Documentation for fiberModel package: modeling regular chromatin fibers

E. F. Koslover, C. J. Fuller,A. F. Straight, A. J. Spakowitz

Last updated November 5, 2010

The code in this package will find locally optimized structures of regular helical chromatin fibers, taking into account the elasticity of the DNA linkers and steric packing of cylindrical nucleosomes and linker DNA. In addition, the code can be used more generically to find optimal paths of a discretized worm-like chain with fixed end conditions (e.g., looping of DNA off of a single nucleosome). Local minimization, certain geometry constraints, and basin hopping to search through different local minima are all implemented. Currently, no internucleosomal potentials outside of cylindrical sterics are included.

Contents

1 Compilation and Testing Instructions

To compile and run the program, you will need the following:

- a compiler capable of handling Fortran90. The code has been tested with gfortran, g95, and pgf90 compilers. The default compiler is gfortran.
- BLAS and LAPACK libraries installed in a place where the compiler knows to look for them
- Python (version 2.5 or higher) to run various auxiliary scripts (e.g., for visualization). The scripts have been tested with Python 2.6.4 only. You will also need the NumPy extension package.
- Recommended: PyMOL to visualize pdb files.

The code has been tested on Ubuntu Linux and Mac OsX systems.

To compile with gfortran, go into the `source` directory. Type `make`. To compile with any other compiler that can handle Fortran90, type

```
make FC=compiler
```

substituting in the command you usually use to call the compiler.

If the compilation works properly, the executable `fiberModel.exe` will appear in the main directory.

To test that the code is running properly, go into the `testing` directory and type

```
./runalltests.py
```

This will run a number of test calculations to make sure everything works properly. The tests may take a few minutes to complete.

2 Usage Instructions

To run the program in the main directory, type:

```
./fiberModel.exe suffix
```

Here, `suffix` can be any string up to 100 characters in length. The program reads in all input information from a file named `param.suffix` where, again, `suffix` is the command-line argument. If no argument is supplied, it will look for a file named `param`. If the desired parameter file does not exist, the program will exit with an error. You can supply multiple suffixes to read in multiple parameter files.

The parameters in the input file are given in the format "*KEYWORD* value" where the possible keywords and values are described in Section 8. Each keyword goes on a separate line. Any line that starts with "#" is treated as a comment and ignored. Any blank line is also ignored. The keywords in the parameter file are not case sensitive. For the most part, the order in which the keywords are given does not matter. All parameters have default values, so you need only specify keywords and values when you want to change something from the default.

The *EXTRAPARAMFILES* keyword can be used to specify additional parameter files to be read for input, in addition to the original `param.suffix` files obtained from the command-line arguments.

Instructions for running specific calculations are given in more detail in Section 5. Example parameter files for the different calculations are provided in the `examples` directory.

3 Examples for a Quick Start

3.1 Example 1

Suppose you want to find several locally minimized structures for a fiber with 45 bp linkers, where the height per nucleosome is fixed to 1.5 nm.

Work in the `examples` directory.

1. Set up the parameter file.

   The parameter file `param.ex.basinhop` is designed to run such a calculation. You do not have to alter it for this example. The line "`LINKLEN 15.3D0`" sets the linker length to the desired value in nm. The line "`STARTHELH 1.50`" indicates that we want to start with a structure that has a height of 1.5 nm per nucleosome and the line "`FIXHELIXPARAM 1`" indicates that we want the height per nucleosome fixed throughout the calculation. Note that the definition of the helical coordinates, in order, is given in Section 6.1. The number of basin hops to run is set with the "`NBASINHOP 50`" line.

2. Run the code.

   On the command line, in the `examples` directory, type

   ```
   ../fiberModel.exe ex.basinhop
   ```

   A calculation with 50 basin hops will be run. The final output will give the energy, helix coordinates, and individual energy parts for each of the ten best structures found. The top 5 structures will be output (with 30 nucleosomes) into the files `ex.basinhop.1.replic.out`, `ex.basinhop.2.replic.out`, etc.

3. Convert output files to pdb.

   On the command line, still in the `examples` directory, type

   ```
   ../scripts/beadbranch2pdb.py ex.basinhop.*.replic.out -nbp 5
   ```

   This will convert the output structures to pdb, with 5 basepairs per linker segment.

4. Visualize pdb structure files.

   Open up PyMOL and make sure its current directory is set to the main fiberModel directory. Load in all the `examples/ex.basinhop.*.replic.pdb` files. A script file for doing so for this particular example is provided in `examples/loadmovie-example1.py` Documentation on how to use PyMOL can be found at `http://www.pymol.org/`.

   You can load in the structures by typing within PyMOL:

   ```
   run examples/loadmovie-example1.py
   ```

   To better visualize the structures, type within PyMOL:

   ```
   @scripts/twistedchain.pml
   ```

   Click on the `cylinders` object to hide the cylinders. Note that the nucleosome cylinders are not currently set up to work with multiple frames in PyMOL. You can now see the five best local minimum structures found with this short basinhop run.

3.2 Example 2

Suppose you want to find a locally minimized structure for a fiber with 40 bp linker length, a nucleosome line density of 8 nucleosomes per 11 nm, and nucleosomes oriented with their symmetry axes perpendicular to the fiber axis. Again, work in the `examples` directory.

1. Set up the parameter file.

   Copy over one of the example optimization files (e.g.: `param.ex2.optimize`) to a new parameter file (e.g.: `param.example2`) in the main directory.

   Open up your new parameter file `param.example2` in a text editor.

   You want a linker length of 40bp $\times \frac{0.34\text{nm}}{\text{bp}} = 13.6\text{nm}$ Alter the linker length line (starting with `LINKLEN`) to "`LINKLEN 13.6`".

   You want to fix the height to $h = 11/8 = 1.375$ nm per nucleosome. So alter the corresponding keyword line to "`STARTHELH 1.375`". This will set the starting structure to the right height per nucleosome.

   You want an angle of $\beta = \pi/2$ between the nucleosome axis and the fiber axis, to alter the corresponding keyword line to: "`STARTHELB 1.57`". This will set the nucleosome tilt angle in the starting structure.

   You want the height per nucleosome (first helix coordinate) and the nucleosome tilt angle (fifth helix coordinate) fixed throughout the calculation, so alter the corresponding keyword line to "`FIXHELIXPARAM 1 5`". Note that the definition of the helical coordinates, in order, is given in Section 6.1.

   Finally, suppose you are interested in watching the fiber as it minimizes. You can tell the program to dump out a structure with 30 nucleosomes after every 20 minimization steps by adding somewhere in the parameter file the line "`DUMPCURRENT 20 30`". In order to have the structures dumped into files named `example2.0.dump.out`, `example2.20.dump.out`, etc. add the following line somewhere in the parameter file: "`DUMPFILE *.#.dump.out`".

   Compare your `param.example2` file to `param.example2.ref` if you want to make sure the parameters are correctly set up.

2. Run the code.

   On the command line, still in the examples directory,(assuming that is where your `param.example2` file is located), type

   ```
   ../fiberModel.exe example2
   ```

   The resulting output will tell you the final helix coordinates and energy associated with the optimized structure.

3. Convert output files to pdb.

   On the command line, type

   ```
   ../scripts/beadbranch2pdb.py example2.*.dump.out -nbp 4
   ```

   This will convert all the output files dumped throughout the minimization to pdb, with 4 basepairs per linker segment.

