Documentation for liquid network simulations

Zuben Scott

Last updated August 22, 2024

The code in this package will run a simulation of a tubular network under constant tension undergoing length minimizing rearrangements. Growth of new tubules from existing edges occurs as a Poisson process and nodes diffuse in addition to feeling the length minimization forces.

1 Compilation Instructions

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

- a compiler capable of handling Fortran90. The code has been tested with the gfortran and compiler. The default compiler is gfortran.
- BLAS and LAPACK libraries installed in a place where the compiler knows to look for them
- Optionally: Matlab to visualize output data

The code has been tested on Ubuntu Linux.

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 `minnetBD.exe` will appear in the main directory.

2 Usage Instructions

To run the program in the main directory, type:

```
./minnetBD.exe suffix > outputfile.out
```

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 4. 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.

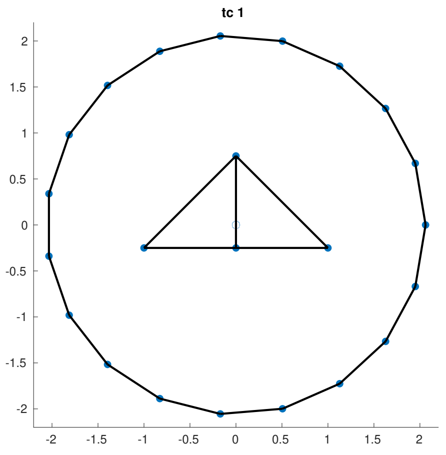
3 Example for a Quick Start

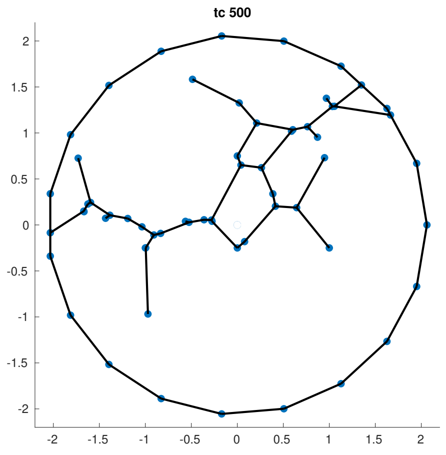
3.1 Example 1

An example parameter file (`param.example1`) is provided. This will run a minimal network simulation for a small triangular network composed of for fixed nodes. New edge growth and diffusion of non-fixed nodes are turned on. The whole network is enclosed within a roughly circular boundary. Run the example with

```
./minnetBD.exe example1 > outputfile.out
```

Use the script `../scripts/checkexample.m` to watch a movie of the snapshots:





4 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.**??**)
  - Possible values are: BROWNDYN
- *BDPRINTEVERY*
  - value: 1 integer; default 1
  - How often to print output (network length, max length, etc) to screen and to the output file (given by *OUTFILE*)
- *BDSTEPS*
  - value: 1 integer; default 1000
  - Number of Brownian dynamics steps to run
- *CATA*
  - value: 1 float; default 1D-8
  - Poisson rate for catastrophe of growing nodes (begin to retract)
- *CENTERENCLOSE*
  - value: none
  - Centers the input network and encloses it with a rough circle of stationary boundary edges. Edge length of boundary edges chosen to be less than half max edge length in initial network.
- *DELT*
  - value: 1 float; default 1D-3
  - Time-step for the Brownian dynamics
- *DIFF*

  - value: 1 float; default 1D-3
  - Diffusivity of free nodes

- *DX*
  - value: 1 float; $\max\left(\sqrt{D \cdot dt}, b \cdot dt\right)$
  - Minimal length within which rearrangements can occur. Automatically set by code, but can also be changed by user.

- *EXTRAEDGEFACT*
  - value: 1 integer; default 100
  - Extra edges to include in network structure (e.g. allowing for growth.) If initial network is 10 edges, EXTRAEDGEFACT of 100 means maximum network size during simulation is $10 * 100 = 1000$ edges

- *EXTRANODEFACT*
  - value: 1 integer; default 100
  - Same idea as above, but for nodes

- *EXTRATRACKFACT*
  - value: 1 integer; default 1000
  - Same idea as above, but for the tracking of nodes.

