UPPSALA
UNIVERSITET

Modelling of Biochemical networks in cells using stochastic simulations

Enhanced SSA and CME

Tomas Sundvall, David Trång, Niklas Wikén

Project in Computational Science

January 2015

PROJECT REPORT

Department of Information Technology

Abstract

The Chemical Master Equation (CME) and the Stochastic Simulation Algorithms are two methods that are used to model living cells. The CME is a deterministic model that is efficient to use when applicable, but due to the curse of dimensionality it can not be used for high dimensional problems. Because SSA is a Monte Carlo method, it deals well with high dimensional problems, but suffers from slow convergence instead. In this project an improved SSA called Enhanced Stochastic Simulation Algorithm (ESSA) and an improved CME (ICME) are presented for steady state problems. As ESSA exploits properties for steady state problems that are not dealt with by standard SSA, the speedup gets very big. For the problems examined in this project, the speedup ranges betwee 10^4 and 10^6 for the ESSA and between 1.5 an 2 for the ICME.

Contents

1 Introduction

Modeling living cells is of great importance in order to understand the very complicated biochemical networks in cells. Inside living cells molecules move in space by diffusion and react with each other forming new compounds. The probability that a certain compound is created or disappears depends partly on to the concentration of molecules in the cell, and are thus modified with every reaction taking place. In classical macroscopic chemistry there are deterministic rate-diffusion laws in the form of differential equations that the concentrations of the reacting substances generally obey. Those laws are, however, built upon the case that there are enough chemically active molecules such as proteins present, which is often not accurate for living cells. Therefore the reactions are better described as random processes [4]. Inside living cells molecules can move more or less freely by random Brownian motion, and the event of molecules reacting with each other can be scheduled as random. For some processes it can be assumed that the systems are well stirred, and thus the space dependency of the solution can be ignored. In this project we only consider well stirred systems, and because of this the problem can be decomposed into two components. The first one is to determine the time until the next reaction occurs and the second is to determine which reaction that took place.

One way to model the concentration of substances after a certain time t is by using the so called Chemical Master Equation (CME), which is a system of first-order ordinary differential equations (ODE)s for determining the probability density function $p(\vec{x}, t)$. The probability function for each substance represents the probability that the system has a particular state at a given time t. The major advantage with CME is speed and accuracy. For systems with few substances, CME works well, but for systems with many substances CME cannot be used. CME suffers from the curse of dimensionality and the memory consumption may be a problem already for some of the two and three dimensional systems. For the cases where CME cannot be used, a Monte Carlo method called the Stochastic Simulation Algorithm (SSA) can be used instead. In SSA memory consumption grows linearly rather than exponentially. The main disadvantage with SSA is that the convergence rate is slow compared to CME. For large problems that has to be simulated for a long period of time it is sometimes too slow to yield a reasonable solution in a reasonable time.

Therefore, some high dimensional problems cannot be approached with either CME or SSA in reasonable time, and thus there is a need too further explore how this problems can be tuckled. Because the system of reactions evolves in time, one has to decide a stopping criteria. For some problems, the concentrations reaches a steady state after some time. These problems are said to reach a steady state, and in this project only steady state problems are considered. This poses another challenge, because when solving steady state problems it must somehow be determined if the system has reached steady state or not.

2 Chemical Master Equation

The Chemical Master Equation or more generally, the Master Equation is a set of first-order ordinary differential equations that describes the time-evolution of the probability to occupy a set of states [1].

We define v_r as the change of molecules for reaction r, $a_r(\vec{x})$ as the rate at which reaction r occurs, and $\vec{x}$ is the actual states, a vector of integrals $p(\vec{x}, t)$ is the probability to be in state $\vec{x}$ at a time t. The probability vector during an infinitesimal time step dt is then defined as

$$p(\vec{x}, t+dt) = \sum_{r=1}^{R} dt \times a_r(x_r - v_r)p(x_r - v_r, t) + \sum_{r=1}^{R}(1 - dt \times a_r(x_r))p(x, t). \quad (2.1)$$

With some simple algebraic transformation we can rewrite the equation as

$$\frac{p(\vec{x}, t+dt) - p(\vec{x}, t)}{dt} = \sum_{r=1}^{R} a_r(x_r - v_r)p(x_r - v_r, t) - \sum_{r=1}^{R} a_r(x_r)p(x_r, t). \quad (2.2)$$

And by letting $dt \to 0$ we obtain the so called general Chemical master equation

$$\frac{dp}{dp_t} = \sum_{r=1}^{R} a_r(x_r - v_r)p(x_r - v_r, t) - \sum_{r=1}^{R} a_r(x_r)p(x_r, t). \quad (2.3)$$

In matrix form it reads

$$\frac{dp}{dt} = Ap. \quad (2.4)$$

To construct the transition matrices A for the CME we start from a reaction model, which we will derive from one of the simplest models. We have one substance A that decays with rate k.

$$X \xrightarrow{\text{k}} \emptyset \quad (2.5)$$

where X is a chemical species of molecules and k is the rate at which the molecule decomposes. The probability that exactly one reaction occurs during the infinitesimal interval $[t, t+dt)$ is $X(t)kdt$ where $X(t)$ is the number of chemicals of type X at time t.

