Documentation for dynamicNetwork: code for a dynamic network with fusion and fission

K.B. Holt, E. F. Koslover

Last updated September 17, 2024

The code in this package will run a dynamic simulation of a network that is capable of fusion and fission. Each edge of the network is a straight segment between 2 nodes and represents (eg) a mitochondrial unit. Each node is an end-point of an edge (no lone nodes). Forces act on nodes. Fission happens at nodes of degree > 1. Fusion happens between degree-1 nodes or a degree-1 and a degree-2 node.

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 compiler.
- Optionally: Matlab to visualize output data

The code has been tested on Ubuntu Linux 22.04.

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

2 Usage Instructions

To run the program in the main directory, type:

```
./dynnetwork.exe suffix > stdout.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 `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.

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.

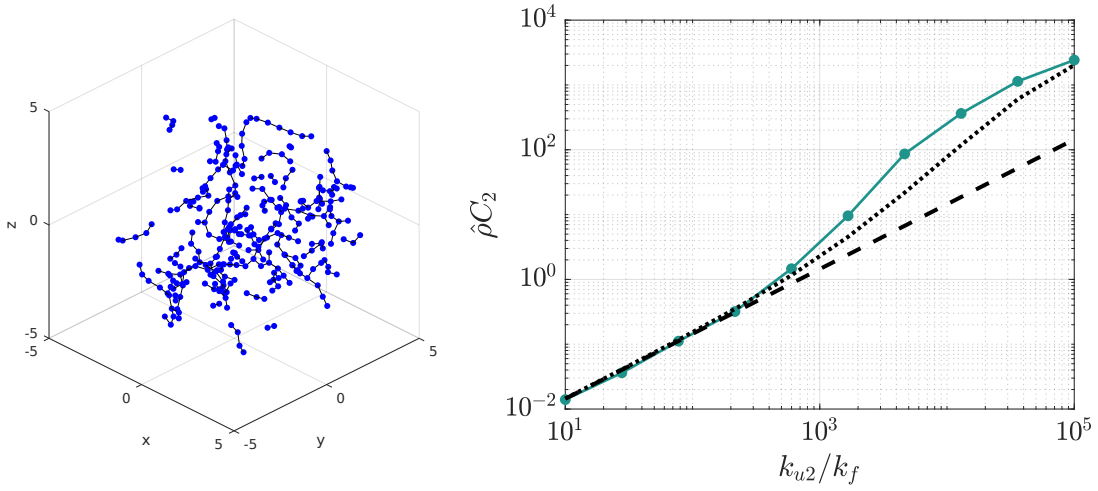
3 Example for a Quick Start

3.1 Example 1

A set of example parameter files (`param.example_0_0_j`) are provided in the `param_files` folder. Each parameter file will run a dynamic network simulation with 250 edges for 2x10^6 simulation time steps and a different value of the fusion rate. To run the examples, navigate to the `param_files` directory and run each example with