- *FIXNODE*
  - value: 1 integer
  - Forces initial network to permanently fix provided node index. Provide multiple lines starting with FIXNODE to fix multiple nodes.

- *FIXNODEFROMNETFILE*
  - value: none
  - Forces initial network to fix nodes based on input .net file. Nodes with a “T” after their node position are fixed

- *FUSEPROB*
  - value: 1 float; default 1D0
  - Fusion probability of a growing node once it hits an edge, must between 0 and 1

- *GASTD*

  - value: 1 float; default 1D-2
  - Width of normal distribution for angles of growth. Defaults to essentially perpendicular growth.

- *GROWTH*
  - value: 1 float; default 1D0
  - Poisson rate for growth events

- *GVEL*
  - value: 1 float; default 1D0
  - Velocity of growing nodes

- *MAXBRANCH*
  - value: 1 integer; default 3
  - Maximum number of branches per node. Code currently only supports up to degree 3 nodes

- *MINNETDIM*
  - value: 1 integer; default 2
  - Dimension of space that network lives in. Code currently only supports dimension of 2, due to method of looking for intersections. A rough approach to dimensionality can be taken by tuning the fusion probability.

- *MINNETFILE*
  - value: 1 string; default *.net
  - File containing network structure. Proper format can be seen by looking at example1.net, and networkObj code.

- *MOB*
  - value: 1 float; default 1D0
  - Mobility of nodes, determines magnitude of length minimization forces

- *NOBROWN*
  - value: none
  - Turn off random Brownian forces (equivalent to zero temperature dynamics)

- *NFIX*
  - value: 1 integer

  - If using RANDFIXNODES, sets number of nodes to be fixed randomly.

- *NPIN*
  - value: 1 integer; default 0
  - Number of nodes from initial network to temporarily pin.

- *OUTFILE*
  - value: 1 string, up to 100 characters; default: *.out
  - Currently outputs step number followed by network length, may change over time.
  - Any * in the file name will be replaced by the command-line argument (suffix)

- *PIN*
  - value: 1 float; default 0
  - Poisson rate of pinning events.

- *PINFILE*
  - value: 1 string; default '*.pin.snap.out'
  - What file to store node pinning information at each snapshot.

- *RANDFIXNODES*
  - value: none
  - If provided, randomly fix NFIX nodes from network. Note NFIX must be provided.

- *RMULT*
  - value: 1 float; default 2D0
  - When CENTERENCLOSE is provided, this sets the radial boundary at RMULT*(max radial distance of initial nodes)

- *RNGSEED*
  - 1 integer; default: 0
  - seed for random number generator
  - value of 0 will seed with system time in milliseconds
  - value of -1 will use the last 5 characters in the suffix
  - value of -2 will use the last 4 charactes in the suffix and the millisecond time
  - other positive value: the seed is used directly for repeatable simulations (should be positive)

- *SNAPSHOTFILE*
  - value: 1 string; default: *.snap.out
  - File for dumping out snapshots. Can also be specified within SNAPSHOTS keyword.
- *SNAPSHOTS*
  - 1 optional integer, 1 optional string, 1 optional logical; defaults: 1, *.snap.out, false
  - Dump snapshots over the course of the calculation
  - integer: how often to dump snapshots; string: snapshot file (* is replaced with suffix); logical: append rather than rewriting the snapshot file
- *STARTSNAPSLATE*
  - 1 integer; default 0
  - only start outputting snapshots of network after provided frame number. Useful for long simulations where you are interested in fine-grained snapshots once network has reached steady-state.
- *TRACKFILE*
  - value: 1 string; default '*.track.snap.out'
  - Snapshot file for tracking nodes.
- *UNPIN*
  - value: 1 float; default 1D0
  - Unpinning rate.
- *VERBOSE*
  - value: 1 logical; default: false
  - Print extra output. Mostly for debugging purposes