Introducing $p_n(t)$ as the probability that there are n molecules of type X at time t, i.e $X(t) = n$. For the infinitesimal time step from t to $t + dt$, there are two ways to reach $p_n(t)$. Either by degradation of one molecule from the state p_{n+1} or that the system remains in state p_n. Thus

$$p_n(t + dt) = p_n(t) \times (1 - kndt) + p_{n+1}(t) \times k(n + 1). \quad (2.6)$$

Subtracting both sides by p_n and dividing by dt yields

$$\frac{p_n(t + dt) - p_n(t)}{dt} = -kn \times p_n(t) + k(n + 1) \times p_{n+1}(t). \quad (2.7)$$

Letting $dt \to 0$ results in the CME for the system

$$\frac{dp_n}{dt} = k(n + 1) \times p_{n+1} - kn \times p_n. \quad (2.8)$$

This system of ODEs can easily be solved approximately using some discrete time stepping scheme such as Euler backwards

$$\frac{p^{k+1} - p^k}{\Delta t} = Ap^{k+1}. \tag{2.9}$$

Some simple algebra then yields

$$p^{k+1} = (I - \Delta t \times A)^{-1} p^k \tag{2.10}$$

where I is the identity matrix and A is the transition matrix. For systems of more molecules we use the subscript $p_{n,m}$ where n and m defines the number of molecules of molecule X and Y respectively.

3 Stochastic Simulation Algorithm

Even though CME is efficient when applicable, its inability to solve even small problems led researchers to start developing Monte Carlo methods to model living cells [1]. The main idea with SSA, as for all Monte Carlo methods, is to simulate the system with its intrinsic randomness many times, save the result and compute the mean values. For chemical reactions, the changes in the system can be regarded as trajectories in an n-dimensional space, where n is the number of reacting substances. An example plot of the trajectories for the one-dimensional problem

$$\begin{aligned} &\emptyset \xrightarrow{k_1} X \\ &X \xrightarrow{X \cdot k_2} \emptyset \\ &k_1 = 10s^{-1}, k_2 = 0.1s^{-1} \end{aligned} \tag{3.1}$$

can be seen in Figure 1. Problem (3.1) converges to the concentration 10 of X. SSA is basically a general-purpose method that is a Monte Carlo method developed for the specific task of simulating chemical reactions [1]. SSA was developed by Gillespie in 1976 [2][3]. The problems that are solved using SSA all consist of a set of reactions, that occur with a certain rate. These rates are denoted $\alpha_1, \alpha_2, ..., \alpha_n$ and for problem (3.1) we thus have

$$\begin{aligned} \alpha_1 &= k_1, \\ \alpha_2 &= X \cdot k_2. \end{aligned}$$

The derivation of SSA is based upon two fundamental questions. These questions are:

- Given the current time t, how long time τ will it take before the next reaction will occur?
- Which reaction will take place at time $t + \tau$?

The time to the next reaction is exponentially distributed with rate α_0 where

$$\alpha_0 = \sum_{i=1}^{n} \alpha_i \tag{3.2}$$

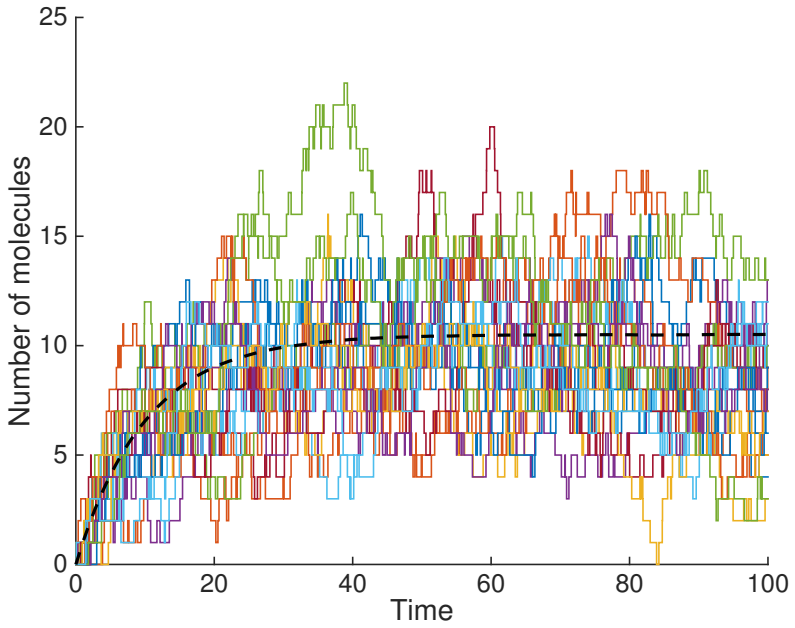

Figure 1: 20 trajectories for problem (3.1). Mean value marked with black dashed line.

and r_1 is a uniformly distributed random number in the interval $[0, 1)$. In order to determine τ, the formula

$$\tau = \frac{1}{\alpha_0} \cdot ln\Big[\frac{1}{r_1}\Big] \tag{3.3}$$

is used. To determine which reaction will occur, another uniformly distributed random number, r_2 have to be generated. The rate of a particular reaction compared to the rate of the other reactions will determine the probability that it was the reaction that took place. The most computationally expensive part of a Monte Carlo method is the generation of random numbers. For SSA two random numbers have to be generated each time a reaction occurs and because of this, the problems that will be the most expensive to solve with SSA are problems with high reaction rates and long time intervals.

The SSA can be summarized by the following four steps, where $\vec{x}$ is the vector containing the current states for the substances and $\vec{y}$ is a vector containing the reactions:

1. Generate two uniformly distributed random numbers r_1 and r_2 in $[0, 1)$.
2. Compute α_0 by (3.2).
3. Use (3.3) to compute the time τ to the next reaction.