4. Visualize pdb structure files.

   Open up PyMOL and make sure its current directory is set to the main `fiberModel` directory. Load in all the `examples/example2.*.dump.pdb` files. A script file for doing so for this particular example is provided in `examples/loadmovie-example2.py` Documentation on how to use PyMOL can be found at `http://www.pymol.org/`.

   You can load in the structures by typing within PyMOL:

   ```
   run examples/loadmovie-example2.py
   ```

   To better visualize the structures, type within PyMOL:

   ```
   @scripts/twistedchain.pml
   ```

   Click on the cylinder object to hide the cylinders. Note that the nucleosome cylinders are not currently set up to work with multiple frames in PyMOL. You can now play the movie within PyMOL to watch the structure minimize.

4 Auxiliary Scripts

Several python scripts have been provided in the `scripts` directory for help in setting up parameter files and visualizing resulting structures.

4.1 Setting up parameter files

- `geomFromPDB.py`
  This script will extract bound DNA beads (filler beads) and edge conditions from a PDB file containing single nucleosome coordinates. **WARNING:** this has only been tested with the 1KX5.pdb structure file and will not necessarily generalize to other files with slightly different formats.

  The script can also generate end conditions for a partially unwrapped nucleosome.

  Run the script without arguments to print out usage information. Simplest run of the script is:

  ```
  scripts/geomFromPDB.py examples/1KX5.pdb paramfile
  ```

  The script will replace any keywords *POSP, POSM, BENDTANP, BENDTANM, XAXP, XAXM* already present in the parameter file. It will also generate a pdb file (`1KX5-mod.pdb` by default) which can be used to visualize the calculated end conditions. Use the `viewnucleosome.pml` script to visualize the resulting nucleosome structure in PyMOL.

- `dnaloopFromPDB.py`
  This script will extract edge geometries and bound DNA beads from a pdb file for a DNA loop with edges fixed between two bound DNA basepairs on a nucleosome structure. **WARNING:** this has only been tested with the 1KX5.pdb structure file and will not necessarily generalize to other files with slightly different formats.

Run the script without arguments to print out usage information. Simplest run of the script (to generate edge conditions for a DNA loop between the 10th and the 110th bound basepair) is:

```
scripts/dnaloopFromPDB.py examples/1KX5.pdb paramfile 10 110
```

The script will replace any keywords *POSP, POSM, BENDTANP, BENDTANM, XAXP, XAXM* already present in the parameter file. It will also generate a pdb file (`examples/1KX5-dnaloop.pdb` by default) which can be used to visualize the calculated end conditions. Use the `viewnucleosome.pml` script to view the resulting nucleosome structure in PyMOL.

4.2 Visualizing structures

- `beadbranch2pdb.py`
  This script will convert a structure file (or multiple files) output by the fibermodel.exe code (usually ending in the extension `.out`) to a PDB file that can be visualized with PyMOL. Run the script without any arguments for usage instructions and optional arguments Simplest usage to convert file `struct.out` to a PDB is,

  ```
  scripts/beadbranch2pdb.py struct.out
  ```

  which will generate the file `struct.pdb` The `struct.out` file can also be replaced by a glob of filenames, such as `struct.*.out`

  Recommended sets of arguments for visualizing the results of the example runs are given within the `examples/param.ex.*` parameter files.

  The resulting pdb files are designed to be visualized in PyMOL with the script `scripts/twistedchain.pml`

- `twistedchain.pml`
  PyMOL script for visualizing a chromatin fiber pdb file generated with `beadbranch2pdb.py`

  Once the pdb file has been loaded into PyMOL, type `@scripts/twistedchain.pml` to run the script (assuming the current directory for the PyMOL session is the main directory)

  In order for the nucleosomes to be visualized as cylinders automatically, the current directory of the PyMOL session must be the main fiberModel directory. Otherwise, visualization of the cylinders can be done manually using the `makeCylinders` function described below

- `makeCylinders`
  This function, designed for use within PyMOL will represent the nucleosomes as CGO cylinders. It is found in the file `scripts/displayCylinders.py`.

  Within PyMOL, to use this function you must first load it in by typing

  ```
  run displayCylinders.py
  ```

  Note that this is done automatically if you run the `twistedchain.pml` visualization script. This needs to be done only once in a given PyMOL session.

  Run the function as follows:

```
makeCylinders AC, AC1, AC2, 52, 55, 0.5
```

This looks for atoms named AC to define the nucleosome centers, atoms named AC1 and AC2 to define the cylinder axes (these defaults are the ones generated by the `beadbranch2pdb.py` script). The nucleosomes are then represented as cylinders of radius 52Å, height 55Å, and transparance of 50%.

Note that to change the cylinder representation, you must first delete any "cylinder" object already present in the PyMOL session. Also, currently cylinders are not set up for multi-frame PyMOL sessions.

- `viewnucleosome.pml`
  PyMOL script to visualize the nucleosome geometry calculated with `bendFromPDB.py`. This will trace out the positions of the filler beads along the DNA axis and show the z-axes and x-axes defined by *BENDTANP, BENDTANM, XAXP, XAXM* at either edge of the bound DNA. To run it within PyMOL (from the main directory, after loading in the 1KX5-mod.pdb or equivalent structure), type:

  ```
  @scripts/viewnucleosome.pml
  ```

- `viewsinglenuc.pml`
  PyMOL script to visualize an output pdb file as created by `beadbranch2pdb.py` after a calculation for a DNA loop on a single nucleosome (see Sec.7).

  This works almost exactly like `twistedchain.pml` but with a slightly different coloring scheme.

5 Description of Specific Calculations

The *ACTION* keyword specifies what type of calculation will be done. The possible actions are to dump out a structure and its energy (*ACTION* GETSTRUCT), run a gradient-based optimization (*ACTION* OPTIMIZE), run a basin-hopping search (*ACTION* BASINHOP), or parse through a database generated by a previous basin-hopping search (*ACTION* DATABASEPARSE).

5.1 GETSTRUCT action

The program simply outputs the starting structure of the nucleosome array, as well as printing out its energy and, if desired, certain geometric information. See Sec. 6.2 for how the starting structure is obtained.

In addition to the keywords involved in setting up the structure (Sec. 6.2) and calculating the energy (Sec. 6.3), the following keywords are relevant for this action: *OUTFILE, REPLICATESTRUCT, PRINTENERGYPARTS, PRINTFIBERDIAM, PRINTNUCDIST*

OUTFILE specifies the output file for the structure containing only 2 nucleosomes and the connecting linker. If *REPLICATESTRUCT* is supplied, then the structure will be replicated as a regular fiber to have more nucleosomes and the replicated configuration will be output in a different file. If *PRINTENERGYPARTS* is supplied, the individual components of the energy (bend, twist, stretch, nucleosome-nucleosome sterics, linker-linker sterics, and nucleosome-linker sterics) will be printed out. If *PRINTFIBERDIAM* is supplied, the overall diameter of the fiber (stretching out to

the furthest points of the cylindrical nucleosomes) will be printed out. If *PRINTFIBERDIAM* is supplied, the code will also print the distances between the cylindrical shell of the first nucleosome and that of each subsequent nucleosome.

`examples/param.ex.getstruct` and `examples/param.ex2.getstruct` are example files for this type of calculation.

5.2 OPTIMIZE action

Starting from a given structure of the fiber, use BFGS optimization[1] to find a nearby local minimum in the configuration space of regular fibers.

