`Brownian_Dynamics.m`, version 1.1 – A MATLAB Function for Computing Brownian Dynamics Trajectories

1 Files in the ZIP package

The ZIP package contains the following files:

- `Brownian_Dynamics.m`
- `license.txt`
- `README.pdf`
- `k.mat`, `z.mat`, `zz.mat`, and `nodes.txt` in the subdirectory `input` (exemplary input files)
- `u_saved.mat` and `AA_saved.mat` in the subdirectory `output` (exemplary output files)

2 A simple example of computing the Brownian dynamics trajectory

In this example, the MATLAB script `Brownian_Dynamics.m` is used to generate a 400-picosecond Brownian dynamics trajectory of the DNA tweezer presented in Figure 2 of the accompanying research paper. More details on the script are available in the help text,

```
>> help Brownian_Dynamics
```

All calculations in this document were performed using MATLAB R2013a.

Step 1. Copy `Brownian_Dynamics.m`, `k.mat`, `z.mat`, `zz.mat`, and `nodes.txt` to the current working directory.

Note. Data in the input files are in different units. See the help text of `Brownian_Dynamics.m` for details. Node 9999999 in the file `nodes.txt` is an auxiliary node, which will be excluded in the Brownian dynamics calculation and post-processing in the next two steps.

Step 2. Run the script,

```
>> rng(0);
>> Brownian_Dynamics(212, 4, 400);
```

The first argument is the number of nodes in the finite element model (212), and the second and third ones are the time step size (4 picoseconds) and the simulation time (400 picoseconds), respectively. The Brownian dynamics trajectory is sampled every ten steps and hence consists of ten frames. When the calculation terminates, two output files `u_saved.mat` and `AA_saved.mat` are saved in the current working directory. Also saved is a temporary file `matrices.mat`.

Note. It is optional to specify a random seed using the function `rng` before calculating the Brownian dynamics trajectory. The files in the subdirectory `output` were generated using a random seed of zero.

Step 3. Process the two output files `u_saved.mat` and `AA_saved.mat`, which contain three variables `u_saved`, `u_saved_f` (in `u_saved.mat`), and `AA_saved` (in `AA_saved.mat`).

The solution $\mathbf{x}$ at frame i of the governing equation for Brownian dynamics,

$$\mathbf{Z}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{f}(t)$$

is stored in `u_saved(:,i)`. For example, the displacement of node 210 at frame 8 ($t = 320$ picoseconds) is

```
>> disp(u_saved(1255:1257,8))
   -0.4728
   -0.3408
    0.3328
```

(in Angstroms) in the three translational degrees of freedoms and

```
>> disp(u_saved(1258:1260,8))
    0.0940
    0.1072
   -0.0558
```

(in radians) in the three rotational degrees of freedoms. In general, the displacement of node j at frame i is in `u_saved(j*6-5:j*6,i)`.

The rigid-body translation of the finite element model at frame i is calculated from `u_saved_f(:,i)`. For example, the rigid-body translation (in Angstroms) at frame 8 is given by,

```
>> tmp=reshape(u_saved_f(:,8),6,212)';
>> disp(mean(tmp(:,1:3))')
   -0.0156
    0.8278
   -0.5147
```

Finally, the rigid-body rotation of the finite element model at each frame about its center of mass at frame i is stored as `AA_saved(i*3-2:i*3)`. For example, the rotation matrix corresponding to the rigid-body rotation of the finite element model from the initial orientation at $t = 0$ to that at frame 8 is given by

```
>> disp(AA_saved(:,22:24))
    0.9996   -0.0194    0.0185
    0.0200    0.9992   -0.0334
   -0.0178    0.0337    0.9993
```