4. Determine which reaction R will take place by

$$R = \begin{cases} y_1 & \text{if } 0 \ll r_2 < (\alpha_1)/\alpha_0 \\ y_2 & \text{if } (\alpha_1)/\alpha_0 \ll r_2 < (\alpha_1 + \alpha_2)/\alpha_0 \\ & \vdots \\ y_n & \text{if } (\alpha_1 + \alpha_2)/\alpha_0 \ll r_2 < (\alpha_1 + \alpha_2 + \cdots + \alpha_n)/\alpha_0 \end{cases} \tag{3.4}$$

and update $\vec{x}$ accordingly. Then continue to step (3) if $t < T$ where T is the stopping time. Otherwise terminate.

As can be seen from Figure 1, even though the mean value of the trajectories converges to a steady state, a single trajectory is not. This means that even though the mean value converges to ten for problem (3.1) the probability that a single trajectory is in state ten will not increase with time when steady state has been reached. As can be seen from Figure 2 the probability of where a single

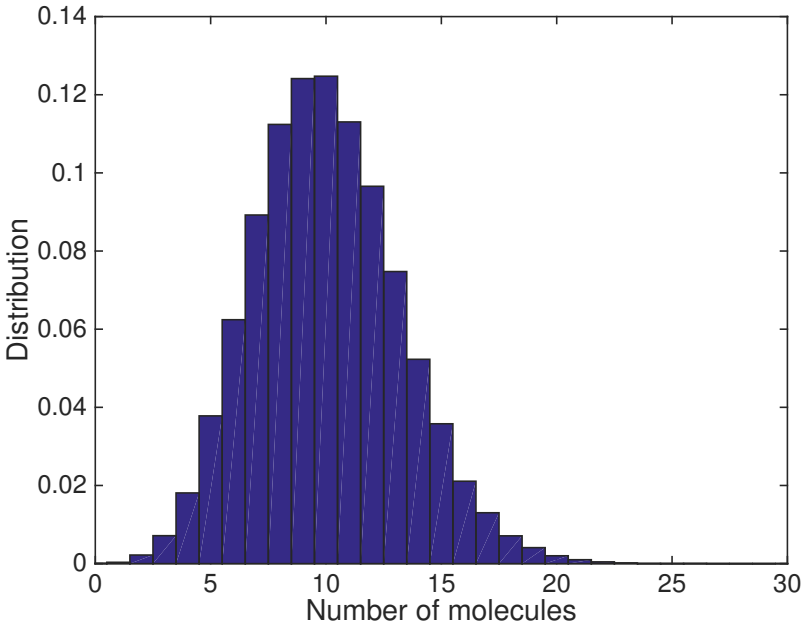

Figure 2: Probability distribution for problem (3.1) in steady state after 10^3 iterations.

trajectory ends up follows the Poisson distribution. In fact, each new reaction that occurs after a trajectory has reached steady state can be considered equivalent to a completely new trajectory. This is a property that can be exploited when solving steady state problems.

4 Improved methods

Improving the performance of SSA and CME is about identifying computationally heavy parts and try to improve them. The main idea with these new

methods is to reduce the transient time, i.e the time to simulate a system from initial point to reach the area of steady state. Since only the probability of how many molecules a system has of each type is of interest, it is of no interest how a chemical system evolve to reach steady state. Hence it does not matter if the evolution is reasonably likely or not as long as the solution at final state is the same. Therefore losing accuracy, or simulating an extreamly unlikely evolution to get to the steady state does not matter, as long as the final solution is correct. We can then perform simulations using system of equations of much coarser mesh than the original one.

4.1 Improved CME for Steady State Problems

As stated above the transient phase of the CME is expensive to compute for a large equation system. The idea presented here is to project the system on to a coarser mesh. We can then simulate on the coarser system until steady state is reached, which is much faster than before. What is obtained is not a solution but a good hint at where the solution is. From that point we interpolate back to the fine mesh and simulate to steady state. We will lose some computational time by performing the projection and interpolation to and back from the coarser mesh. But that is just a small fraction of the gain by starting the simulation on the fine mesh closer to the steady state.

4.1.1 Coarsing the mesh

To coarsen the transition matrix and the probability vector we introduce the operator $E^{m\times n}$, where $n = 2m$, as

$$E = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & & & \ddots & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \tag{4.1}$$

and operator $F^{n\times m}$ as the transpose of E divided by two (i.e. $F = \frac{1}{2}E^T$) to conserve the probability. The transition matrix A_0 in (2.4) from the CME is coarsened algebraically by multiplying it with E and F

$$A_m = EA_{m-1}F \tag{4.2}$$

and the solution vector p is then

$$p_m = Ep_{m-1}. \tag{4.3}$$

We then coarsen until p_{M_0} and A_{M_0} for some predefined M_0. This method may easily be extended to higher dimensions by taking the Kroenecker product of E with itself in the number of dimensions that is desired.

4.1.2 Interpolation

To return to the fine mesh we need to interpolate from the coarse mesh. The simplest way of interpolating is by using the F operator,

$$p_{m-1} = Fp_m \tag{4.4}$$

which splits the probability on the coarse mesh to the two new nodes, which corresponds to a constant interpolation. The probability for a system to be constant is very unlikely. To improve the quality of the interpolation we choose a linear interpolation F' instead.

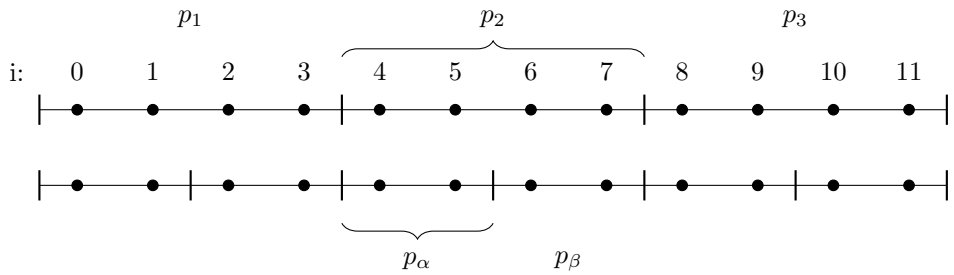

Figure 3: In the upper scale the box-size is of length four and in the lower finer mesh has a box-size of length 2. The fine mesh consits of the black circles and the numbers correspond to a linear distribution over the states.