In addition to the keywords associated with setting the starting structure (Sec.6.2), calculating energies (Sec.6.3), and outputting the final minimized structure (see all keywords in Sec.5.1), the following keywords are relevant for this action: *OPTOL, DUMPCURRENT, DUMPFILE, OPTPRINTFREQ, MAXOPTSTEP, MAXOPTATTEMPT, VERBOSE, MAXEJUMP, MAXDCR, STEPDCR, MAXSTEPSIZE*

The structure will be minimized until the mean square force falls below *OPTOL*. If *DUMPCURRENT* is set, then the current structure after every few minimization steps will be dumped out into *DUMPFILE*. This allows you to create a movie of the structure as it minimizes.

Information about the current energy, mean square force, and height per nucleosome of the structure is printed throughout the minimization at every *OPTPRINTFREQ* steps. The maximum number of allowed optimization steps is *MAXOPTSTEP*.

`examples/param.ex.optimize` and `examples/param.ex2.optimize` are example files for this type of calculation.

5.3 BASINHOP action

Basin hopping[2] is a global optimization method which combines Monte-Carlo (MC) steps and gradient-based local minimization. After each MC step, the configuration is allowed to slide down into a nearby local minimum and it is this local minimum that defines the energy corresponding to that MC step.

The following keywords are relevant specifically to the basin-hopping calculation: *RNGSEED, NBASINHOP,BHTOL, BHTEMP, NCONFIGSAVE, NCONFIGDUMP, DBCONFIGMAXE, DATAFILE, READDATAFILE, MCSTARTRANGE, BHCHECKRANGE, BHFACC, BHFSAME, SAMECONF, MAXHOPTRY, FLIPNUCMC DBMINIMIZE, OPTOL, DBSORT, PRINTENERGYPARTS, PRINTFIBERDIAM, PRINTNUCDIST, DUMPFILE, REPLICATESTRUCT*

The *RNGSEED* keyword can be used to seed the random number generator.

The program carries out a total of *NBASINHOP* Monte Carlo hops, each of which is followed by local minimization to a tolerance of *BHTOL*. The effective temperature used in deciding whether to accept each Monte Carlo hop is set by *BHTEMP*. Each new configuration found is saved in a database that contains up to *NCONFIGSAVE* structures ordered from lowest to highest energy. Any structures with an energy above *DBCONFIGMAXE* are not saved. Each time a new configuration is added to the database, the entire database of structures is output to the second file name specified by *DATAFILE*, which also tracks how many hops have been carried out thus far. If *READDATAFILE* is set, then the program starts by reading in all structures from a previously generated database (first file name specified with *DATAFILE* and continuing the basin-hopping run from that point (starting with a certain number of hops already done).

The initial range of MC hop sizes is set with *MCSTARTRANGE*. These ranges are adjusted every *BHCHECKRANGE* steps, in an attempt to maintain the fraction of accepted minima at *BHFACC* and the fraction of hops that end up minimizing into the same catchment basin at *BHFSAME*. The cutoffs used to determine whether two structures are the same are set with *SAMECONF*. In rare cases where the optimization to a local minimum fails, the MC step is repeated up to a number of tries set by *MAXHOPTRY*. If *FLIPNUCMC* is set, then at every Monte Carlo hop, the nucleosomes are flipped upside down relative to the fiber axis with a certain probability (this can extend the range of structures accessible in a short basin-hopping run).

At the end of the basin hopping calculation, each structure in the database can be optionally minimized (if *DBMINIMIZE* is set) to a tolerance set by *OPTOL*, and the resulting database can be re-sorted by energy if *DBSORT* is set. After this, information on the energy and helical coordinates for each structure in the database is printed out. Extra information is printed for each structure. if *PRINTENERGYPARTS, PRINTFIBERDIAM, PRINTNUCDIST* keywords are turned on. Finally, the first *NCONFIGDUMP* lowest-energy structures in the database are output to individual structure files with names set by *DUMPFILE*, and optionally (if *REPLICATESTRUCT* is set), each structure is replicated to a certain number of nucleosomes and dumped into a different file.

`examples/param.ex.basinhop` is an example file for a basin hopping calculation

5.4 DATABASEPARSE action

Parse through a database file output during a previous basin-hopping run, print out information on the structures, and output individual structure files. Note that this can be used to look at the structures found by a basin-hopping run while that run is still going.

The keywords specific to this action are: *NCONFIGSAVE, DATAFILE, DBMINIMIZE, OPTOL, ENERGYRECALC, DBSORT, SAMECONF, DBCONFIGMAXE, PRINTENERGYPARTS, PRINTFIBERDIAM, PRINTNUCDIST, NCONFIGDUMP, DUMPFILE, REPLICATESTRUCT*

Up to the first *NCONFIGSAVE* structures are read from a database file specified with *DATAFILE*. If *DBMINIMIZE* is turned on, the structures are each optimized to the a tolerance of *OPTOL*. Alternately, if *ENERGYRECALC* is turned on, the energy for each structure in the database is recalculated using the current energetic parameters (see Sec.6.3). Next, if *DBSORT* is turned on, the structures are sorted from lowest to highest energy, with any duplicates (as determined using *SAMECONF* parameters) removed from the database. Only structures with energy below *DBCONFIGMAXE* are retained in the database. The new database of structures (if any minimization, sorting, or energy recalculation was done) is dumped into the second file specified with *DATAFILE*. Information about each structure is printed out, with additional printng specified by *PRINTENERGYPARTS, PRINTFIBERDIAM, PRINTNUCDIST*. Each structure of the first *NCONFIGDUMP* in the database is dumped out into *DUMPFILE*. Replicated structures are also dumped out if *REPLICATESTRUCT* is set. Only structures with energy below *DBCONFIGMAXE* are dumped out.

`examples/param.ex.database` is an example file for a database parsing calculation

6 Calculation Details

6.1 Representation of regular fiber structure

A regular helical fiber is fully defined by the position and orientation of the first two nucleosomes, and the path of the intervening linker. In this code, the fiber is represented by the following helical coordinates:

- height per nucleosome along fiber axis (h)
- angle per nucleosome around fiber axis (θ)
- radius of fiber, from center axis to center of nucleosomes (R)
- three Euler angles (α, β, γ) giving the orientation of the first nucleosome relative to the fiber coordinate system. The fiber coordinate system is defined with the fiber axis along the z-axis and the initial nucleosome falling on the x-axis. The Euler angles are in the $z-x-z$ convention (rotation by angle α around z axis, rotation by angle β around new x axis, rotation by angle γ around new z axis). Thus, β gives the angle between the nucleosome symmetry axis and the fiber axis.

The linker DNA is represented as a discretized worm-like chain, with number of segments set by the *NSEGPERLINK* keyword. The length of the linker is set with the *LINKLEN* keyword.

The geometry of the nucleosomes is specified by providing the position of the DNA at the entry and exit (*POSP, POSM* keywords), the tangent of the DNA (*BENDTANP, BENDTANM*), and the x-axis defining the coordinate system of the DNA at the points where it enters and exits the nucleosome (*XAXP, XAXM*). Note that the DNA coordinate system has the z-axis along the DNA tangent, with the plus end (exit) tangent pointing away from the nucleosome and the minus end (entrance) tangent pointing towards the nucleosome. All DNA entry/exit geometry information is given relative to the coordinate system of the nucleosome. Note that *POSP, POSM* are treated as the position of the center of the last bound basepair of DNA. That is, the actual linker DNA is attached on one side at point $POSP+BENDTANP*\frac{bp}{2}$ relative to the first nucleosome and on the other side at point $POSM-BENDTANM*\frac{bp}{2}$ relative to the second nucleosome. The length of a basepair (bp) is set with the *BPLEN* keyword.

