Documentation for particledynamics1D: code for particle dynamics in a 1D domain

S. S. Mogre, E. F. Koslover

Last updated March 23, 2021

The code in this package will run a simulation of particle dynamics in a one-dimensional domain.

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. 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 `partdynamics1D.exe` will appear in the main directory.

2 Usage Instructions

To run the program in the main directory, type:

```
./partdynamics1D.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 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

An example parameter file (`param.example1`) is provided. This will run a simulation for the dynamics of a system with 2 different types of protein-carrying particles. The particle types start at opposite ends of a 1D domain, moving towards the other end at a constant velocity. Particle type 1 undergoes stochastic reversals of direction. Particles of different types can fuse with each other. Run the example with

```
./partdynamics1D.exe example
```

Use the script `checkexample.m` to visualize the movement and protein flux into one end of the domain. (NOT IMPLEMENTED)

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 are set with respect to the domain length (*DOMLEN*). Time units are set using a combination of velocity (*VEL*) and domain length.

- *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: RUNDYNAMICS
  - Input format: ACTION string
  - 'string' can contain up to 20 characters.
- *ACIDIFICATIONRATE*
  - type: float; size: *NTYPE*; default: 0

- Rate of acidification/enzyme acquisition (used for vATPase pump model)
- Input format: ACIDIFICATIONRATE k_1 k_2 ... k_{NTYPE}
- k_i is the acidification rate for particle type i

- *BUFLEN*
  - type: float; size: 1; default: 0
  - Buffer length to determine an encounter event between particles
  - Input format: BUFLEN value
- *DIFFCONST* (not implmented)
  - type: float; size: *NTYPE*; default: 0
  - Diffusion coefficient for each particle type. If only one value is entered, all particles types will have the same diffusion coefficient.
  - Input format: DIFFCONST D_1 D_2 ... D_{NTYPE}
  - D_i is the diffusion coefficient for particle type i.
- *DOMLEN*
  - type: float; size: 1; default: 1
  - Length of the simulation domain.
  - Input format: DOMLEN value
- *DELT*
  - type: float; size: 1; default: 1D-1
  - Time-step for dynamics
  - Input format: DELT value
- *DOFUSION*
  - type: logical; size: 1; default: F
  - Determines whether particles undergo fusion.
  - Input format: DOFUSION value
- *KFUSE*
  - type: float; size: $NTYPE \times NTYPE$; default: 0
  - Rate of fusion between particle types.
  - Input format: KFUSE i $k_{i,1}$ $k_{i,2}$ ... $k_{i,NTYPE}$
  - $k_{i,j}$ is the fusion rate between particle types i and j
  - Multiple lines should be used for different values of i.
- *KPROD*

- type: float; size: *NTYPE*; default: 0
- Rate of production for all particle types. If only one value is entered, all particles are produced at that rate.
- Input format: KPROD k_1 k_2 ... k_{NTYPE}
- k_i is the rate at which particle type i is produced.

- *KREV*
  - type: float; size: *NTYPE*; default: 0
  - Rate of direction reversal for all particle types.
  - Input format: KREV k_1 k_2 ... k_{NTYPE}
  - k_i is the rate at which particle type i reverses movement direction.

- *KSTOP*
  - type: float; size: *NTYPE*; default: 0
  - Rate of pausing for all particle types. If only one value is entered, all particle types are assigned that pause rate.
  - Input format: KSTOP k_1 k_2 ... k_{NTYPE}
  - k_i is the rate at which particle type i pauses moving.

- *MAXNFUSE*
  - type: integer; size: *NTYPE*; default: 1
  - Number of fusion events needed to saturate particle types. If only one value is entered, all particle types are assigned that value.
  - Input format: MAXNFUSE N_1 N_2 ... N_{NTYPE}
  - N_i is the number of fusion events needed to saturate a particle of type i.

- *NPART*
  - type: integer; size: *NTYPE*; default: 1
  - Initial number of particles of each type. If only one value is entered, all particle types are assigned that value.
  - Input format: NPART N_1 N_2 ... N_{NTYPE}