From the setting figure 3 we get the following equations

$$p_\alpha = w_1p_1 + w_2p_2 + w_3p_3, \tag{4.5a}$$
$$p_\beta = w_3p_1 + w_2p_2 + w_1p_3, \tag{4.5b}$$

due to symmetry. To create the interpolation matrix F' we must choose weights w_i such that the probability mass p_m is exact for

$$p_m = c \ \ (constant),$$
$$p_m = c + i \ \ (linear).$$

We have that

$$w_1c + w_2c + w_3c = \frac{1}{2}c \tag{4.6}$$

for a constant state. Using the data from Figure 3 and equation (4.5), then with a linear p_m in (4.5) we obtain

$$\frac{c}{2} + 9 = w_1(6 + c) + w_2(22 + c) + w_3(38 + c), \tag{4.7a}$$
$$\frac{c}{2} + 13 = w_3(6 + c) + w_2(22 + c) + w_1(38 + c). \tag{4.7b}$$

The equation (4.7) can be reduced to

$$9 = 6w_1 + 22w_2 + 38w_3, \tag{4.8a}$$
$$13 = 6w_3 + 22w_2 + 38w_1. \tag{4.8b}$$

With some algebra we get that

$$w_1 = \frac{1}{8} + w_3. \tag{4.9}$$

By inserting equation (4.9) in equation (4.6) we have that

$$w_2 = \frac{1}{2} - w_1 - w_3 = \frac{3}{8} - 2w_3. \tag{4.10}$$

From this we see that w_1 and w_2 satisfy equation (4.6) and w_3 can be chosen arbitrarily. By letting $w_3 = 0$ we get $w_1 = 1/8$ and $w_2 = 3/8$. Finally we insert the weights w_i to the equations (4.5)

$$p_\alpha = \frac{1}{8}p_1 + \frac{3}{8}p_2 \tag{4.11a}$$

$$p_\beta = \frac{3}{8}p_2 + \frac{1}{8}p_3, \tag{4.11b}$$

and from here we can easily construct the interpolation matrix F'.

4.2 Enhanced SSA for Steady State Problems

By analysing the SSA method with MATLAB's profiler, it was determined that most of the computational time is spent on generating random numbers, calculating probabilities and if-statements (checking which reaction that occurs). Things that can be optimized are computational time spent in each Monte Carlo step, the overall computation time spent on simulating trajectories to steady state, and reducing the number of Monte Carlo steps needed to reach steady state. Examined optimizations are presented in Section 4.2.2, and the best combination of these is presented in Section 4.2.3.

4.2.1 Simulating in Steady State to Generate Points

A collection of SSA trajectories can be described as an approximation of a map that contains the probability that a chemical system has a specific number of molecules of each kind. The steady state solution to an SSA problem is then reached when the probability no longer depends on time. In other words, the likelihood that a chemical system has a specific number of molecules of each substance at a given time is constant. While a chemical system is at steady state, it is still changing the concentrations of different kinds of molecules in the system, but the change is according to the probability. It is the probability distribution that is not changing and hence is in the steady state.

If a chemical system is in steady state, then it is possible to build the probability map by recording how many times the system has had a specific number of molecules of each type during a long specified time divided by the total number of visits. We refer this long specified time to as a *time window*. The wider the time window is, the more data points are recorded and the better the probability map is. Hence, it is possible to decide how many data points in the solution one wants, which affects the accuracy.

Calculating the probability map by simulation a large number of chemical systems and record the final point for each simulation is equivalent to simulating one chemical system and record how the system changes in steady state. This is valid unless the time window is too narrow, since a short walk in steady state only generates data points in a small neighbourhood instead of being randomly distributed which would be the case when using many different simulations.

4.2.2 Investigated modifications

The first idea was to use the transition matrix from CME to reduce time spent in each Monte Carlo step since the probabilities would not need to be computed. The Monte Carlo step would then only need to generate two random numbers and some if-statements to decide a reaction. The problem is that this modification suffers from the curse of dimensionality in the same way CME does. Hence, this limits the solver to only be able to solve systems as large as CME can handle, therefore this modification is useless no matter how fast the speedup would have been. Another problem is that MATLAB computes the probabilities faster than it can read from the transition matrix. It might also be the case for compiled languages due to data locality problems.

Another idea was to remove the time dimension because only the steady state is of interest. This modification would reduce the time spent in a Monte Carlo step since only one random number would have had to be generated, instead of two. The problem is that this modification changes the probability map for most chemical system. In many chemical systems, the time between reactions is smaller if there are many molecules present. If there are many molecules present, then the mean time to next reaction is smaller than if there are few molecules. This means that the mean value would be shifted upward if the time dimension is removed. This makes the modification useless since the steady state solution is not valid.

Grouping together trajectories in the simulation until the steady state is reached and them split them up was an attempt to reduce total time spent on simulating trajectories before reaching steady state. The idea was to simulate only a few chemical systems and let other systems just follow until steady state is reached. This modification greatly reduces the computation time dedicated to simulating trajectories to steady state. One downside is that some problems, for example toggle switch, have more than one equilibrium. Such a problem needs to have approximately the same number of trajectories in each equilibrium. Using too few trajectories leads to a larger probability of an uneven distribution of trajectories among the equilibriums.