The *FILLBEAD* keyword can be used to specify the position and orientation of a "filler bead" of DNA attached to the nucleosome. These beads are used only for final output and visualization of the structure and are not involved in the actual calculations. Also for purposes of visualization, the length of DNA tails on either side of the fiber can be set with *NTAILSEG*.

The nucleosome attachment geometry can also be specified as an idealized spiral using the *SPIRALBENDS* keyword, which will set all the other geometry parameters, and the filler beads, automatically.

6.2 Initial fiber configuration

The relevant keywords for setting the original configuration of the fiber are: *STARTHELIX, RESTART, CHECKRESTART, RESTARTDUMP, RANDSTART, STARTHELH, STARTHELT, STARTHELR, STARTHELA, STARTHELB, STARTHELG*

By default, the fiber is set up in the regular helix corresponding to straight DNA linkers for the given nucleosome geometry and linker length. If *STARTHELIX* is set, the fiber is started

from a given set of 6 helical parameters, with the linker DNA placed in a straight line between the attachment points once the nucleosomes are set.

If the *RESTART* keyword is used, the fiber is started from an output file containing the resulting structure from a previous calculation. This keyword has a switch that allows only nucleosome positions to be read in from the supplied file, or to interpolate linker segments along a piecewise linear path defined by the linker in the supplied file. The latter can be used to restart from a structure which had a different length or number of linker segments, while still maintaining the same path of the linker.

If *CHECKRESTART* is turned on then a previous configuration will only be used if the specified file exists, and otherwise the starting structure setup will revert to defaults or *STARTHELIX*. If *RESTARTDUMP* is set, then if the file specified with *DUMPFILE* exists, the fiber will be restarted from that file rather than the other specifications.

The *RANDSTART* keyword causes the linker DNA beads to be perturbed slightly from their default positions in whatever starting structure is used. The *STARTHELH, STARTHELT, STARTHELR, STARTHELA, STARTHELB, STARTHELG* keywords will overwrite specific helical parameters regardless of how the starting structure is generated.

6.3 Energy calculations

6.3.1 Elastic energies

The linker DNA is treated as a discretized worm-like chain, with harmonic potentials for bend, twist, and stretch. The bend modulus is set with the *LP* keyword, the twist modulus with *LT*, and the stretch modulus with *LSTRETCH*. The natural twist of the DNA is set with *TWIST*, and the natural length of the linker segments is calculated based on the linker length (*LINKLEN*) and the number of segments (*NSEGPERLINK*).

6.3.2 Steric energies

The cylindrical shape of the nucleosomes for purposes of steric exclusion is set with the *STERICSHAPE* keyword. The linker DNA segments are also treated as cylinders, with radius set by *DNARAD*. The steric energies are quadratic in the overlap between cylinders, with the modulus set by *ESTERIC*.

The steric exclusion potential between nucleosome–nucleosome, linker–linker, or nucleosome–linker can be turned of with the *NONUCSTERICS, NOSEGSTERICS, NONUCSEGSTERICS* keywords.

The cylinder–cylinder overlap is calculated with the aid of tabulated minimal approach distances. The tabulations are saved in (or optionally, read from) binary files specified with *NUCCYLFILE, SEGCYLFILE, NUCSEGCYLFILE*.

Steric interactions are cut off for nucleosomes and linkers separated by more than *MAXBENDINTER* repeats.

7 DNA looping off single nucleosome

In addition to finding configurations of regular chromatin fibers, this code can also be used for optimizing DNA loops attached to a single nucleosome. More generally, it can be employed for

finding minimum energy configurations of a discretized twisted worm-like chain with the end positions, tangents, and orientations constrained.

To model a DNA loop off a single nucleosome, fix all helical parameters, with the height per nucleosome (*STARTHELH*) and angle per nucleosome (*STARTHELT*) set to 0. *MAXBENDINTER* should be set to 0 so that no replicated nucleosomes are involved in the energy calculation. *NONUCSTERICS* should be turned on to ignore the steric overlap between two nucleosomes directly on top of each other. If running a basin-hopping calculation, *SEGHOPEVERY* should be set to 1 to ensure that each hop perturbs the free DNA segments rather than the nucleosome. Finally, *NTAILSEG* should be set to 0 to get rid of DNA tails on either end.

POSP, BENDTANP, XAXP are used to set the position, tangent, and x-axis for the DNA at one end of the loop and *POSM, BENDTANM, XAXM* set the geometry at the other end of the loop.

The script `scripts/dnaloopFromPDB.py` can be used to set up end constraints for a loop held fixed between two bound basepairs on a nucleosome.

The recommended set of options for visualizing the resulting structures is,

```
scripts/beadbranch2pdb.py <filename>  -orientnucz -connectmax 2 -nbp <bpperseg> -nr
```

where `filename` is the name of the output file to be visualized and `<bpperseg>` is the number of basepairs per linker segment. This will result in a pdb file that has the nucleosome axis along the z axis, the first end of the free DNA loop along the x axis, and the nucleosome center at the coordinate origin.

An example parameter file for basin-hopping with a loop on a single nucleosome is provided in `examples/param.ex.singlenuc`.

8 Keyword Index

The code will attempt to read parameters out of a file named `param.suffix` where "suffix" is the command line argument. If no command line arguments are supplied, it will look for a file named `param`. If multiple arguments are supplied, it will read multiple parameter files in sequence.

The parameter file should have one keyword per line and must end with a blank line. All blank lines and all lines beginning with # are ignored. For the most part, the order of the lines and the capitalization of the keywords does not matter. All keywords except *ACTION* are optional. The default values for each parameter are listed below. If a keyword is supplied, then values may or may not be needed as well. Again, the required and optional value types are listed below.

Keywords and multiple values are separated by spaces.

When reading the parameter file, lines longer than 500 characters will be truncated. To continue onto the next line, add "+++" at the end of the line to be continued. No individual keyword or value should be longer than 100 characters.

Floating point numbers can be formated as 1.0, $1.1D0$, $10e-1$, $-1.0E+01$, etc., where the exponential notation specifier must be D or E (case insensitive). Integer numbers can also be specified in exponential notation without decimal points (eg: 1000 or 1E3). Logical values can be specified as T, F, TRUE, FALSE, 1, or 0 (with 1 corresponding to true and 0 to false).

The length units used for the defaults are in nm and the energy units are in kT.

