARF	This quantity can be used to calculate functions of the distribution of collective variables for the atoms that lie in a particular, user-specified part of of the cell.
BIASVALUE	BIAS	Takes the value of one variable and use it as a bias
BRIDGE	MCOLVAR	Calculate the number of atoms that bridge two parts of a structure
CELL	COLVAR	Calculate the components of the simulation cell

CENTER	VATOM	Calculate the center for a group of atoms, with arbitrary weights.
CH3SHIFTS	COLVAR	This collective variable calculates a scoring function based on the comparison of calculated andexperimental methyl chemical shifts.
CLASSICAL_MDS	ANALYSIS	Create a low-dimensional projection of a trajectory using the classical multidimensional scaling algorithm.
COMBINE	FUNCTION	Calculate a polynomial combination of a set of other variables.
COMMITTOR	ANALYSIS	Does a committor analysis.
COM	VATOM	Calculate the center of mass for a group of atoms.
CONSTANT	COLVAR	Return a constant quantity.
CONTACTMAP	COLVAR	Calculate the distances between a number of pairs of atoms and transform each distance by a switching function.The transformed distance can be compared with a reference value in order to calculate the squared distancebetween two contact maps. Each distance can also be weighted for a given value. CONTACTMAP can be used togetherwith FUNCPATHMSD to define a path in the contactmap space.
COORDINATIONNUMBER	MCOLVAR	Calculate the coordination numbers of atoms so that you can then calculate functions of the distribution ofcoordination numbers such as the minimum, the number less than a certain quantity and so on.
COORDINATION	COLVAR	Calculate coordination numbers.
CS2BACKBONE	COLVAR	This collective variable calculates a scoring function based on the comparison of backcalculated andexperimental backbone chemical shifts for a protein (CA, CB, C', H, HA, N).
DEBUG	GENERIC	Set some debug options.
DENSITY	MCOLVAR	Calculate functions of the density of atoms as a function of the box. This allows one to calculatethe number of atoms in half the box.

DHENERGY	COLVAR	Calculate Debye-Huckel interaction energy among GROUPA and GROUPB.
DIHCOR	COLVAR	Measures the degree of similarity between dihedral angles.
DIPOLE	COLVAR	Calculate the dipole moment for a group of atoms.
DISTANCES	MCOLVAR	Calculate the distances between one or many pairs of atoms. You can then calculate functions of the distribution ofdistances such as the minimum, the number less than a certain quantity and so on.
DISTANCE	COLVAR	Calculate the distance between a pair of atoms.
driver	TOOLS	driver is a tool that allows one to to use plumed to post-process an existing trajectory.
DRMSD	DCOLVAR	Calculate the distance RMSD with respect to a reference structure.
DUMPATOMS	ANALYSIS	Dump selected atoms on a file.
DUMPDERIVATIVES	ANALYSIS	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
DUMPFORCES	ANALYSIS	Dump the force acting on one of a values in a file.
DUMPMULTICOLVAR	ANALYSIS	Dump atom positions and multicolvar on a file.
DUMPPROJECTIONS	ANALYSIS	Dump the derivatives with respect to the input parameters for one or more objects (generally CVs, functions or biases).
ENERGY	COLVAR	Calculate the total energy of the simulation box.
ENSEMBLE	FUNCTION	Calculates the replica averaging of a collective variable over multiple replicas.
EXTERNAL	BIAS	Calculate a restraint that is defined on a grid that is read during start up
FAKE	COLVAR	This is a fake colvar container used by cltools or various other actionsand just support input and period definition
FCCUBIC	MCOLVAR	
MOLECULES	MCOLVAR	Calculate the vectors connecting a pair of atoms in order to represent the orientation of a molecule.
FIT_TO_TEMPLATE	GENERIC	This action is used to align a molecule to a template.