The fourth modification was to reduce the number of Monte Carlo steps needed for a trajectory to reach steady state. This was done by letting chemical systems perform a multiple number of a chosen reaction instead of just one for each randomized time dt. This in turn requires the time dt to be larger since more reactions happen at dt. The time dt is approximated to be multiplied by

the same number as the number of reactions performed by the chemical system. This modification coarsens the system, hence we call the factor χ. The time is now calculated as

$$dt = \chi \frac{1}{\alpha_0} log(1/r). \tag{4.12}$$

The drawback with this modification is that the probability map is smoothed out since it is more likely that the system has extreme concentrations of molecules. To make this modification useful, then χ has to be reduced when approaching steady state.

The last modification tried out was to use the fact that generating a large number of trajectories gives the same probability map as recording the movement of a trajectory in steady state for a long time. This has a potential to be a huge improvement since generating an extra point for a trajectory in steady state is a fraction of the whole compared to simulate another trajectory from scratch. It is however only applicable in steady state problems. Another drawback is for problems with multiple equilibria where the probability to move between equilibria are small. One of these equilibria might get very explored but others may be unvisited even after a very long time. This effect can be reduced by using more trajectories to enable an even distribution of trajectories between the equilibria. This reduces the potential improvement in computing time but gives a more accurate result.

4.2.3 Enhanced SSA method explained

The last two modifications described above were successful and resulted in an improved SSA method. These modifications can be used and implemented in many ways. The success of the method presented is heavily depending on how the parameters are set. The selection of parameters is for now based on a trial and error approach, however, no heavy computational work is needed to pick parameters which are good enough. The enhanced SSA method is for now on referred to as ESSA.

The idea is to divide the chemical time which goes from zero to steady state, into a number of time frames. Each time frame is going to have a specific χ in a decreasing manner, so that the first time frame has the highest χ and the last time frame should have χ equal to one. The size of the first time frame should be quite large, the size of the last time frame should be about half of the total time. The size of the time frames in between should be small. A system simulated through these time frames is in steady state if the parameters are defined in a sufficiently good way.

When a system is simulated to steady state, the simulation continues while recording the number of each molecule type after a reaction occurs and updating the probability map. The simulation continues until a stop time is reached.

The method can be summarized as follows:

1. Determine the number of time frames, the width of each time frame and the value of χ to be used in each time frame. Also initialize the chemical systems and stop times.

2. For every chemical system

   (a) For every time frame, do standard SSA simulation but update the molecules with a multiple of χ.

   (b) Continue the simulation by doing standard SSA simulation but record the number molecules of each kind and update the probability map accordingly. Continue the simulation until a stop time is reached.

5 Results

Three example problems were used to evaluate the methods, *toggle switch 2D*, *two metabolites 2D* and *two metabolites 3D*. CME could only be ran on the 2D problems since the 3D problem becomes too memory consuming. *Toggle switch 2D* has two equilibria while *two metabolites 2D* and *two metabolites 3D* have only one. *Two metabolites 2D* has the fastest reaction rate, while *two metabolites 3D* has the slowest reaction rate. The equilibria is much further away from the initial point in the *two metabolites 2D* compared to the other test problems. A chemical system in equilibrium has extreme low probability to drastically change concentration of molecules, the system rather stays in the equilibria. Problems with multiple equilibrium require special care when using ESSA since using too few trajectories might lead to a disproportionate distribution of trajectories among the equilibriums.

5.1 Enhanced Chemical Master Equation

The chemical time, which we define as the time simulated on MATLAB, required for reaching steady state for Toggle Switch and Two metabolites are approximately 50'000 seconds and 20'000 seconds respectively.

When running the improved method both problems have a minimum state limit set to 32 states in each dimension. The time step is equally long at all time. For achieving best possible speedup most of the chemical simulation time is done on the coarsest mesh and the least time on the finest mesh. The amount of chemical time that should be spent in each state is a trade-off between accuracy and computational speed. We have tuned this setting by starting from a very accurate solution and tried speeding up the computation as much as possible without increasing the error.

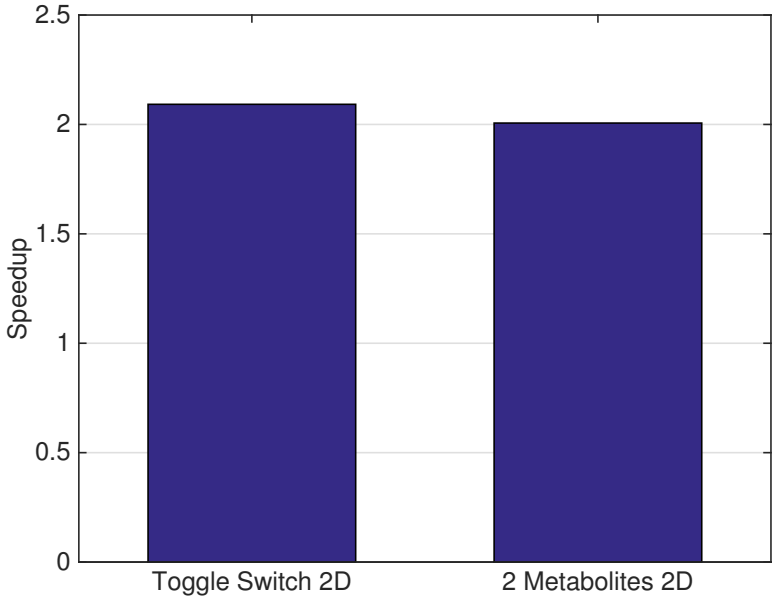

Figure 4: Speedup for 2 different setups. The Toggle Switch setup has 320×320 number of states and 2 Metabolites has 128×128 number of states.