- *ACTION*
  - value: 1 string of at most 20 characters; no default

  - This keyword sets the overall calculation performed by the program (see Sec.5)
  - Possible values are: GETSTRUCT, OPTIMIZE, BASINHOP, DATABASEPARSE

- *BENDTANM*
  - value: 3 floats; default 0D0 0D0 1D0
  - Specify the direction of the DNA as it comes off the bottom edge of the nucleosome. This is the minus end tangent (pointing inward towards the nucleosome), relative to the coordinate system of the nucleosome.
  - This keyword is overwritten by the SPIRALBENDS keyword

- *BENDTANP*
  - value: 3 floats; default 0D0 0D0 1D0
  - Specify the direction of the DNA as it comes off the top edge of the nucleosome. This is the plus end tangent (pointing outward from the nucleosome), relative to the coordinate system of the nucleosome.
  - This keyword is overwritten by the SPIRALBENDS keyword

- *BHCHECKRANGE*
  - value: 1 integer; default: 50
  - when running a basinhopping calculation, adjust the size of the Monte Carlo hops after the given number of steps. The hop size is adjusted to maintain a given fraction of accepted configurations (*BHFACC*), and a given fraction of hops that fall into the same local minimum basin (*BHFSAME*)

- *BHFACC*
  - values: 1 float; default 0.5D0
  - When running a basin-hopping calculation, the hop size is dynamically adjusted to maintain the fraction of accepted structures within 0.1 of the value specified here.

- *BHFSAME*
  - values: 1 float; default 0.2D0
  - When running a basin-hopping calculation, the hop size is dynamically adjusted to maintain the fraction of new configurations that fall into the same local minimum basin as the configuration before the hop. This fraction is maintained within 0.1 of the value specified here.

- *BHTEMP*
  - values: 1 float; default 1D0
  - The effective temperature used in the basinhopping calculation (as a multiple of standard temperature). Energies are scaled by this effective temperature when deciding whether to accept each new local minimum using a Metropolis Monte Carlo criterion.

- *BHTOL*
  - values: 1 float; default 1D-4
  - When running a basin-hopping calculation, specifies how tightly (in rms force) to converge the gradient-based minimization after each hop. If further minimization of the best local minima found is desired at the end, use the *DBMINIMIZE* keyword with a tolerance specified by *OPTOL*.
- *BPLEN*
  - values: 1 float; default 0.34D0
  - Length of a basepair (default units of nm). This is used to offset the edges of the linker DNA by half a basepair from the nucleosome edges at each end, and also in setting the default number of filler beads when using *SPIRALBENDS*.
- *CHECKRESTART*
  - no values
  - When restarting from an output file (*RESTART* keyword), first check to see if the file exists. If the file does not exist, then create a new structure as if the RESTART keyword was not present (from helical parameters given by *STARTHELIX* or with the straight linker structure for the given nucleosome geometry and linker length). By default, if this keyword is not present and the file for restarting is not found, then the program crashes with an error.
- *CYLINDERPTS*
  - values: 3 integers; default: 50 50 50
  - Number of points in each of 3 dimensions to use in tabulating closest approach distances for cylindrical nucleosomes. The first two dimensions are for the cosine of the angle between each nucleosome orientation and the vector between them. The last is for the dihedral angle between the nucleosomes.
- *DATAFILE*
  - value: 2 string up to 100 characters (DATAFILE, NEWDATAFILE); 2nd string is optional; default: *.data.out *.datanew.out
  - When running a DATABASEPARSE calculation or a BASINHOP calculation with *READDATAFILE* turned on, the DATAFILE string gives the name of the file containing the database of structures to be read in
  - The NEWDATAFILE string gives the name of the file into which the newly updated database of structures is saved during a BASINHOP calculation or after a DATABASEPARSE calculation if *DBSORT*, *DBMINIMIZE*, or *ENERGYRECALC* are turned on.
  - if the character * is present in the file names, then the last occurence of the * is replaced by the command-line suffix
  - DATAFILE and NEWDATAFILE may be the same for reading in and dumping out to the same database file

- *DBCONFIGMAXE*
  - 1 float; default: ∞
  - When building up a database (in the BASINHOP calculation or the DATABASEPARSE calculation with *DBSORT* turned on), do not include any configurations with energy above the give value.
  - This does not apply to reading in a database from file unless that database is then resorted with *DBSORT*
- *DBMINIMIZE*
  - no value;
  - When running a DATABASEPARSE calculation or a BASINHOP calculation minimize each of the structures in the final database to a tolerance specified by *OPTOL*
  - The structures are not re-sorted by energy after minimization unless the *DBSORT* keyword is specified
- *DBSORT*
  - no value;
  - When running a DATABASEPARSE calculation or a BASINHOP calculation, re-sort the final database from lowest to highest energy. This is done at the very end before outputting database information (after minimization of the structuresi f *DBMINIMIZE* is set.
  - If running a DATABASEPARSE calculation and *ENERGYRECALC* is not turned on, then the configurations are sorted by the energy saved in the database file as the energies are not recalculated.
- *DNARAD*
  - value: 1 float; default: 1D0
  - the steric radius of a DNA bead. Used as the cylinder radius in DNA-nucleosome and DNA-DNA steric interactions.
- *DUMPCURRENT*
  - value: 2 integers (DUMPSTEPS,DUMPNBEND), both optional; default 100 2
  - Turns on periodic dumping of structures when running an OPTIMIZE calculation. The structures are dumped after every DUMPSTEPS steps, into the file specified by the *DUMPFILE* keyword.
  - prior to dumping, the regular helical structures are replicated to have DUMPNBEND nucleosomes (DUMPNBEND must be at least 2)
- *DUMPFILE*
  - value: 1 string up to 100 characters; default: *.dump.out

  - When running an OPTIMIZE calculation with the *DUMPCURRENT* keyword turned on, this gives the filename for dumping structures periodically if the. The last instance of "#" in the specified file name is replaced by the number corresponding to the current optimization step (as a 6-digit integer).
  - When running a DATABASEPARSE calculation or at the end of a BASINHOP calculation, this is the file name for dumping each configuration in the database (without replication of the regular helix). The last instance of "#" in the file name is replaced by the index of the configuration in the database
  - if DUMPFILE contains the character *, the last instance is replaced with the command line suffix

- *ENERGYRECALC*
  - no values
  - When running a DATABASEPARSE calculation, recalculate the energy of each configuration according to the specifications in the current parameter file (instead of just using the energy already saved in the database). The keyword *DBSORT* can be used to re-sort the database according to these new energies. The final database will be output anew to the NEWDATAFILE filename specified with the *DATAFILE* keyword.
  - This keyword is unnecessary if *DBMINIMIZE* is turned on

- *ESTERIC*
  - 1 float; default: 1D3
  - Strength of steric interaction energies (nucleosome-nucleosome, nucleosome-linker, linker-linker). This is the multiplier for the harmonic function in the steric overlap.

- *EXTRAPARAMFILES*
  - 1 to 10 strings of up to 100 characters; no default
  - in addition to the current parameter file(s) (supplied as a command-line argument(s)), also read information from all the specified parameter files.
  - The last instance of "*" in each extra parameter file is replaced by the command-line argument
  - The additional parameter files have the same format as the original parameter file and can include any of the keywords described in this section. This keyword allows multiple parameter files to access the same information from a single file.
  - Files specified by this keyword will be read after those specified on the command line and may overwrite the parameters

- *FILLBEAD*
  - value: 6 floats; no default
  - Specifies a single filler bead to be used when outputting configurations. The 6 values are the position of the bead and the position of its branch, relative to the center and orientation of the bend. The branch is along the x-axis of the coordinate system associated with the DNA at that point.

  - at most 1000 such beads can be specified
  - **WARNING**: do not use together with the *SPIRALBENDS* keyword

- *FIXDIAMETER*
  - value: 1 float; no default
  - Fix the full diameter of the chain to the given value. The full diameter stretches across the regular helix between the outermost points of the nucleosomes (as approximated using cylindrical nucleosomes). It thus depends on the steric shape of the nucleosomes as well as the helix parameters. The diameter is fixed throughout the calculation by treating the helix radius as a dependent function of the other 5 regular helix coordinates.