FLUSH	GENERIC	This command instructs plumed to flush all the open files with a user specified frequency.Notice that all files are flushed anyway every 10000 steps.
FUNCPATHMSD	FUNCTION	This function calculates path collective variables.
FUNCSUMHILLS	FUNCTION	This function is intended to be called by the command line tool sum_hillsand it is meant to integrate a HILLS file or an HILLS file interpreted as a histogram i a variety of ways. Therefore it is not expected that you use this during your dynamics (it will crash!)
gentemplate	TOOLS	gentemplate is a tool that you can use to construct template inputs for the variousactions
GHOST	VATOM	Calculate the absolute position of a ghost atom with fixed coordinatesin the local reference frame formed by three atoms. The computed ghost atom is stored as a virtual atom that can be accessed inan atom list through the the label for the GHOST action that creates it.
GPROPERTYMAP	COLVAR	Property maps but with a more flexible framework for the distance metric being used.
GROUP	GENERIC	Define a group of atoms so that a particular list of atoms can be referenced with a single labelin definitions of CVs or virtual atoms.
GYRATION	COLVAR	Calculate the radius of gyration, or other properties related to it.
HISTOGRAM	ANALYSIS	Calculate the probability density as a function of a few CVs either using kernel density estimation, or a discretehistogram estimation.
IMD	GENERIC	Use interactive molecular dynamics with VMD
INCLUDE	GENERIC	Includes an external input file, similar to "#include" in C preprocessor.
info	TOOLS	This tool allows you to obtain information about your plumed version
kt	TOOLS	Print out the value of k_BT at a particular temperature
LOAD	GENERIC	Loads a library, possibly defining new actions.

LOCAL_AVERAGE	MCOLVARF	Calculate averages over spherical regions centered on atoms
LOCAL_Q3	MCOLVARF	Calculate the local degree of order around an atoms by taking the average dot product between the q_3 vector on the central atom and the q_3 vectoron the atoms in the first coordination sphere.
LOCAL_Q4	MCOLVARF	Calculate the local degree of order around an atoms by taking the average dot product between the q_4 vector on the central atom and the q_4 vectoron the atoms in the first coordination sphere.
LOCAL_Q6	MCOLVARF	Calculate the local degree of order around an atoms by taking the average dot product between the q_6 vector on the central atom and the q_6 vectoron the atoms in the first coordination sphere.
LOWER_WALLS	BIAS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
manual	TOOLS	manual is a tool that you can use to construct the manual page for a particular action
MATHEVAL	FUNCTION	Calculate a combination of variables using a matheval expression.
METAD	BIAS	Used to performed MetaDynamics on one or more collective variables.
MOLINFO	TOPOLOGY	This command is used to provide information on the molecules that are present in your system.
MOVINGRESTRAINT	BIAS	Add a time-dependent, harmonic restraint on one or more variables.
MULTI-RMSD	DCOLVAR	Calculate the RMSD distance moved by a number of separated domains from their positions in a reference structure.
NLINKS	MCOLVARF	Calculate number of pairs of atoms/molecules that are "linked"
NOE	COLVAR	Calculates the deviation of current distances from experimental NOE derived distances.
PARABETARMSD	COLVAR	Probe the parallel beta sheet content of your protein structure.
PATHMSD	COLVAR	This Colvar calculates path collective variables.

PATH	COLVAR	Path collective variables with a more flexible framework for the distance metric being used.
PIECEWISE	FUNCTION	Compute a piecewise straight line through its arguments that passes througha set of ordered control points.
POSITION	COLVAR	Calculate the components of the position of an atom.
PRINT	ANALYSIS	Print quantities to a file.
PROPERTYMAP	COLVAR	Calculate generic property maps.
Q3	MCOLVAR	Calculate 3rd order Steinhardt parameters.
Q4	MCOLVAR	Calculate 4th order Steinhardt parameters.
Q6	MCOLVAR	Calculate 6th order Steinhardt parameters.
RANDOM_EXCHANGES	GENERIC	Set random pattern for exchanges.
RDC	COLVAR	Calculates the Residual Dipolar Coupling between two atoms.
READ	GENERIC	Read quantities from a colvar file.
RESTART	GENERIC	Activate restart.
RESTRAINT	BIAS	Adds harmonic and/or linear restraints on one or more variables.
RMSD	DCOLVAR	Calculate the RMSD with respect to a reference structure.
SIMPLECUBIC	MCOLVAR	Calculate whether or not the coordination spheres of atoms are arranged as they would be in a simplecubic structure.
simplemd	TOOLS	simplemd allows one to do molecular dynamics on systems of Lennard-Jones atoms.
SORT	FUNCTION	This function can be used to sort colvars according to their magnitudes.
SPRINT	MCOLVARF	Calculate SPRINT topological variables.
sum_hills	TOOLS	sum_hills is a tool that allows one to to use plumed to post-process an existing hills/colvar file
TARGET	DCOLVAR	This function measures the pythagorean distance from a particular structure measured in the space defined by some set of collective variables.
TEMPLATE	COLVAR	This file provides a template for if you want to introduce a new CV.
TETRAHEDRAL	MCOLVAR	
TIME	GENERIC	retrieve the time of the simulation to be used elsewere