Table 1: The L_1 error at steady state for each level of coarseness of the mesh. The reference solution is computed on a fine mesh.

Coarse level	error
0	0
1	0.5198
2	0.9252
3	1.2031
4	1.3963

The error in the solution is large already for the first level of coarseness, as can be seen from Table 1. This is due to the fact that the solution smears each time that we coarsen the mesh. At a first glance it might seem unnecessary to coarsen the mesh in the first place, since the error is still big. On the other hand the computational time taken to reach that stage is still relatively small compared to the simulation on the finest mesh.

5.2 Enhanced Stochastic Simulation Algorithm

The enhanced stochastic simulation algorithm (ESSA) is evaluated by first comparing the solution to a CME solution, then see how it converges when increasing the number of trajectories used and the recording time. Finally, the theoretical speedup of ESSA is compared with that of SSA.

In Figures 5 and 6 the ESSA and CME results are plotted together. It is clearly visible that the solutions are consistent. The convergence for ESSA is seen in Figures 7 and 8, it is measured by using the L_1 with a high resolution solution. Both convergence rates are reasonable since it is about the same convergence rate for a Monte Carlo method. Adding more trajectories seems a little more effective than increasing the recording time. This is reasonable since a trajectory mainly walks in its neighbourhood if the recording time is small. Adding another trajectory would then result in a more accurate result compared to just doubling the recording time.

It is hard to compare ESSA to SSA regarding speedup since it depends on how many trajectories and how long recording time ESSA is using. ESSA also has other parameters such as number of time frames to be used, which *coarsening factors* to be used and how long each time frame is. In the following result, three time frames have been used with the *coarsening factors* four, two and one for respective time frame. The time spent in each time frame is adjusted depending on the problem. The parameters have not been optimized but are good enough. Figure 9 shows the wall time it takes for SSA and ESSA to simulate one trajectory from initial point to steady state with these settings used. The enhancement to use a *coarsening factor* is clearly visible, ESSA is about twice as fast as SSA with this parameter setup. The wall time is calculated by taking the mean of 5000 trajectories.

Figure 10 shows how much additional time ESSA and SSA require to add another data point in the solution. The idea is that ESSA is already at steady state and to generate a new point is just to generate a new reaction. SSA has to simulate a new trajectory. Hence, the difference is huge. The wall time is computed by taking the mean of 5000 trajectories. Since the difference in wall time between ESSA and SSA is huge, the potential speedup is enormous. Reducing the number of trajectories used by ESSA makes the wall time to generate a large number of data points very short. The optimal number of trajectories depends on the problem. Problems with multiple equilibria require more trajectories since they require an even distribution of trajectories among the equilibria.

Since it takes very long time to generate a good solution to a problem with SSA, a speedup can only be theoretically calculated. Knowing the number of points in a good solution, the theoretical wall time SSA would require to compute a similar result is calculated as the number of points times the mean time of SSA for one trajectory to steady state. Figure 11 shows a speedup between theoretically calculated SSA and real ESSA runs.

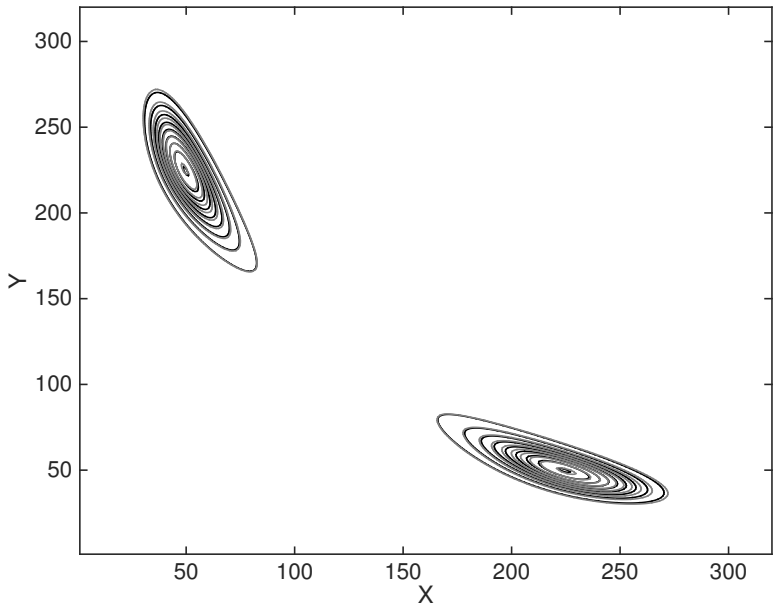

Figure 5: Solutions to the Toggle Switch problem computed with ESSA (gray) and CME (black).

