

Documentation for the Stochastic Immersed Boundary Method (SIB)

Paul J. Atzberger

Package Description:

This release is meant for instructional purposes and as a basic demonstration of the Stochastic Immersed Boundary (SIB) method. This document describes the package along with tutorial which demonstrates how to perform simulations. The numerical codes were written in ANSI C. Matlab scripts are also included for generation and visualization of data files.

Related Papers:

Please cite when using the codes

A Stochastic Immersed Boundary Method for Fluid-Structure Dynamics at Microscopic Length Scales, Atzberger, P.J., Kramer, P.R., and Peskin, C.S., J. Comp. Phys., Vol. 224, Iss. 2, (2007).

```
@article{StochasticImmersedBoundaryMethod,
title = "A stochastic immersed boundary method for fluid-structure dynamics at microscopic length scales",
author = "Paul J. Atzberger and Peter R. Kramer and Charles S. Peskin",
journal = "Journal of Computational Physics",
volume = "224",
number = "2",
pages = "1255 - 1292",
year = "2007",
doi = "https://doi.org/10.1016/j.jcp.2006.11.015"
}
```

For additional papers, see <http://atzberger.org/>

Installation and Organization of the Package:

The installation is self-contained in the subtree from the root file directory. Special scripts do not need to be executed to install the package. The package is organized as follows. Matlab scripts are given which generate input data files for the SIB codes and for visualization of output files from the SIB codes. The C codes contain the implementation of the numerical methods used in the SIB method. For each specific application the model is generated from a Matlab script with prefix 'run_generate_*'. An application specific force law and data processing routines are also specified in application C codes starting with the prefix 'applCodes_*' and 'systemEnergyForce_*'. Make files are also included to aid with compilation. The package is most easily used for UNIX and LINUX platforms. For Windows users, 'Cygwin' is a good alternative to use when working with the package. The best approach to get familiar with the codes is to follow one of the tutorials to setup a model.

Polymer Model (Example)

In the directory ./examples/polymer/ are files for a model to simulate the stochastic dynamics of polymer chains incorporating hydrodynamic interactions and thermal fluctuations. To model the

elasticity of the polymer, both stretching and bending force terms are incorporated. In general, setting up such a model for the SIB codes requires the following basic steps:

- (i) Generate SIB data files for the initial configuration of control points and initial velocity of the fluid.
- (ii) Generate application specific data files, such as the polymer bonding topology, elastic stiffness, and bending rigidity.
- (iii) Generate a master parameter file for the SIB codes which includes the numerical simulation parameters and a base file name which is to be used for all input and output data files. The different data file types are distinguished by their extensions.
- (iv) Implement in the C codes with prefix 'applCodes_*' and 'systemEnergyForce_*' the application specific codes for the model, such as the force law for the immersed structures as represented by the collection of elementary control points.
- (v) Compile the application specific codes and link them with the SIB numerical package.

We now discuss in more detail how to carry out these steps for the polymer model. To generate the necessary data files in steps (i) - (iii) for the polymer model, a Matlab script was implemented. To execute this script run in Matlab the file "run_generate_polymer_1.m." In the script file the numerical and model parameters can be adjusted. For a description of these parameters see the comments in the code and the section on parameter descriptions.

Step (iv) has already been carried out for the polymer model mentioned above with implementation in the file 'applCodes_polymer.c' and 'systemEnergyForce_polymer.c'.

The codes can be compiled and linked with the SIB numerical package by using the 'makefile' script which for the polymer model is executed by the command "make polymer." This compiles and links the appropriate force law and data processing routines into the SIB numerical package. This script then creates the application specific executable named 'run_polymer_1_run01' (extension may be appended with .exe depending on the platform).

To run the executable requires an input file name for the master parameter file. For the polymer model this is: `./polymer_1_run01/polymer_1.params`. For convenience a c-shell script can be generated to run the codes. In the present example a script is generated with the name: `"start_polymer1_run01.csh"`.

To visualize the results of a simulation with the SIB codes a Matlab script was implemented named "run_visualize_polymer_1.m". This plots the polymer configuration at each time step, computes statistics for the model, and saves PNG image files for generation of animations of the simulation.

