

# *StochKit* in a Nutshell

## – Introduction and Use

Hong Li      Linda Petzold

### 1 Overview

*StochKit* 1.0 is an efficient, extensible software toolkit for discrete stochastic and multiscale simulation of chemically reacting systems. *StochKit* aims to make stochastic simulation accessible to practicing biologists and chemists, while remaining open to extension via new algorithms. This document gives basic instructions for the installation and initial use of *StochKit*. For more detailed information, see the *StochKit* User’s Guide and the README files provided in each directory.

*StochKit* 1.0 includes basic simulation modules such as Gillespie’s SSA algorithm, Optimized Direct Method SSA, explicit, implicit, and trapezoidal tau-leaping methods. It also includes adaptive-stepsize, non-negativity-preserving tau-leaping/SSA, and an implementation of the Slow-Scale SSA method(ssSSA). Useful tools are provided to make stochastic simulation more convenient. We provide a Java Converter to convert an SBML file specifying the chemical mechanism to the input files needed for our software. The Kolmogorov distance and histogram distance routines for quantifying differences in statistical distribution shapes are provided in the Matlab language. For those who need to run the Monte Carlo simulations a large number of times to collect the ensemble, we provide a MPI interface enabling the Monte Carlo simulation to run on a parallel cluster. *StochKit*’s core computational modules [1] are written in the C++ language.

*Stochkit* can be downloaded from <http://www.engr.ucsb.edu/~cse/>.

### 2 Installation Instructions (Unix Only)

- Prerequisites:
  - StochKit release file
  - Java (Optional, required only for the SBML2StochKitConverter. Tested with Java version “1.4.2”)
  - SPRNG (Optional parallel random number generator. Required only for the parallel tool.)

- Matlab (Optional, required only for the DataAnalysis tool)
- Decompress *StochKit* release file `StochKit.tar/StochKit.tar.gz`  
 Use the command “`tar -xvf StochKit.tar`” or “`gtar -zxvf StochKit.tar.gz`” to decompress the *StochKit* release file, which will create a directory named “StochKit”. The directory should include :
  - `makefile` (makefile for the *StochKit* package)
  - `makefile.config` (makefile’s configuration of the *StochKit* package)
  - `test` (the demo code)
  - `Documents` (detailed UserGuide for the *StochKit* package)
  - `Tools` (The MPITemplate, DataAnalyzer, SBML2StochKitConverter)
  - `Math` (The linear algebra library for *StochKit*)
  - `StochRxn` (the core computational algorithms)
  - `README` (Version information, record of revisions to *StochKit*)
  - `INSTALL` (Simple installation instructions)
- Make *StochKit*  
 Go to the *StochKit* directory, and type the following commands:
  - `make clean`
  - `make`
- Install and Compile SPRNG (Optional, required only for the parallel tool)
  - Go to <http://sprng.cs.fsu.edu/> to find SPRNG and the installation details
  - Go to `StochKit/Math/sprng2.0` to change the Makefile and compile
  - Change the `makefile.config` under the *StochKit* directory as follows:  
 change  
`SPRNGOPTION = _NO_SPRNG`  
`#SPRNGOPTION = _USE_SPRNG`  
 to  
`#SPRNGOPTION = _NO_SPRNG`  
`SPRNGOPTION = _USE_SPRNG`
- Install Parallel *StochKit* (Optional, required only for the parallel tool)
  - If MPI (Message Passing Interface) is not installed on your computer, you should ask your system administrator to do it
  - Set environment variables

- \* Set the environment in your `.cshrc` (`.cshrc` is for Cshell; for other shells, you may set the environment in the corresponding profile)  
You may find some examples of `.chsrc` in the directory `StochKit/tools/MPItemplate/README/envTemplate`
- \* Set the environment variables to specify the default path for the `mpich/mpi` (if the system didn't include this environment variable)
- \* Example: `setenv PATH $PATH:./usr/local/mpich/bin/`
- Prepare to use PStochKit  
Before using PStochkit, you should have installed StochKit.
- Compile the package  
Compile all the libraries in this package:
  - \* Compile SPRNG for the parallel machine  
See the README for the SPRNG `StochKit/Math/sprng2.0` in the detailed UserGuide
  - \* Compile the Math lib and the *StochRxn* lib  
Simply build the whole package
- Write your parallel code using the template  
Details can be found in the README for the parallel code *StochKit*

## 3 StochKit Documentation

### 3.1 Learn how it works

- Go to `StochKit/test`
- Go to DIMER. The directory should include:
  - Adaptive (Adaptive Method for DimerDecay)
  - ODM (Optimized Direct Method for DimerDecay)
  - DM (Direct Method for DimerDecay)
  - ExpTau (Explicit Tau Leaping Method for DimerDecay)
  - ImpTau (Implicit Tau Leaping method for DimerDecay)
  - SSSA (The Slow Scale SSA for DimerDecay)
  - Makefile
- Go to each directory above and type the following command:
  - `make clean`
  - `make`
- Run each demo by typing either

- XXXstat N result.txt (N represents how many realizations you need to run; XXX represents the model name), for an ensemble run  
or
- singledrive result.txt (XXX represents the model name), for a single run

## 3.2 Make your own driver

- To write the driver by hand
  - Go to StochKit/test
  - Make a new directory with your desired name. It is suggested to use your model name
  - Decide which method you will use
  - Copy the corresponding directory from the DIMER directory
  - Go to the subdirectory
  - Either rename “DimerStats.cpp” to your desired name and modify the corresponding content for your model according to the Detailed UserGuide [1] for an ensemble run, or rename “SingleDriver.cpp” to your desired name and modify the corresponding content according to the Detailed UserGuide [1] for a single run
  - Modify ProblemDefinition.cpp to insert the corresponding content for your model according to the Detailed User’s Guide [1]
  - Modify the Makefile with your own filenames
  - Type the commands “make clean; make” to generate the executable code
  - Run each demo by typing either
    - \* XXXstat N result.txt (N represents how many realizations you need to run; XXX represents the model name), for an ensemble run  
or
    - \* singledrive result.txt (XXX represents the model name), for a single run
- To provide the driver via the SBML2StochKitConverter
  - Prepare your model in SBML format. Currently our Converter can accept SBML level1 models and some SBML level 2 models. We are working on it to accept SBML level2 models.
  - Go to StochKit/tools/SBML2StochKitConverter
  - Put your SBML model in this directory

- Type the following Command: `run XXXX.xml 1` (XXX represents the model name), which will create a subdirectory with the name of your model name inside the SBML file
- Go to this subdirectory for your model. The directory should include
  - \* `Stat.cpp` (The driver. You need to change the configuration to satisfy your specific needs.)
  - \* `ProblemDefinition.cpp` (The model definition)
  - \* `forviewtest.xml` (To read your model with Internet Explorer)

### 3.3 To write parallel code with the template

- See the “READMEforPStochkit”  
which is in the directory `StochKit/tools/MPItemplate/README`

## 4 Contact Information

- `stochkit@cs.ucsb.edu`
- `hongli@cs.ucsb.edu`

## References

- [1] Andrew Hall, Yang Cao, Hong Li, Sotiria Lampoudi, User’s Guide for Stochkit. <http://www.engr.ucsb.edu/~cse>