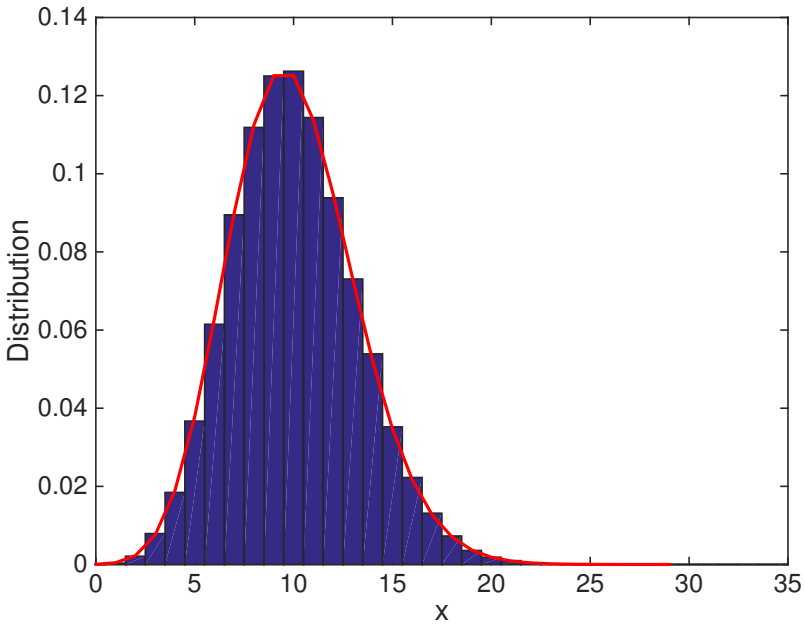

Figure 6: Solutions to the one dimensional problem (3.1) computed with ESSA and CME

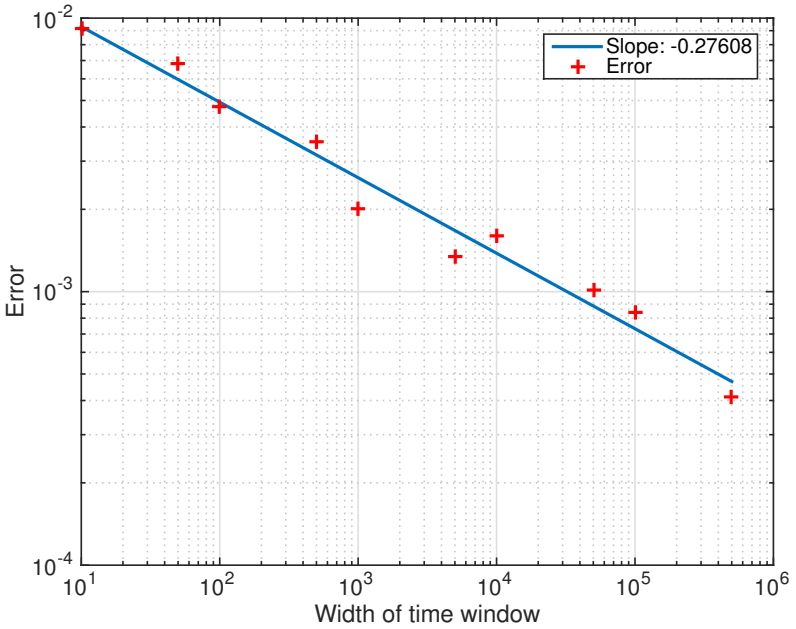

Figure 7: Convergence for ESSA when increasing the data point recording time. Doubling the recording time will double the data points in the solution.

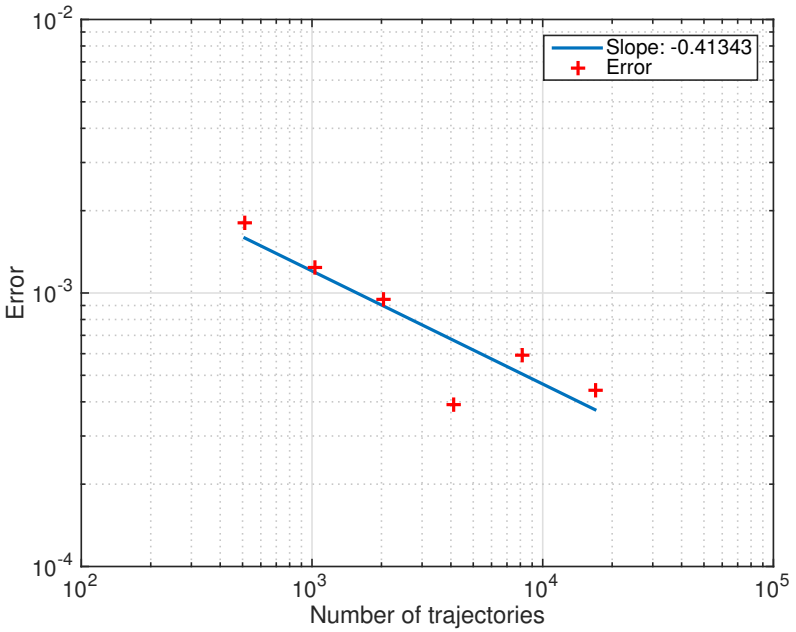

Figure 8: Convergence for ESSA when increasing number of trajectories used. Doubling the number of trajectories will double the number of data points in the solution.

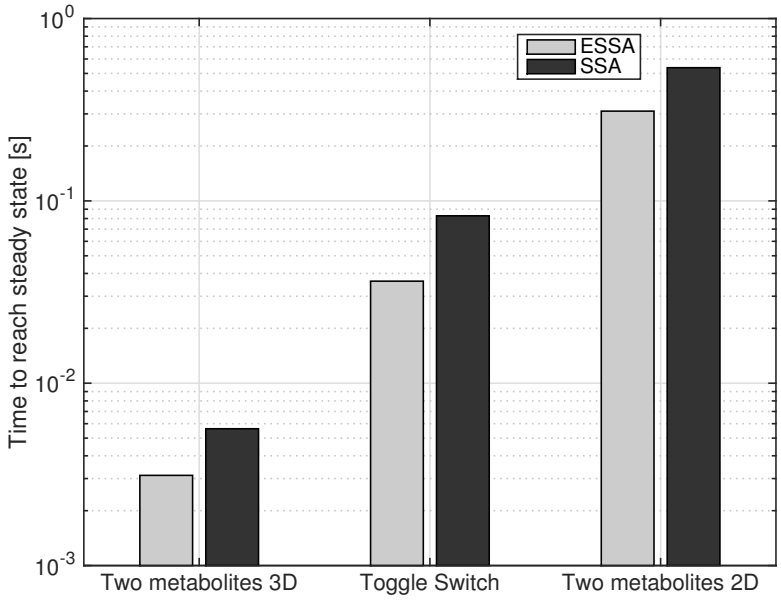

Figure 9: Wall time needed for a single trajectory in SSA and ESSA to reach steady state.