TORSIONS	MCOLVAR	Calculate whether or not a set of torsional angles are within a particular range.
TORSION	COLVAR	Calculate a torsional angle.
UNITS	GENERIC	This command sets the internal units for the code. A new unit can be set by eitherspecifying how to convert from the plumed default unit into that new unit or by usingthe shortcuts described below. This directive MUST appear at the BEGINNING of the plumed.dat file. The same units must be used througout the plumed.dat file.
UPPER_WALLS	BIAS	Defines a wall for the value of one or more collective variables, which limits the region of the phase space accessible during the simulation.
UWALLS	MCOLVARB	Add UPPER_WALLS restraints on all the multicolvar values
VOLUME	COLVAR	Calculate the volume of the simulation box.
WHOLEMOLECULES	GENERIC	This action is used to rebuild molecules that can become split by the periodicboundary conditions.
XDISTANCES	MCOLVAR	Calculate the x components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution ofvalues such as the minimum, the number less than a certain quantity and so on.
YDISTANCES	MCOLVAR	Calculate the y components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution ofvalues such as the minimum, the number less than a certain quantity and so on.
ZDISTANCES	MCOLVAR	Calculate the z components of the vectors connecting one or many pairs of atoms. You can then calculate functions of the distribution ofvalues such as the minimum, the number less than a certain quantity and so on.

Chapter 12

Bug List

Page amber14

Charges passed from amber to plumed are in wrong units and thus lead to wrong results for variables depending on their values. See http://github.com/plumed/plumed2/issues/165 for more details.

Page ENERGY

Acceptance for replica exchange when ENERGY is biased is computed correctly only of all the replicas has the same potential energy function. This is for instance not true when using GROMACS with lambda replica exchange of with plumed-hrex branch.

Page HISTOGRAM

Option FREE-ENERGY without USE_ALL_DATA is not working properly. See #175.

Page MOLINFO

At the moment the HA1 atoms in a GLY residues are treated as if they are the CB atoms. This may or may not be true - GLY is problematic for secondary structure residues as it is achiral.

If you use WHOLEMOLECULES RESIDUES=1-10 for a 18 amino acid protein (18 amino acids + 2 terminal groups = 20 residues) the code will fail as it will not be able to interpret termnal residue 1.

Page namd-2.8

NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only

Page namd-2.9

NAMD does not currently take into account virial contributions from PLUMED. Please use constant volume simulations only

Bibliography

[1] Massimiliano Bonomi, Davide Branduardi, Giovanni Bussi, Carlo Camilloni, Davide Provasi, Paolo Raiteri, Davide Donadio, Fabrizio Marinelli, Fabio Pietrucci, Ricardo A Broglia, and Michele Parrinello. PLUMED: A portable plugin for free-energy calculations with molecular dynamics. *Computer Physics Communications*, 180(10):1961–1972, 2009. 1

[2] Gareth A. Tribello, Massimiliano Bonomi, Davide Branduardi, Carlo Camilloni, and Giovanni Bussi. Plumed 2: New feathers for an old bird. *Comput. Phys. Commun.*, 185(2):604–613, 2014. 1

[3] F. Pietrucci and A. Laio. A collective variable for the efficient exploration of protein beta-structures with metadynamics: application to sh3 and gb1. *J. Chem. Theory Comput.*, 5(9):2197–2201, 2009. 38, 41, 62

[4] Aleksandr B Sahakyan, Wim F Vranken, Andrea Cavalli, and Michele Vendruscolo. Structure-based prediction of methyl chemical shifts in proteins. *J. Biomol. NMR*, 50(4):331–346, 2011. 45

[5] Paul Robustelli, Kai Kohlhoff, Andrea Cavalli, and Michele Vendruscolo. Using NMR chemical shifts as structural restraints in molecular dynamics simulations of proteins. *Structure*, 18(8):923–933, 2010. 45, 51