- *FIXHELIXPARAM*
  - value: list of integers between 1 and 6 (inclusive); default: turned off
  - When running an OPTIMIZE or BASINHOP calculation, fix the listed regular helix coordinates and do not allow them to vary
  - The 6 coordinates, in order, are: height per nucleosome along the fiber axis, angle per nucleosome around helix axis, radius, and the three euler angles that define the orientation of the nucleosome relative to the helix axis. For the euler angles: the helix axis is the z-axis, the vector from the axis to the nucleosome is the x axis; roughly, beta gives the tilt of the nucleosome relative to the helix axis, alpha tells in what direction it is tilted, and gamma is the nucleosome's rotation around it's own symmetry axis

- *FLIPNUCMC*
  - value: 1 optional float; default: turned off, value 0.5D0
  - For a BASINHOP calculation, turn on random flipping upside down of the nucleosomes during the Monte Carlo hops. The supplied value gives the probability of flipping the nucleosomes at each hop.

- *LINKLEN*
  - value: 1 optional float; default: 17D0
  - Linker length for the chromatin fiber (in nm by default)

- *LP*
  - value: 1 optional float; default: 50D0
  - Bending persistence length of DNA (in nm by default)

- *LSTRETCH*
  - value: 1 float; default: 268.2927D0
  - the stretch persistence length of DNA (magnitude of the harmonic potential for stretching the bead-to-bead distance)

- *LTW*

  - value: 1 float; default: 110D0
  - the twist persistence length of the DNA (in nm by default)

- *MARKPOINT*
  - value: 1 string up to 3 characters, 3 floats (no defaults)
  - Create a marked point on the nucleosome. These points are not involved in the calculation, but are marked on each nucleosome in the output file. Can be used to visualize the nucleosome axes in PyMOL, or to see location of points of interest.
  - First string gives the point name, which will correspond to the atom name in the final pdb file created by the beadbranch2pdb.py script. String is truncated above 3 characters. The following names should not be used to avoid clashing with the DNA and nucleosome atoms in the final pdb file: A1, A2,A3, F1,F2,F3,AC,AC1,AC2,FT,CH
  - Floats are x,y,z coordinates of the marked point in the coordinate system of the nucleosome.

- *MAXBENDINTER*
  - value: 1 integer; default: 20
  - When calculating internucleosome interactions (sterics or the Gay-Berne or DiSCO potential), only calculate interactions between the 0th nucleosome (and its associated linker) and subsequent nucleosomes up to the MAXBENDINTER nucleosome with its associated linker.

- *MAXDCR*
  - value: 1 integer; default: 10
  - For the OPTIMIZE calculation. If the step along the search direction increases the energy by more than *MAXEJUMP*, then the step size will be decreased by a factor *STEPDCR*. However, if *MAXDCR* decreases are done and a lower energy still hasn't been found, the Hermitian matrix estimate will be reset and steps will be attempted in the opposite direction

- *MAXEJUMP*
  - value: 1 float; default: 1D-9
  - The maximum allowed increase in energy when selecting the step size for optimization

- *MAXHOPTRY*
  - value: 1 integer; default: 10
  - For the BASINHOP calculation, the maximum number of attempts allowed for each hop which result in the configuration failing to minimize. After the given number of attempts, the program will give up and exit with an error.

- *MAXOPTATTEMPT*
  - value: 1 integer; default: 10

- Occasionally, the BFGS optimization of the structure will fail. This is most often due to the structure hitting gimbal lock (when one of the linker segments points directly along the negative z axis), two linker beads approaching too close to one another, or a 180° angle forming between two consecutive linker segments. These problems usually arise only when working with DNA loops on a single nucleosomes or when the initial structure has enormous steric clashes.
- When this occurs, the code attempts to restart the optimization, resetting the Hessian approximation. For the single nucleosome case, the entire structure is rotated for each attempt.
- This keyword sets the maximum number of optimization restarts before the entire optimization is declared a failure.

- *MAXOPTSTEP*
  - value: 1 integer; default: 10000
  - The maximum number of steps to take in an optimization calculation, before giving up and exiting unsuccessfully
- *MAXSTEPSIZE*
  - value: 1 float; default: 1D0
  - The maximum allowed step size during an optimization calculation
- *MCSTARTRANGE*
  - value: 3 floats, last 2 optional; default: 1D0, 2π, 2D0
  - For a BASINHOP calculation, the initial ranges used in taking the Monte Carlo hops. The first range is used for moving linker beads, helix height, and radius. The second is used for randomly generating an angle by which to rotate the nucleosomes and linker segment orientations. The last is used for generating the axis around which the nucleosomes are rotated.
- *NBASINHOP*
  - value: 1 integer; default: 100
  - For a BASINHOP calculation, the number of Monte Carlo hops to carry out
- *NCONFIGSAVE*
  - value: 1 integer; default: 1000
  - For a BASINHOP or DATABASEPARSE calculation, maximum size of the database and thus the maximum number of configurations to save and output information about.
- *NCONFIGDUMP*
  - value: 1 integer; default: 1
  - For a BASINHOP or DATABASEPARSE calculation, maximum number of configurations to output as individual structure files. No configurations with energy above

- *NONUCSTERICS*
  - no values
  - Turn off nucleosome-nucleosome steric interactions
- *NONUCSEGSTERICS*
  - no values
  - Turn off nucleosome - linker segment steric interactions
- *NOSEGSTERICS*
  - no values
  - Turn off linker segment - linker segment steric interactions
- *NSEGPERLINK*
  - 1 integer; default: *LINKLEN/BPLEN*
  - Number of segments in each discretized DNA linker
  - Default gives segments corresponding to a (approximately) single basepair of DNA
- *NTAILSEG*
  - 2 integers, second optional; default: 0 0
  - Number of segments in the tails at each end of the fiber structure; used for output only
  - If only the first tail length is give, the other tail is assumed to be of the same length
- *NUCCYLFILE*
  - 1 string up to 100 characters, 1 optional logical; defaults: cylinder?.bin 1
  - File in which to store and possibly retrive the cylindrical sterics tabulation for nucleosome – nucleosome interactions
  - If present, the last instance of "*" in the file name is replaced with the command line argument
  - If present, the last instance of "?" in the file name is replaced with a string "R#H#R#H#" where the # are the radius and height for the two cylinders whose interactions are being tabulated, rounded to 2 decimal places
  - The logical input is a switch for whether or not to attempt to read in from file. If it is TRUE, attempt to read in the tabulated data from the given file (if the file exists); otherwise, recalculate the tabulation and output into the file with the given name
  - If reading tabulated data from file and the tabulated values correspond to cylinders of a different size than the ones currently used, the program print a warning, recalculate the cylinder data and save the new data in file new-filename.bin where filename is the value supplied in the parameter file.
- *NUCSEGCYLFILE*

  - 1 string up to 100 characters, 1 optional logical; defaults: cylinder?.bin 1
  - File in which to store (and possibly retrive) the cylindrical sterics tabulation for nucleosome – linker segment interactions
  - If present, the last instance of "*" in the file name is replaced with the command line argument
  - If present, the last instance of "?" in the file name is replaced with a string "R#H#R#H#" where the # are the radius and height for the two cylinders whose interactions are being tabulated, rounded to 2 decimal places
  - The integer is a switch for whether or not to attempt to read in from file. If it is TRUE, attempt to read in the tabulated data from the given file (if the file exists); otherwise, recalculate the tabulation and output into the file with the given name
  - If reading tabulated data from file and the tabulated values correspond to cylinders of a different size than the ones currently used, the program print a warning, recalculate the cylinder data and save the new data in file `new-filename.bin` where filename is the value supplied in the parameter file. Note that the size of the linker segment cylinders depends on the *LINKLEN* and *NSEGPERLINK* parameters

- *OPTPRINTFREQ*
  - value: 1 integer; default: 10
  - when running a optimization, print the current energy and RMS force every so many steps

- *OPTOL*
  - value: 1 float; default: 1D-4
  - the convergence tolerance for an OPTIMIZE calculation. The optimization stops when the RMS force reaches this value.
  - this is also the convergence tolerance used with *DBMINIMIZE* for a DATABASEPARSE calculation
  - If running a basinhopping calculation then the optimization tolerance is set instead by *BHTOL* except for any final minimization with *DBMINIMIZE*.