More information on how to use each of these codes and how to make modifications for your specific application needs can be found in the comments of the script files and c codes. This basic model and tutorial are meant to serve as a starting point to get familiar with some of the features of the SIB codes.

Parameter Description:

params.idum : Seed for the random number generator.

params.numRuns :	Number of times to execute the codes (for example to generate samples for Monte-Carlo methods).
params.numTimeSteps :	Number of time steps to perform.
params.saveSkip :	Number of time steps to skip between saving data to disk.
params.restartSaveSkip :	Number of time steps to skip between saving a complete data set which can be used to restart the codes.
params.numVelWaveNumber :	Number of Fourier modes in each dimension to resolve for the fluid velocity field.
params.numConcGrids :	Number of concentration fields.
params.numConcGridPoints :	Number of mesh points in each direction to resolve in the concentration fields. In present implementation should be consistent with params.numVelWaveNumber.
params.numGeoConcGrids :	Number of concentration fields restricted to geometrically defined sub-domain.
params.baseFilename :	Master file name base to use in generating all file names for input and output.
params.L :	Length of the domain box.
params.deltaT :	Time step.
params.KB :	Boltzmann's constant in units used in the simulation.
params.T :	Temperature of the system.
params.mu :	Dynamic viscosity of the fluid.
params.rho :	Density of the fluid.
params.sparseConcFlag :	Flag to indicate if a sparse subset of the concentration field data is to be written to disk.
params.concIndicesFilename :	Filename of file in which to write the sparse index data.
params.sparseGeoConcFlag :	Flag to indicate if a sparse subset of the geometrically restricted concentration field data is to be written to disk.
params.geoConcIndicesFilename :	Filename of file in which to write the sparse index data.
params.sparseVelFlag :	Flag to indicate if a sparse subset of the velocity field is to be written to disk.
params.velIndicesFilename :	Filename of file in which to write the sparse index data.
params.sparseForceFlag :	Flag to indicate if a sparse subset of the structure generated force field on the fluid is to be written to disk.
params.forceIndicesFilename :	Filename of file in which to write the sparse index data.
params.sparsePressureFlag :	Flag to indicate if a sparse subset of the structure generated pressure field of the fluid is to be written to disk.
params.pressureIndicesFilename :	Filename of file in which to write the sparse index data.
params.sparseFiberFlag :	Flag to indicate if a sparse subset of the structure configuration data is to be written to disk.
params.fiberIndicesFilename :	Filename of file in which to write the sparse index data.
params.debugLevel :	Level of debugging information to be displayed.
params.verboseFlag :	Level of information to be displayed during execution of the codes.

Package File List and Descriptions:

Main Package:

concCodes.h

concCodes.c:	Codes which implement a finite volume method for a concentration field over the entire fluid lattice.
concCodes_version4_0.h concCodes_version4_0.c:	Codes which implement a finite volume method for a concentration field over the entire fluid lattice.
discreteFFT.h discreteFFT.c:	Codes which implement the Discrete Fast Fourier Transform.
fiberCodes.h fiberCodes.c:	Codes which implement procedures related to the immersed fibers and particles.
gaussianConc4.h gaussianConc4.c:	The main core of the stochastic immersed boundary method which calls lower level procedures to update all the fibers, fluid, concentration fields and also to handle input and output of data.
gaussianConc4Types.h:	This header file defines abstract data types used in the main core of the stochastic IB code which is implemented in gaussianConc4.c.
generalFileIO.h generalFileIO.c:	Codes which implement file operations performed in the read / write procedures.
geoCellConcCodes.h geoCellConcCodes.c:	Codes which implement a finite volume method for a concentration field restricted to a subset of the fluid lattice.
paramParse.h paramParse.c:	Codes which implement file parsing procedures for the parameter files.
randomVariates.h randomVariates.c:	Codes which implement random number generators used in the code.
systemEnergyForce4_base.h systemEnergyForce4_base.c:	Codes which serve as a base for application specific energy and force calculations.
units1.h:	Header file for physical parameters used in the codes.
vectorArray.h vectorArray.c:	Codes which implement procedures related to processing and calculations involving vectors.
velCodes.h velCodes.c:	Codes which implement procedures related to the fluid velocity fields.
Examples:	
polymer/ applCodes_polymer.h applCodes_polymer.c:	Codes which implement application specific procedures for initialization of the code, read / write of data, and other misc. procedures.
run_generate_polymer_1.m:	Matlab codes which generate application specific data files.
makefile:	The standard file used with "make" command which builds executables.
readUserExtras4_Polymer4.m:	Matlab code which reads application specific data file.
readUserExtrasMisc4_Polymer4.m:	Matlab code which reads application specific data file.
systemEnergyForce_polymer.h	

systemEnergyForce_polymer.c: Application specific codes which implement the force and energy computing procedures.