[6] Daniele Granata, Carlo Camilloni, Michele Vendruscolo, and Alessandro Laio. Characterization of the free-energy landscapes of proteins by NMR-guided metadynamics. *Proc. Natl. Acad. Sci. U.S.A.*, 110(17):6817–6822, 2013. 45, 51

[7] Carlo Camilloni, Paul Robustelli, Alfonso De Simone, Andrea Cavalli, and Michele Vendruscolo. Characterization of the Conformational Equilibrium between the Two Major Substates of RNase A Using NMR Chemical Shifts. *J. Am. Chem. Soc.*, 134(9):3968–3971, 2012. 45, 51

[8] Carlo Camilloni, Andrea Cavalli, and Michele Vendruscolo. Assessment of the Use of NMR Chemical Shifts as Replica-Averaged Structural Restraints in Molecular Dynamics Simulations to Characterize the Dynamics of Proteins. *J. Phys. Chem. B*, 117(6):1838–1843, 2013. 45, 51

[9] KJ Kohlhoff, Paul Robustelli, Andrea Cavalli, Xavier Salvatella, and Michele Vendruscolo. Fast and accurate predictions of protein NMR chemical shifts from interatomic distances. *J. Am. Chem. Soc.*, 131(39):13894–13895, 2009. 51

[10] Trang N. Do, Paolo Carloni, Gabriele Varani, and Giovanni Bussi. Rna/peptide binding driven by electrostatics—insight from bidirectional pulling simulations. *Journal of Chemical Theory and Computation*, 9(3):1720–1730, 2013. 52

[11] C. Bartels and M. Karplus. Probability Distributions for Complex Systems: Adaptive Umbrella Sampling of the Potential Energy. *J. Phys. Chem. B*, 102(5):865–880, 1998. 57

[12] M. Bonomi and M. Parrinello. Enhanced sampling in the well-tempered ensemble. *Phys. Rev. Lett.*, 104:190601, 2010. 57, 258

[13] Vojtech Spiwok and Blanka Králová. Metadynamics in the conformational space nonlinearly dimensionally reduced by Isomap. *Journal of Chemical Physics*, 135(22):224504, December 2011. 58, 68

[14] Jiří Vymětal and Jiří Vondrášek. Gyration- and Inertia-Tensor-Based Collective Coordinates for Metadynamics. Application on the Conformational Behavior of Polyalanine Peptides and Trp-Cage Folding. *J. Phys. Chem. A*, page 110930112611005, 2011. 59

[15] Davide Branduardi, Francesco Luigi Gervasio, and Michele Parrinello. From A to B in free energy space. *J. Chem. Phys.*, 126(5):054103, Feb 2007. 64, 65, 82

[16] S. K. Kearsley. On the orthogonal transformation used for structural comparison. *Acta Cryst. A*, 45:208–210, 1989. 78

[17] Gareth A. Tribello, Jérôme Cuny, Hagai Eshet, and Michele Parrinello. Exploring the free energy surfaces of clusters using reconnaissance metadynamics. *J. Chem. Phys.*, 135(11):114109, 2011. 92

[18] Wolfgang Lechner and Christoph Dellago. Accurate determination of crystal structures based on averaged local bond order parameters. *The Journal of Chemical Physics*, 129(11):–, 2008. 146, 149

[19] M. Marchi and P. Ballone. Adiabatic bias molecular dynamics: A method to navigate the conformational space of complex molecular systems. *J. Chem. Phys.*, 110(8):3697–3702, 1999. 181

[20] D. Provasi and M. Filizola. Putative active states of a prototypic g-protein-coupled receptor from biased molecular dynamics. *Biophys. J.*, 98:2347—2355, 2010. 181

[21] C. Camilloni, R. A. Broglia, and G. Tiana. Hierarchy of folding and unfolding events of protein g, ci2, and acbp from explicit-solvent simulations. *J. Chem. Phys.*, 134:045105, 2011. 181

[22] A. Laio and M. Parrinello. Escaping free energy minima. *Proc. Natl. Acad. Sci. USA*, 99:12562–12566, 2002. 188, 251, 275, 290

[23] V. Babin, C. Roland, and C. Sagui. Adaptively biased molecular dynamics for free energy calculations. *J. Chem. Phys.*, 128:134101, 2008. 188

