BDpack

BDpack is a package to numerically calculate the configurational evolution of polymeric solution using Brownian dynamics simulation of bead-spring micro-mechanical model. The algorithm used in BDpack incorporates high-fidelity and computationally efficient calculation of hydrodynamic interactions (HI) and excluded volume (EV) forces. BDpack is written in parallel by employing message passing interface (MPI) for distributed-memory-architectures.

The aim of this project is to write the codes self descriptive, documented, yet very efficient to enable its applicability and development. To this end, the codes are written in modular fashion using Fortran.

Installation and Running

Dependencies

The program uses intel math kernel library (MKL), i.e., levels 1-3 of MKL_BLAS and MKL_LAPACK to perform the large scale linear algebra. In particular, GBMV is used for banded matrix-vector products. SYMV and SYMM are used for cases in which one of the operands is a symmetric matrix, TRMV and TRMM are used when one of the operands is a triangular matrix, and GEMV, GEMM are called for the general matrix-vector and matrix-matrix multiplication, respectively. The QR factorization involved in the block Lanczos algorithm is done by GEQRF from the MKL_LAPACK followed by ORGQR. The Cholesky decomposition is performed with the POTRF routine.

MKL can be accessed free of charge for individual use. Please conduct the intel free software tools for more information. Due to extensive application of MKL, the package works best if `mpif90` is used with intel fortran compiler. However, gnu compiler can be used as well.

To install the package, the user has to make sure that MKL is installed properly. The root directory of MKL should be automatically saved in the macro `MKLROOT` which can be checked by typing `echo $MKLROOT`. Otherwise, it will be specified explicitly.

It is important to compile BDpack with the same compilers which are used for making MKL libraries `mkl_blas95_lp64.a` and `mkl_lapack95_lp64.a`. This can be done very easily by recompiling the modules `mkl95_blas` and `mkl95_lapack` which are located in `interfaces/blas95` and `interfaces/lapack95` of MKL root directory, respectively.

Step by step installation

BDpack has been tested on several machines using OpenMPI MPI fortran compiler, `mpif90`. For complete installation of the package, the following steps should be performed:

1- Make sure that `mpif90` is installed on your machine:

```
$ mpif90 --version
```

If you wish to use intel compiler, use:

```
$ export OMPI_FC=ifort
```

and for gnu compiler, use:

```
$ export OMPI_FC=gfortran
```

2- If `MKLROOT` is not recognized, it should be explicitly specified in `make.inc`.

3- The final step is to compile the codes which are located in the `src` directory. Simply type in the `src` directory:

```
$ make
```

Running BDpack projects

1- Upon proper installation of the program, the executable file `BDpack.exe` is built in `projects` directory.

2- The program can get started in any directory which contains `BDpack.exe` and `input.dat` files and `data` directory by using the standard run command of the OpenMPI:

```
$ mpirun -np x BDpack.exe
```

with x replaced by the number of processes.

3- A sample form of `input.dat` is provided in `projects` directory. The parameters are specified with their names followed by a separator `:` and their values:

```
driver        : dilute_bs
#--------------------------------------
# Configuration parameters
#--------------------------------------
nchain        : 2
nseg          : 10
tplgy         : Comb
Arms          : 2 2 4
nseg_ar       : 2
Rel-Model     : Rouse
...
```

4- All the output data will be provided in the directory `data`.

Description of different drivers

The current version of BDpack provides the tools to simulate polymers in infinitely dilute solutions using bead-spring model (**dilute_bs** driver). However, the codes for semi-dilute/concentrated solutions of bead-spring model and infinitely dilute solutions of bead-rod chains have also been implemented and will be provided in the later versions of the package.

In what follows, the parameter description of different drivers will be given in detail. In case the parameters are not specified by user, their default values will be set in the program. However, the parameters with U as their default value should be specified by user for proper behavior of the program. Note that the logical parameters are specified with **TRUE** or **FALSE**.

Infinitely dilute solution using bead-spring model (dilute_bs)

The governing equations and numerical algorithms for infinitely dilute solution which is used in BDpack can be found in our recent article; Saadat and Khomami (2014).