- *OUTFILE*
  - value: string up to 100 characters; default: *.out
  - When running a GETSTRUCT or OPTIMIZE calculation, this is the file into which the final configuration is output (without replicating the regular factor)
  - When running a BASINHOP calculation, the best configuration found so far is output into this file throughout the calculation
  - If the string provided has a * in it, then the last * in the string is replaced with the command-line argument used when running the program.

- *POSM*
  - value: 3 floats; default: 0,1D0,-1D0

  - the position of the minus end of the DNA coming off the nucleosome, relative to the nucleosome center, using the nucleosome coordinate system
  - This keyword is not used if *SPIRALBENDS* is set

- *POSP*
  - value: 3 floats; default: 0,1D0,-1D0
  - the position of the plus end of the DNA coming off the nucleosome, relative to the nucleosome center, using the nucleosome coordinate system
  - This keyword is not used if *SPIRALBENDS* is set

- *PRINTFIBERDIAM*
  - no values
  - At the end of the calculation, print out the overall diameter of the fiber (stretching out to the furthest point of the nucleosomes). This depends on the cylindrical shape of the nucleosomes.
  - If running a BASINHOP or DATABASEPARSE calculation, the diameter of each structure in the database is printed out.

- *PRINTNUCDIST*
  - no values
  - At the end of the calculation, print out the distances between the cylindrical shells of the first nucleosome and of each subsequent nucleosome. This depends on the cylindrical shape of the nucleosomes.
  - If running a BASINHOP or DATABASEPARSE calculation, the distances for each structure in the database are printed out.

- *PRINTENERGYPARTS*
  - no values
  - At the end of a GETSTRUCT or OPTIMIZE calculation, print out the individual energy components for the chain structure. When running a BASINHOP or DATABASEPARSE calculation, the energy components are printed out for each individual structure in the final database.
  - The energy components (if not using DiSCO or Gay-Berne potentials) are, in order: (1) elastic bend energy, (2) twist energy, (3) stretch energy, (4) steric overlap energy between nucleosomes, (5) steric overlap energy between linker segments, (6) steric overlap energy between nucleosomes and linker segments.

- *RANDSTART*
  - 1 float; default: 1D-4
  - When initializing the structure, randomize the position and orientation of the DNA linker beads by an amount equal to the specified value.

- If this keyword is not set, then the linker is just placed as a straight linke from the plus end of one nucleosome to the minus end of the other

- *READDATAFILE*
  - no value
  - When running a BASINHOP calculation, begin by reading in a database of structures from the file specified by *DATAFILE* (if it exists). All subsequently found configurations are added on to this database. If the file does not exist, the calculation starts from scratch.

- *REPLICATESTRUCT*
  - values: integer (NBENDREPLIC), optional string (REPLICFILE); default string: *.replic.out
  - at the very end of the calculations, replicate the regular fiber structure to have NBENDREPLIC nucleosomes and output the result in the file REPLICFILE
  - The last instance of "*" in REPLICFILE is replaced by the command line argument
  - If running a BASINHOP or DATABASEPARSE calculation, each structure in the final database is replicated and output into REPLICFILE, where the last instance of "#" in REPLICFILE is replaced by the index of the structure in the database.

- *RESTART*
  - value: 1 string up to 100 characters, 1 optional integer; default: *.out 2
  - Start the calculation using an output file from a previous run of the program. If the filename contains a *, the last instance of the * is replaced by the command line argument.
  - The regular helix parameters of the structure read in from the restart file can be individually overwritten with the *STARTHELH, STARTHELT, STARTHELR, STARTHELA, STARTHELB, STARTHELG* keywords.
  - If *CHECKRESTART* is turned on then only restart from file if the file exists, otherwise start as normal. If *CHECKRESTART* is not turned on, then the program crashes with an error if the file does not exist.
  - the optional integer is a switch: if it is equal to 0, then only the bend positions and orientations are read from the output file, and the beads are filled in between bends in a straight-line fashion. If it is equal to 1 then the bead positions are used to define a piecewise linear curve and beads are placed equidistant along that curve. If it is 2, the entire chain configuration (including bead positions) is obtained from the output file.
  - If the switch is 2, then the number of linker segments must equal what it was in the restart file.
  - **Warning**: if obtaining bead positions from the output file (switch=1 or 2), the nucleosome geometry needs to be identical to that used in generating the output file to begin with. Otherwise, there may be strange behavior with no error messages.

- *RESTARTDUMP*
  - no values

  - Restart from the file specified by *DUMPFILE* if it is present. Otherwise start as normal (either from *RESTART* or *STARTHELIX* or with a straight-linker structure. This is useful for restarting an OPTIMIZE calculation after a crash.
  - Note: the last instance of "*" in *DUMPFILE* is replaced by the command line argument, but any instance of "#" is not replaced for restarting purposes

- *RNGSEED*
  - value: 1 integer; default: 0
  - Seed for the random number generator.
  - If the value is 0 then the generator is seeded off the time, and the results will be different on each run (unless the runs are started within a millisecond of each other)
  - If the value is -1 then the seed is based on the last 5 characters in the command line argument (not counting SPACE,various parentheses,",'). A unique seed is produced for all possible sets of 5 characters.
  - If the value is -2 then the seed is based on the last 4 characters of the command line argument, and the millisecond time
  - Otherwise, the integer gives the random generator seed to use and the run should be repeatable with the exact same results
  - The random number generator is only used for a BASINHOP calculation, or if *RANDSTART* is set.

- *SAMECONF*
  - value: 2 floats (SAMEECUT, SAMEDISTCUT); default 0.5D0 1D0
  - when building up a database of distinct low-energy structures (for a BASINHOP calculation or with *DBSORT*), these parameters define what it means for structures to be considered the same
  - 2 structures are the same if their energy is within SAMEECUT of each other and the distance between the centers of the second nucleosomes relative to the first is within SAMEDISTCUT*LP and the angle necessary to rotate the orientation of the first nucleosome from one structure to the other is within SAMEDISTCUT and the individual linker beads are within SAMEDISTCUT*LP in the two structures.

