ChromMC

It is a Python package for folding chromatin fibers based on information about proteins interactions with chromatin using the Monte Carlo method. User can save trajectory in pdb file and create a contact map (similar to Hi-C map). The package gives the possibility to continue the simulation (option -i).
Details of the calculation and possible usage is available in the publication Tuszynska, Bednarz, Wilczynski, **Black chromatin is indispensable for accurate simulations of *Drosophila melanogaster* chromatin structure.** Only information about the usage of scripts is provided here.

Requirements

To run this package following dependencies are required (pickle, json and msgpack methods are used to save the last step of simulation that allow the continuation of starting simulation, while matplotlib is used in HiCreconst.py script to create a contact map):

- python (3.6.8)
- numpy (1.17.2)
- pickle
- json (2.0.9)
- msgpack
- matplotlib (3.3.4)

Usage

There are 2 scripts to use:

- chroMC.py - run the simulation for chromatin structure prediction
- HiCreconstr.py - create a contact map for chosen frames from the chromatin trajectory

Chromatin simulation:
To see all options that are available for chroMC.py just run:

```
>python chrMC.py -h
Usage: chrMC.py [<options>]

Options:
  -h, --help          show this help message and exit
  -p OUT_STR          An output filename with MC trajectory
  -i IN_STR           An input state (pickled, json or msgpack file)
  -s STEPS            Number of steps for simulation (default 100000)
  -l CH_LENGHT        Lenght of the chain (default 512)
  -b BINDERS          Number of binders (default 256)
```

```
  -e SAVE            Save every .... accepted step (default 100
  -n REGULAR_BSITES  Full path to the file names with regular binding
sites,
                     separated by comma. Mandatory!
  -a LAMIN_BSITES    Full path to the file names with lamin binding
sites,
                     separated by comma.
  -d DUMP_METH       Method to save the intermediate state. Default is
                     msgpack, possible to choose: msgpack, pickle, json.
```

Mandatory options are:

-s - number of steps for running simulation

-l - length of the chain in beads (the program uses reduced representation of chromatin, one bead represents a user defined number base pairs, Kbp in our simulation, described in the publication)

-b - number of binders, if there are more than one types of binders separate it by a comma

-n - path to the file name/s with binding sites of binders on the chromatin chain. The file name has to have one column with numbers that represent binding sites of defined binders, the files number should be the same as a number of binder types

An example command to produce simulation, described in the publication:

```
>python3 chrMC.py -p chr_ins_pol_k27_black.out -s 5000 -l 1000 -b
1000,1000,1000,1000 -n  example_inp/
regular_bsites_H3K27Me3_L2_5kbp_1000.txt,example_inp/
regular_Insul_5.0kbp_2L_1000.txt,example_inp/
regular_PolII_1620_CS_5kbp_1000.txt,example_inp/
GSE22069_Black_5.0kbp_4andMore.txt
```

All needed files are available in the example_inp directory.
The above command creates 4 files (placed in the example_out directory):

1. output_name.out (in above command output_name is chr_ins_pol_k27_black) describes energy and contains four columns: 1. Monte carlo step, 2. accepted step, 3. energy, 4. gyration radius
2. output_name.pdb the trajectory in pdb format, each frame is separated by the MODEL line
3. output_name_lamin.pdb - separate pdb with the lamin layer
4. output_name.msgpack - file for running the longer trajectory starting from the last step of current trajectory (use the option -i instead of -l -b and -n flags)

Contact map generation:

To see all options that are available for chroMC.py just run:

```
>python HiCreconstr.py
Usage: HiCreconstr.py  [<options>]

Options:
  -h, --help     show this help message and exit
```

```
  -f FIRST_TRAJ  The first file with trajectory in pdb format
  -b             Only BOU and LAM atoms will be considered
  -s START       The first step to make a map
  -l END         The last step to make a map
  -t STEP        The step size to collect structures and make a HiC map
```

To produce a contact map for first 5 frames from the trajectory file run:

```
>python HiCreconstr.py -f  example_out/chr_ins_pol_k27_black.pdb -l 5
```

The contact map (HiC.png) is in the example_out directory.