```
../dynnetwork.exe example_0_0_j > stdout.example_0_0_j
```

If your machine has multiple processors, the simulations can be run in parallel. Snapshots of the network structure will be written to the file (`example_0_0_j.snap.out`) every 1000 simulation steps. Each fusion and fission event will be recorded in the file (`example_0_0_j.ffevents.out`). Use the script `checkexample.m` (in the scripts folder) to visualize a given network structure over time, and the resulting plot of $\hat{\rho}C_2$ once all 10 simulations are finished. To run this script successfully, you must also add `github.com/lenafabr/networktools/NetworkObj.m` to your path. The resulting visualizations should look something like this:



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).

Simulation Setup

- *ACTION*
  - value: 1 string of at most 20 characters; no default
  - Possible values are: RUNDYNAMICS
- *BDSTEPS*
  - value: 1 integer; default 1000
  - Number of Brownian dynamics steps to run
- *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)
  - Negative values < -2: undefined, do not use (may give identical simulations or not; could be compiler dependent).
- *DIM*
  - value: 1 integer; default 3
  - Dimensionality of the space in which the network is embedded
  - Currently not set up for use in DIM $\neq 3$
- *USEGRID*
  - value: 1 logical; default true
  - Whether to use a spatial grid method to speed up pairwise distance calculations
- *USEEDGEGRID*
  - value: 1 logical; default false

  - Whether to use a spatial grid method to further speed up steric interaction calculations between edges. Requires USEGRID set to true

- *USENODEGRID*
  - value: 1 logical; default false
  - Whether to use a spatial grid method to speed up pairwise distance calculations for fusion between nodes. Requires USEGRID set to true

- *GRIDSPACES*
  - value: 1 or DIM integers; default 10
  - Number of grid cubes in the spatial grid along each spatial dimension

Mechanics

- *ESTR*
  - value: 1 float; default 1000
  - Stretch modulus for individual segments

- *BENDMOD1*
  - value: 1 float; default 2.0
  - Bending modulus for degree 2 nodes. This value also sets the angular sensitivity for tip-tip fusion unless ALPHA1 is set

- *BENDMOD2*
  - value: 1 float; default 6.0
  - Bending modulus for degree 3 nodes. This value also sets the angular sensitivity for tip-side fusion unless ALPHA2 is set

- *THETA0*
  - value: 1 float; default $\pi/3$
  - Equilibrium "bond angle" for degree 3 nodes

- *STERICMOD*
  - value: 1 float; default 1D4
  - Energy prefactor for steric interactions

- *STERICRAD*
  - value: 1 float; default 0.1
  - Radius at which edges begin to sterically repel

- *USEOUTOFPLANE*

  - value: 1 logical ; default: false
  - Whether to incorporate an explicit out-of-plane bending energy term at degree-3 junctions (also used to define fusion rates).

- *BENDMODPLANE*
  - value: 1 float ; default: 0.6
  - Bending stiffness prefactor for out-of-plane deviations at degree-3 junctions.

- *ALPHAPLANE*
  - value: 1 float; default 2×BENDMODPLANE
  - Fusion sensitivity parameter for fusion between a degree-1 node and a degree-2 node corresponding to the out-of-plane energy.

Fusion and Fission

- *FISSRATE*
  - value: 1 float ; default: 0.0
  - Fission rate, per time, per fissable node (degree>1)

- *D3FISSMULT*
  - value: 1 float; default 1.5
  - The multiplier defining the likelihood of fission at degree-3 junctions relative to degree-2 junctions.

- *RECHARGERATE*
  - value: 1 float; default -1.0
  - Rate at which nodes return to an active-fusion state following fission. Values less than zero will set this rate to ∞, meaning nodes are always allowed to re-fuse during the next time step after a fission event occurs.

- *FUSERATE1*
  - value: 1 float ; default: 0.0
  - Fusion rate, per time, per fuseable pair of degree-1 nodes

- *FUSERATE2*
  - value: 1 float ; default: 0.0
  - Fusion rate, per time, per fuseable pair of degree-1, degree-2 nodes

- *CONTACTRAD*
  - value: 1 float; default: 0.15

  - Fusion is allowed when nodes are separated by a distance 2×CONTACTRAD or less.

- *ALPHA1*
  - value: 1 float ; default: 2×BENDMOD1
  - Fusion sensitivity parameter for fusion between degree-1 nodes

- *ALPHA2*
  - value: 1 float ; default: 2×BENDMOD2
  - Fusion sensitivity parameter for fusion between a degree-1 and degree-2 node

Dynamics

- *DELT*
  - value: 1 float; default 1D-4
  - Time-step for the dynamics

- *FRICT*
  - value: 1 float; default: 1.0
  - Frictional coefficient for nodes

- *KT*
  - value: 1 float; default: 1.0
  - Temperature (k_bT) for Brownian dynamics

Network and Boundary Setup

- *MITOLEN*
  - value: 1 float; default: 0.5
  - Ground-state edge length

- *MAXNEDGE*
  - value: 1 integer; default: 250
  - Maximum allowed number of edges in the network (code will exit with error if this is exceeded)

- *MAXNNODE*
  - value: 1 integer; default: 500
  - Maximum allowed number of nodes in the network (code will exit with error if this is exceeded). Safest to set MAXNNODE to 2×MAXNEDGE

- *USERANDOMFRAGMENTEDNETWORK*

  - value: 1 logical; default: true
  - Whether to start the simulation from a random network generated by placing isolated edges (all nodes will initially be degree-1) uniformly throughout the spherical volume

- *NETFILE*
  - value: 1 string; default: *.net
  - Input network file (full format with NODE lines, etc) used to start the simulation. If USEFRAGMENTEDNETWORK is set to true, this network file will not be used.
  - Any * in the file name will be replaced by the command-line argument (suffix)

- *CONFSPHERE*
  - value: 1 to 2 floats; defaults: -1D0, 1D4
  - CELLRAD1, ECONF (confinement radius, energy prefactor)
  - Confine the network inside a sphere of radius CELLRAD1. The confinement energy is quadratic with prefactor ECONF. These two parameters may alternatively be specified on their own lines

- *NPLANE*
  - value: 1 integer ; default: 0
  - If using planes to confine the network (rather than the default spherical boundary), the number of planes making up the boundary.

- *PLANEFILE*
  - value: 1 string; default: *.pl
  - Input file used to build the planes defining the confining boundary of the network. NPLANE must match the number of planes specified in the file
  - Any * in the file name will be replaced by the command-line argument (suffix)

Outputting Data

- *PRINTEVERY*
  - value: 1 integer; default: 1
  - Sets the step interval at which a status update should print to the screen

- *SNAPSHOTFILE*
  - value: 1 string; default: *.snap.out
  - File for dumping out snapshots. Can also be specified within SNAPSHOTS keyword.

- *REMODELINGFILE*
  - value: 1 string; default: *.ffevents.out

  - File for dumping out fission/fusion event records.

- *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
  - Snapshot file contains multiple snapshots of network structure and other info. Should be read with `parseDynNetworkSnapshots.m` script

- *STARTSNAPSTEP*
  - value: 1 integer; default: 1
  - Simulation step to begin saving snapshots

- *VERBOSE*
  - value: 1 logical; default: false
  - Print extra output.

Yeast Model

- *DOYEAST*
  - value: 1 logical; default false
  - Whether to tether nodes to the cell boundary, creating a network geometry in which edges are constrained to the inner surface of a spherical shell.

- *YEASTCONF*
  - value: 1 float; default 100
  - The confinement energy tethering the nodes to the cell boundary is quadratic with prefactor YEASTCONF.

- *YEASTBINDRANGE*
  - value: 1 float; default 2.0
  - Maximum distance from the cell boundary at which a node can become tethered.

- *YEASTONRATE*
  - value: 1 float; default 1.0
  - Rate for each node to become tethered to the cell boundary when within YEASTBINDRANGE.

- *YEASTOFFRATE*
  - value: 1 float; default 1.0
  - Rate for each node to fall off of the cell boundary when tethered.

Diffusion on the Network

- *NSPECIES*
  - value: 1 integer ; default: 0
  - Number of different types of particles to propagate along the network via diffusion (uses Finite-Volume-Method).
- *STARTDIFFSTEP*
  - value: 1 integer; default: 1
  - Simulation step to begin propagating diffusing particles through the network.
- *DIFF*
  - value: NSPECIES floats ; default: 0.0
  - Diffusion constant for each of the NSPECIES types of diffusing particles.
- *NSUBSTEPS*
  - value: 1 integer ; default: 0
  - The number of steps which should be taken during the diffusion calculation for each timestep used by the rest of the simulation. (i.e. value 10 means a timestep of DELT/10 will be used to perform the diffusion calculation)
- *PRODRATE*
  - value: 1 or NSPECIES floats ; default: -1.0
  - Rate at which new particles of each type are added to producing edges.
- *DECAYRATE*
  - value: 1 or NSPECIES floats; default -1.0
  - Rate at which diffusing particles of each type decay on the network.
- *NUMTRIGGERUNITS*
  - value: 1 integer ; default: 0
  - The number of edges at which diffusing particles should be produced.
- *UNIQUETRIGGERS*
  - value: 1 logical ; default: false
  - Whether to randomly select a unique set of producing edges for each type of diffusing particle, or use the same set for all types.
- *FIXEDGES*
  - value: 1 logical; default false
  - Whether to hold the concentration of the set of producing edges at a fixed value of 1.

5 .net network file structure

Here is an example network file for use to begin a simulation from a pre-built network structure. Note that edge lengths need not be specified.