[24] A Barducci, G Bussi, and M Parrinello. Well-tempered metadynamics: A smoothly converging and tunable free-energy method. *Phys. Rev. Lett.*, 100(2):020603, Jan 2008. 189, 251, 276, 291

[25] D Branduardi, G Bussi, and M PARRINELLO. Metadynamics with adaptive Gaussians. *J. Chem. Theory Comput.*, 8(7):2247–2254, 2012. 189, 235

[26] Fahimeh Baftizadeh, Pilar Cossio, Fabio Pietrucci, and Alessandro Laio. Protein folding and ligand-enzyme binding from bias-exchange metadynamics simulations. *Curr Phys Chem*, 2:79–91, 2012. 189

[27] P. Raiteri, A. Laio, F.L. Gervasio, C. Micheletti, and M. Parrinello. Efficient reconstruction of complex free energy landscapes by multiple walkers metadynamics. *J. Phys. Chem. B*, 110:3533–3539, 2006. 189, 193

[28] Alejandro Gil-Ley and Giovanni Bussi. Enhanced conformational sampling using replica exchange with collective-variable tempering. *Journal of chemical theory and computation*, 11(3):1077–1085, 2015. 189, 283

[29] Pratyush Tiwary and Michele Parrinello. From metadynamics to dynamics. *Phys. Rev. Lett.*, 111:230602, Dec 2013. 193

[30] H. Grubmüller, B. A. Heymann, and P. Tavan. *Science*, 271:997–999, 1996. 193

[31] C. Jarzynski. Nonequilibrium equality for free energy differences. *Phys. Rev. Lett.*, 78:2690–2693, 1997. 193

[32] Stefano Piana and Alessandro Laio. A bias-exchange approach to protein folding. *J. Phys. Chem. B*, 111(17):4553–9, 2007. 215

[33] G.M. Torrie and J.P. Valleau. Nonphysical sampling distributions in monte carlo free energy estimation: Umbrella sampling. *J. Comput. Phys.*, 23:187–199, 1977. 242

[34] Alessandrio Laio and Francesco Luigi Gervasio. Metadynamics: a method to simulate rare events and reconstruct the free energy in biophysics, chemistry and material science. *Rep. Prog. Phys.*, 71:126601, 2008. 252, 276, 291

[35] Alessandro Barducci, Massimiliano Bonomi, and Michele Parrinello. Metadynamics. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(5):826–843, 2011. 252, 276, 291

[36] Ludovico Sutto, Simone Marsili, and Francesco Luigi Gervasio. New advances in metadynamics. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 2(5):771–779, 2012. 252, 276, 291

[37] Yuji Sugita and Yuko Okamoto. Replica-exchange molecular dynamics method for protein folding. *Chem. Phys. Lett.*, 314(1–2):141–151, November 1999. 257

[38] Giovanni Bussi, Francesco Luigi Gervasio, Alessandro Laio, and Michele Parrinello. Free-energy landscape for beta hairpin folding from combined parallel tempering and metadynamics. *J. Am. Chem. Soc.*, 128(41):13435–41, 2006. 258

[39] Michael Deighan, Massimiliano Bonomi, and Jim Pfaendtner. Efficient simulation of explicitly solvated proteins in the well-tempered ensemble. *Journal of Chemical Theory and Computation*, 8(7):2189–2192, 2012. 258, 263

[40] Andrea Cavalli, Carlo Camilloni, and Michele Vendruscolo. Molecular dynamics simulations with replica-averaged structural restraints generate structural ensembles according to the maximum entropy principle. *J. Chem. Phys.*, 138(9):094112, March 2013. 281

[41] Carlo Camilloni, Andrea Cavalli, and Michele Vendruscolo. Replica-Averaged Metadynamics. *J. Chem. Theory Comput.*, 9(12):5610–5617, December 2013. 281

[42] Carlo Camilloni and Michele Vendruscolo. Statistical mechanics of the denatured state of a protein using replica-averaged metadynamics. *J. Am. Chem. Soc.*, 136(25):8982–8991, June 2014. 281

[43] Wouter Boomsma, Kresten Lindorff-Larsen, and Jesper Ferkinghoff-Borg. Combining Experiments and Simulations Using the Maximum Entropy Principle. *PLoS Comput. Biol.*, 10(2):e1003406, February 2014. 281