- *SEGCYLFILE*
  - 1 string up to 100 characters, 1 optional logical; defaults: cylinder?.bin 1
  - File in which to store (and possibly retrive) the cylindrical sterics tabulation for linker segment – linker segment interactions
  - If present, the last instance of "*" in the file name is replaced with the command line argument
  - If present, the last instance of "?" in the file name is replaced with a string "R#H#R#H#" where the # are the radius and height for the two cylinders whose interactions are being tabulated, rounded to 2 decimal places

  - The logical value is a switch for whether or not to attempt to read in from file. If it is TRUE, attempt to read in the tabulated data from the given file (if the file exists); otherwise, recalculate the tabulation and output into the file with the given name
  - If reading tabulated data from file and the tabulated values correspond to cylinders of a different size than the ones currently used, the program print a warning, recalculate the cylinder data and save the new data in file `new-filename.bin` where filename is the value supplied in the parameter file. Note that the size of the linker segment cylinders depends on the *LINKLEN* and *NSEGPERLINK* parameters

- *SEGHOPEVERY*
  - 1 integer; default: -1
  - When running a BASINHOP calculation, how often to take a Monte Carlo hop that moves the linker segments only rather than the nucleosome positions
  - if the given value is less than 0, all hops move the nucleosomes
  - if the given value is 1, all hops move the linker segments only
  - Otherwise, move the linker segments every so many hops.

- *SPIRALBENDS*
  - 1 float (SPLEN), 1 integer(FILLINBEADS), 4 floats (SPRADIUS, SPHEIGHT, SPTWIST, SPTWIST0), all optional; default: 49.64D0, *[1], 4.18D0, 2.39D0, *[2], 0D0
  - Set up the geometry of the nucleosomes as an idealized spiral
  - SPLEN is the length of DNA bound on the nucleosome (in nm, by default)
  - FILLINBEADS is the number of filler beads bound on the nucleosome. (*[1]) By default, this is set to SPLEN/*BPLEN*+1 (i.e, 147 for the default length bound)
  - SPRADIUS is the radius of the spiral, out to the center of the DNA helix (in nm, by default)
  - SPHEIGHT is the height per turn of the spiral (in nm by default)
  - SPTWIST is the twist of the DNA bound on the nucleosome. (*[2]) By default this is set to be the same as the natural twist of the linker DNA (as set by the *TWIST* keyword)
  - SPTWIST0 sets the phase of the DNA twist; this should not matter except for visualizing the output
  - This keyword overwrites any other nucleosome geometry specifications, specifically those given by *POSP, POSM, BENDTANP, BENDTANM, XAXM, XAXP*.
  - **WARNING**: do not use together with the *FILLBEAD* keyword

- *SPIRALHAND*
  - 1 integer (1 or -1); default: -1
  - When using idealized spiral for the nucleosome geometry (with *SPIRALBENDS* keyword), this sets the handedness of the spiral.
  - 1 is a right-handed spiral; -1 is a left-handed spiral

- *STARTHELIX*
  - value: 6 floats, no default
  - start the chain from a regular helical conformation with the specified parameters
  - the 6 values are, in order: height per nucleosome along fiber axis, angle per nucleosome around fiber axis, radius of fiber, then 3 euler angles giving the orientation of each nucleosome relative to the fiber coordinate system (where the z axis is the fiber axis and the x axis goes from the center of the fiber to the center of each nucleosome)
  - Values from this keyword is overriden by values from the following keywords: *RESTART, RESTARTDUMP, STARTHELH, STARTHELT, STARTHELR, STARTHELA, STARTHELB, STARTHELG*
  - If none of the keywords *STARTHELIX, RESTART, RESTARTDUMP* are provided, the default initial conformation is the one with straight linkers for the given nucleosome geometry and linker length.
- *STARTHELA*
  - value: 1 float; default: no default
  - set the α Euler angle for the nucleosome relative to the fiber axis in the initial regular helix structure
  - If *RESTART* is used, then the regular helix coordinates are obtained from an output file and then this particular coordinate is overwritten with the given value. This keyword also overwrites the corresponding value set by *STARTHELIX*.
- *STARTHELB*
  - value: 1 float; default: no default
  - set the β Euler angle for the nucleosome relative to the fiber axis in the initial regular helix structure
  - If *RESTART* is used, then the regular helix coordinates are obtained from an output file and then this particular coordinate is overwritten with the given value. This keyword also overwrites the corresponding value set by *STARTHELIX*.
- *STARTHELG*
  - value: 1 float; default: no default
  - set the γ Euler angle for the nucleosome relative to the fiber axis in the initial regular helix structure
  - If *RESTART* is used, then the regular helix coordinates are obtained from an output file and then this particular coordinate is overwritten with the given value. This keyword also overwrites the corresponding value set by *STARTHELIX*.
- *STARTHELH*
  - value: 1 float; default: no default
  - set the starting height per nucleosome for regular helix coordinates.

- If *RESTART* is used, then the regular helix coordinates are obtained from an output file and then this particular coordinate is overwritten with the given value. This keyword also overwrites the corresponding value set by *STARTHELIX*.

- *STARTHELR*
  - value: 1 float; default: no default
  - set the starting fiber radius for regular helix coordinates.
  - If *RESTART* is used, then the regular helix coordinates are obtained from an output file and then this particular coordinate is overwritten with the given value. This keyword also overwrites the corresponding value set by *STARTHELIX*.

- *STARTHELT*
  - value: 1 float; default: no default
  - set the starting angle per nucleosome around the fiber axis for regular helix coordinates.
  - If *RESTART* is used, then the regular helix coordinates are obtained from an output file and then this particular coordinate is overwritten with the given value. This keyword also overwrites the corresponding value set by *STARTHELIX*.

- *STEPDCR*
  - value: 1 float; default: 0.1D0
  - For the gradient-based optimization, when trying to pick a step size, if the energy increases by more than *MAXEJUMP* then the step size is decreased by factor *STEPCRD*, up to *MAXDCR* times.

- *STERICSHAPE*
  - value: 2 floats, second one optional; defaults: 5.2D0 5.5D0
  - This specifies the size of the cylindrical nucleosomes to use for steric exclusion calcualtions. First value is the radius of the cylinder; second value is its height (in nm, by default)

- *TENSION*
  - value: 1 value; default: 0
  - External tension applied to the chain. Positive tension stretches the fiber and negative tension compresses it. Default units: kT/nm (per nucleosome)

- *TWIST*
  - value: 1 float; default: 1.7667
  - the twist density of the DNA in units of per length (default length units are nm)

- *VERBOSE*
  - no values
  - Turn on extra output for optimization calculations

- *XAXM*
  - value: 3 floats; default: 1,0D0,0D0
  - The position of the x-axis of the DNA coordinate system at the minus end of the nucleosome (relative to the nucleosome coordinate system). Note that the center of the minus end DNA coordinate system is set by *POSM*, while the z-axis of this coordinate system is set by *BENDTANM*
  - This vector will be normalized and orthogonalized to *BENDTANM* before being used
- *XAXP*
  - value: 3 floats; default: 1,0D0,0D0
  - The position of the x-axis of the DNA coordinate system at the plus end of the nucleosome (relative to the nucleosome coordinate system). Note that the center of the plus end DNA coordinate system is set by *POSP*, while the z-axis of this coordinate system is set by *BENDTANP*
  - This vector will be normalized and orthogonalized to *BENDTANP* before being used

References

[1] J. Nocedal and S. Wright, *Numerical Optimization*, Springer series in operations research, Springer, 1999.

[2] D. J. Wales, *Energy Landscapes*, Cambridge University Press, Cambridge, 2003.