Configuration parameters

Parameter	Description	Type	Default
nchain	total number of chains	integer	U
nseg	total number of segments of one chain	integer	U
tplgy	topology of the chain, currently **Linear** is implemented and tested. **Comb** topology is implemented but not tested.	character	Linear
Arms	(Only **Comb** topology) the first entry is the number of arms followed with the bead at which the arms are grafted. For example, 3 2 4 5 specifies a chain with 3 arms which are located at beads 2, 4, and 5	integer	U
nseg_ar	(Only **Comb** topology) number of segments in arms	integer	U
Rel-Model	the model which accounts for dimensionless longest relaxation time: **Rouse**, **Zimm**, **Tanner**, and **Self** are the options. If **Self** is selected, its value should be provided as the next entry	character	Rouse

Stochastic differential equation (SDE) parameters

Parameter	Description	Type	Default
initmode	denotes the initial state of the configuration, **st** is for starting from a known configuration (70% of maximum extension in equilibrium condition or from {q.st.dat, CoM.st.dat} files in non-equilibrium condition. **rst** is for restarting from {q.rst.dat, CoM.rst.dat} files. The files should be located in `data` directory	character	st
tend	end time in units of chain relaxation time	real	10.0
tss	steady state time in units of chain relaxation time	real	5.0
trst	(Only **rst** initmode) time prior to restarting the simulation in units of chain relaxation time	real	0.0
dt	the range of time step size, initial and final values and the method of spacing (**Linear** or **Log**)	real/character	0.01
tol	the convergence criteria for second corrector step of predictor corrector scheme	real	1.e-4
nroots	number of roots used in constructing look-up table	integer	10^6
PrScale	A factor >=1 which increases the precision of look-up table	integer	1
CoM	tracking center of mass	logical	FALSE
CoHR	tracking center of hydrodynamic resistance	logical	FALSE

Force parameters

The force-extension relation is obtained based on flexibility of the macromolecules in solution. For flexible chains, Hookean,, FENE, ILCCP, and RWS are the alternatives. ILCCP stands for Cohen's Pade approximation to inverse Langevin function and RWS is the random walk spring model proposed by Underhill and Doyle (2004). For semi-flexible chains, approxiamtion models to real worm-like chain are used, e.g., WLC_MS, WLC_UD proposed by Marko and Siggia (1995) and Underhill and Doyle (2006), respectively.

Parameter	Description	Type	Default
SPR-	spring force-extension relation. Options are: **Hookean**, **FENE**, **ILCCP**, **RWS**		

Force	for flexible chain and **WLC_MS** or **WLC_UD** for semi-flexible chains	character	Hookean
Truncation	The **FENE** and **ILCCP** force vs. extension can be truncated at specific value of relative extension. The first entry is the method for truncation which can be **None**, **Linear**, or **Cnst**. The second entry is the relative extension $0<qr<1$ after which the truncation is applied.	character/real	None
N_Ks	(Only **RWS** and **WLC_UD** models) number of Kuhn steps per spring	real	U
b	squared maximum dimensionless length of the springs	real	U
EXT-Force	if external force is applied to the ends of the chain (constant force ensemble) followed with the value of the force if the first entry is TRUE	logical/real	FALSE

(Non)equilibrium parameters

Parameter	Description	Type	Default
Flow-Type	the type of external applied flow: **1**: equilibrium, **2**: shear, **3**: uniaxial extension, **4**: biaxial extension, **5**: planar extension	integer	1
nWi	total number of Weissenberg numbers (*Wi*) to be considered.	integer	1
Wi	the range of *Wi*, initial and final values and the method of spacing (**Linear** or **Log**)	real/character	0.0