```
# file defining network structure
# made with matlab NetworkObj output

# list of node indices and xyz positions
NODE 1        -0.4002000000         0.0100000000         0.0100000000
NODE 2         0.0000000000         0.0200000000         0.0100000000
NODE 3         0.5000000000         0.0004000000         0.0010000000
NODE 4         0.7803000000         0.0000000000         0.2800000000
NODE 5        -0.2002000000         0.3500000000        -0.0100000000
NODE 6        -2.1000000000         0.2600000000        -1.0300000000
NODE 7        -2.1003040000        -0.0400000000        -1.0000100000
# list of edge indices with the 2 nodes they link. Edge length
EDGE 1 1 2         0.4000000000
EDGE 2 2 3         0.5000000000
EDGE 3 3 4         0.4000000000
EDGE 4 2 5         0.4000000000
EDGE 5 6 7         0.3000000000
```

6 .pl planes file structure

Here is an example planes file for use to begin a simulation in a non-spherical geometry. Each plane segment is specified as a vertex with (x,y,z) coordinates and two direction vectors. The magnitude of each direction vector sets the length of the plane in that direction. The plane will be "oriented" in the direction given by the cross product of the unit vectors corresponding to the two direction vectors. Nodes which are "below" the plane (as defined by its orientation) will experience a force normal to the plane in the direction of its orientation.

```
#this file contains information about plane segments which set
#the geometry for a dynamic network simulation
#First plane
VERTEX 1 0.000 0.000 0.000
DIR1 1 16.000 0.000 0.000
DIR2 1  0.000 8.000 0.000
#2nd plane
VERTEX 2 0.000 0.000 0.000
DIR1 2  0.000 0.000 4.000
DIR2 2 16.000 0.000 0.000
#3rd plane
VERTEX 3 0.000 0.000 0.000
DIR1 3 0.000 8.000 0.000
DIR2 3 0.000 0.000 4.000
#4th plane
VERTEX 4 16.000 8.000 4.000
DIR1 4   0.000 -8.000  0.000
DIR2 4 -16.000  0.000  0.000
#5th plane
VERTEX 5 16.000 8.000 4.000
DIR1 5 -16.000  0.000  0.000
```

```
DIR2 5   0.000  0.000 -4.000
#6th plane
VERTEX 6 16.000 8.000 4.000
DIR1 6   0.000  0.000 -4.000
DIR2 6   0.000 -8.000  0.000
```