  - N_i is the initial number of particles of type i.

- *NPROT*
  - type: integer; size: *NTYPE*; default: 1
  - Number of protein types within each particle type. If only one value is entered, all particle types are assigned that value.
  - Input format: NPROT N_1 N_2 ... N_{NTYPE}
  - N_i is the number of protein types in particle type i.

- *NREG*
  - type: integer; size: 1; default: 0
  - Number of special regions within the domain.
  - Input format: NREG value
- *NSTEPS*
  - type: integer; size: 1; default: 0
  - Number of simulation steps.
  - Input format: NSTEPS value
- *NTRIALS*
  - type: integer; size: 1; default: 0
  - Number of independent simulation trials.
  - Input format: NTRIALS value
- *NTYPE*
  - type: integer; size: 1; default: 0
  - Number of particle types. Maximum 5 types allowed.
  - Input format: NTYPE value
- *OUTFILE*
  - 1 string; default: *.out
  - File to which output is written. Can also be specified in OUTPUT.
- *OUTPUT*
  - 1 optional integer, 1 optional string; defaults: 1, *.out
  - Information about writing output to file.
  - integer: how often to write output; string: output file (* is replaced with suffix)
- *PFUSE*
  - type: float; size: $NTYPE \times NTYPE$; default: 0
  - Probability of fusion between particle types. If this option is used, probabilities will be used instead of rates to determine fusion events.
  - Input format: PFUSE i $p_{i,1}$ $p_{i,2}$ ... $p_{i,NTYPE}$
  - $p_{i,j}$ is the fusion rate between particle types i and j
  - Multiple lines should be used for different values of i.
- *PRAD*
  - type: float; size: $NTYPE$; default: 1D-2

- Rate of direction reversal for all particle types. If only one value is entered, all particles are assigned that reversal rate.
- Input format: PRAD r_1 r_2 ... r_{NTYPE}
- r_i is the radius of particles of type i.

- *PRINTEVERY*
  - type: integer; size: 1; default: 0
  - How often simulation status is displayed on the console.
  - Input format: PRINTEVERY value
- *PROTCONV*
  - type: float; size: $NPROT \times NPROT \times NTYPE$; default: 0
  - Rates of conversion between different protein types within different particle types.
  - Input format: PROTCONV i j $k_{i,j,1}$ $k_{i,j,2}$ ... $k_{i,j,NTYPE}$
  - $k_{i,j,k}$ is the conversion rate for protein j to protein i in particle type k.
  - Multiple lines should be used for different values of i and j.
- *REGPOS*
  - type: float; size: $NREG$; default: -1
  - Positions of special regions within the domain. A negative value results in equally spaced regions.
  - Input format: REGPOS x_1 x_2 ... x_{NREG}
  - x_i is the position of region i.
- *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)
- *SNAPFILE*
  - 1 string; default: *.snap.out
  - File to which snapshot of particles is written. Can also be specified within SNAPSHOTS.
- *SNAPSHOTS*

- 1 optional integer, 1 optional string; defaults: 1, *.snap.out
- Dump snapshots over the course of the simulation.
- integer: how often to dump snapshots; string: snapshot file (* is replaced with suffix).

- *STARTPOS*
  - type: float; size: $NTYPE$; default: 0
  - Position at which particles of different types are produced.
  - Input format: STARTPOS x_1 x_2 ... x_{NTYPE}
  - x_i is the starting position for particles of type i.
- *STARTPROT*
  - type: float; size: $NTYPE \times NPROT$; default: 0
  - Initial protein content in particles.
  - Input format: STARTPROT i $c_{i,1}$ $c_{i,2}$ ... $k_{i,NPROT}$
  - $c_{i,j}$ is the amount of protein of type j within particle type i
  - Multiple lines should be used for different values of i.
- *STARTDIR*
  - type: float; size: $NTYPE$; default: 0
  - Initial transport state of particles. 1 = moving in the positive x-direction. -1 = moving in the negative x-direction. 0 = stationary.
  - Input format: STARTDIR a_1 a_2 ... a_{NTYPE}
  - a_i is the starting state for particles of type i.
- *USEPFUSE*
  - type: logical; size: 1; default: F
  - Determines whether fusion probability is used instead of a fusion rate.
  - Input format: USEPFUSE value
- *VEL*
  - type: float; size: $NTYPE$; default: 0
  - Velocity of different particle types.
  - Input format: VEL v_1 v_2 ... v_{NTYPE}
  - v_i is the velocity particles of type i.