userExtras4_Polymer4.h: Header file defining application specific abstract data types.

run_visualize_polymer_1.m: Matlab code which visualizes application specific data.

writeUserExtras4_Polymer4.m: Matlab code which write application specific data structures.

Matlab:

genGeoCells.m: Matlab code which generates data structures for a concentration field on a subset of the fluid lattice.

gen_fpackSparseCleanScripts.m

gen_fpackSparseFiberScripts.m

gen_fpackSparseScripts.m

gen_scpScript.m

gen_scpScriptUpdate.m: Matlab code which generates scripts useful in updating data files on a remote server and packaging files for transfer.

gen_sparseIndices_sliceZ.m: Matlab code which generates indices used to specify a sparse planar slice of the fluid lattice.

plot3DVectorFieldSlice.m: Matlab code to plot a vector field.

plotArrows.m: Matlab code to plot a vector field.

plotCells.m

plotCells2.m: Matlab code to plot cells of a concentration field on a subset of the fluid lattice.

plotConcSlice.m: Matlab code to plot a concentration field on the fluid lattice.

plotConnections.m: Matlab code to plot points paired in a connection data structure.

plotGeoConcSlice.m: Matlab code to plot cells associated with concentration field on a subset of the fluid lattice.

plotPressureSlice.m: Matlab code to plot the pressure field over a slice of the fluid lattice.

plotSlice.m

plotVelSlice.m: Matlab code to plot vector field over a slice of the fluid lattice.

readArray.m

readGaussianConc4Params.m

readGeoCellData.m

readSparseArray.m

read_fiber_force.m

read_fiber_vel.m

read_force.m

read_force_modes.m

readModelDataPolymer1.m

read_physical.m

read_pressure.m

read_sp_conc.m

read_sp_conc_indices.m

read_sp_fiber.m

read_sp_fiber_force.m

read_sp_fiber_indices.m

read_sp_fiber_vel.m

read_sp_force.m

read_sp_geoConc.m

read_sp_pressure.m

read_sp_vel.m
read_sp_vel_modes.m
read_vel.m
read_vel_modes.m: Matlab codes to read data files.
units1.m: Matlab code which specifies important physical parameters used throughout the codes.

writeArray.m
writeConcFile.m
writeFiberFile.m
writeForceFile.m
writeGaussian4_scpScriptUpdate.m
writeGaussian4_sftpScript.m
writeGaussian4_startScript.m
writeGeoCellData.m
writeGeoConcFile.m
writeParamsGaussianConc4.m
writeScpScriptGetData.m
writeScpScriptLocalPrep.m
writeScpScriptRemotePrep.m
writeScpScriptUpdate.m
writeSparseConcIndices.m
writeSparseFiberIndices.m
writeSparseForceIndices.m
writeSparseGeoConcIndices.m
writeSparseIndices.m
writeSparseVelIndices.m
writeSparseVelModesIndices.m
writeUserExtras4_Polymer4.m
writeVelFile.m
writeZSliceConcSparseIndices.m
writeZSliceForceSparseIndices.m
writeZSlicePressureSparseIndices.m
writeZSliceVelSparseIndices.m: Matlab codes to write data files.

Utility:

fpack.c: Utility program which allows for multiple files to be packaged into a single file. This is useful to avoid excessive number of files being transferred by SFTP, etc.

fpackSparse.csh
fpackSparseClean.csh: Scripts useful in packaging files.
makefile: File used with "make" to build executables.
unfpack.c: Utility program which allows for multiple files to be packaged into a single file. This is useful to avoid excessive number of files being transferred by SFTP, etc.