Hydrodynamic interaction (HI) parameters

Parameter	Description	Type	Default
hstar	h* the dimensionless hydrodynamic interaction strength	real	0.0
HITens	(Only if h* is nonzero) hydrodynamic interaction tensor. Options are **RPY** (Rotne-Prager-Yamakawa), **Zimm** (equilibrium pre-averaged Oseen), **OB** (Oseen-Burgers), and **RegOB** (regularized Oseen-Burgers given by Zylka and Öttinger (1989))	character	RPY
DecompMeth	(Only if h* is nonzero) decomposition method. Options are **Cholesky**, **Lanczos** (Krylov subspace method), or **Chebyshev**	character	Cholesky
ncols	(Only if h* is nonzero) the number of coloumns in a block of block decomposition methods	integer	1
m	(Only if h* is nonzero and DecompMeth is **Lanczos**) the first two entries are initial value and upper bound of iteration number in (block)Lanczos method. The third entry specifies if the value of *m* is fixed at initial value in decomposition algorithm	integer/logical	3 15 FALSE
L	(Only if h* is nonzero and DecompMeth is **Chebyshev**) the first two entries are initial value and upper bound of iteration number in (block)Chebyshev method. The third entry specifies if the value of *L* is fixed at initial value in decomposition algorithm	integer/logical	2 20 FALSE
AveIter-rep	(Only if h* is nonzero and DecompMeth is **Lanczos** or **Chebyshev**) if the value of iteration needs to be recorded	logical	FALSE
errormin	(Only if h* is nonzero and DecompMeth is **Lanczos** or **Chebyshev**) the minimum error used in iteration procedure	real	1.e-2

upfactr	(Only if h^* is nonzero and DecompMeth is **Lanczos** or **Chebyshev**) the update frequency of initial value of iteration number in units of number of columns (ncols)	integer	50

Excluded volume (EV) parameters

Parameter	Description	Type	Default
EVForceLaw	the method for calculation of EV force. Options are **NoEV**, **Gauss**, and **LJ**. The latter two options stand for Gaussian potential proposed by Prakash and Öttinger (1999)	character	NoEV
zstar	(Only if EVForceLaw is **Gauss**) the EV potential strength for soft Gaussian potential	real	U
dstar	(Only if EVForceLaw is **Gauss**) the EV potential broadness for soft Gaussian potential. As the second entry, the method of calculating dstar is specified. If **Self** is chosen, the first entry is the value of dstar. If **Kumar** is selected, the first entry specifies parameter *K* in $d^* = Kz^{*1/5}$ proposed by Kumar and Prakash (2003)	real/character	1.0 Kumar
LJ-Par	(Only if EVForceLaw is **LJ**) there are 4 entries which are dimensionless $\epsilon, \sigma,$ truncation and cutoff radii, respectively. Note that ϵ is nondimensionalized with k_BT	real	U
minNonBond	(Only if EVForceLaw is not **NoEV**) the minimum distance between the beads along the chain to count the EV potential	integer	1

Output parameters

The program gives the user the option to extract configurational information, e.g., end-to-end distance, average segmental length, rheological material functions from the simulation. Also, the radius of gyration and the average cosine of the neighboring springs can also be obtained automatically using BDpack. Note that if SDE parameters CoM and CoHR are TRUE, the diffusivity of center of mass and center of hydrodynamic resistance are calculated in the program. Any further post processing is possible by using the dumped files {R.equil.dat, CoM.equil.dat} for equilibrium or {R.final.dat, CoM.final.dat} for nonequilibrium condition. The file which starts with R stores the bead to center of mass of all beads of all chains and the other file stores the center of mass position vectors of all chains. The user also has the option to dump configuration of the chains at specific strains.

Parameter	Description	Type	Default
frm-rt-rep	the rate of reporting the time passed in units of relaxation time	real	0.1
frm-rt-pp	the rate of recording data for post processing in the program in units of relaxation time	real	0.002
frm-rt-rst	the rate of recording restart files in units of relaxation time	real	0.1
frm-rt-dmp	the rate of dumping data in units of relaxation time	real	0.1
Conf-anal	if the configurational information is desired	logical	TRUE
Timer-rep	if timing of the program is desired	logical	FALSE
Rg	if the calculation of the radius of gyration is desired	logical	FALSE
cosTh	if the calculation of the cosine of the neighboring springs is desired	logical	FALSE
Dumpstr	if dumping configurational info at specific strains is desired. If the first entry is TRUE, the next entry is the number of strains followed by the	logical/integer/real	FALSE

	value of strains		