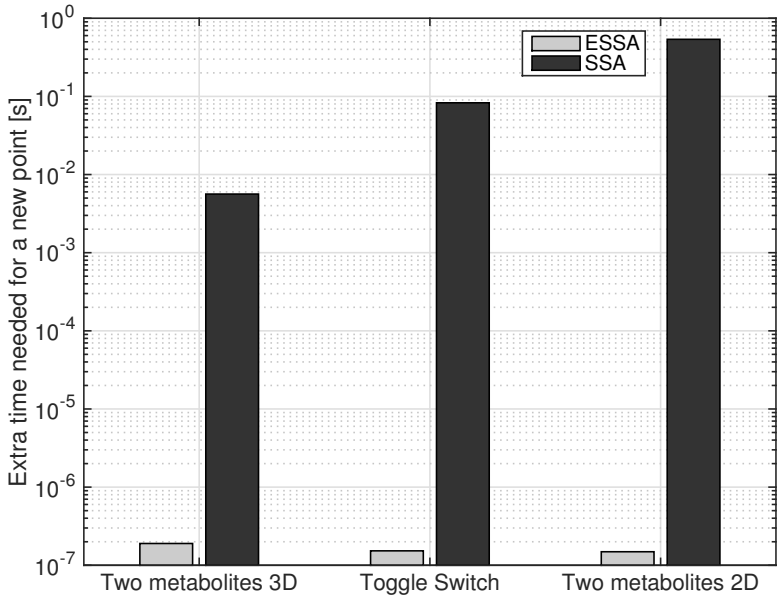

Figure 10: Wall time needed for SSA and ESSA to generate a new point in the solution if the chemical systems are in steady state. ESSA only simulates one extra reaction, SSA has to run a new simulation from scratch.

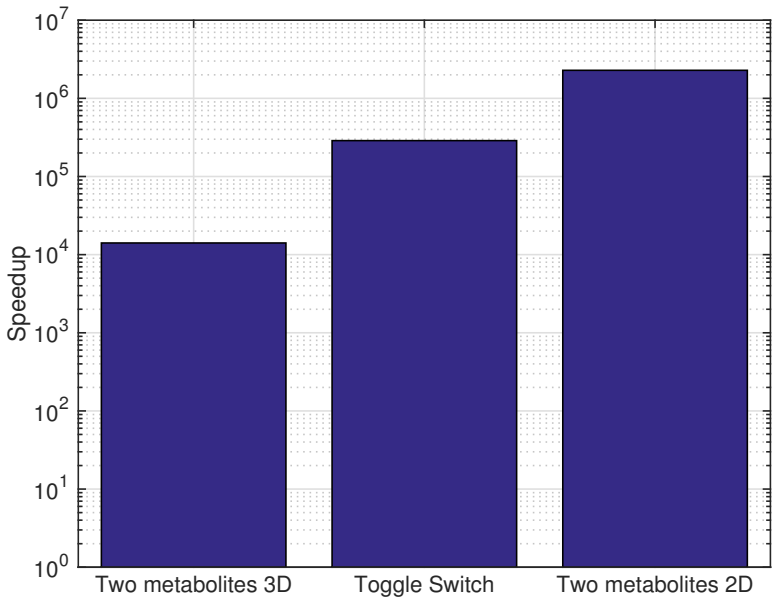

Figure 11: Example theoretical speedup. Real ESSA wall time compared to theoretical SSA wall time on one CPU core. The speedup can be improved by optimizing the number of trajectories used.

6 Conclusions

The Improved Chemical Master Equation (ICME) that was developed in this project resulted in a speedup of between 1.5 and 2.5 for the different problems, with a negligible error. The main issue with the CME was however not the speed, but rather that the memory consumption grew exponentially with the number of dimensions for the problems, making it impossible to solve even problems in a low dimension. This issue could not be solved with the ICME which greatly reduces the utility of the solution.

Another issue with ICME is that we perform an LU-factorization at each matrix transformation. The time spent on performing the LU-factorization is accounting for a major part of the total simulation time, hence even if we improved the method even more we would not be able to perform much better. A better solution would be to use an iterative solver instead.

Yet another issue that we encountered with the ICME, that should be regarded, is a boundary error that occurs when we coarsen the mesh. This error is though rare and only occur for systems with two or more molecules that can react, and there are an equilibrium at one of the boundaries (i.e. the number of one of the molecules is 0). The problem emerge from the fact that we add probability to the molecules to react when the actual probability for a reaction should be 0. This problem is easily fixed in two dimensions but becomes very hard as the number of dimensions increase.

In contrast to the ICME, the Enhanced Stochastic Simulation Algorithm yielded both interesting and very useful results. The speedup for ESSA compared to SSA approximately ranged between 10^3 and 10^6 which makes it possible to solve problems that would take too long for SSA, within a reasonable timespan for ESSA. The main explanation behind the large speedups is that ESSA exploits properties of steady state problems that are ignored by SSA. Thus, ESSA could also be described as an SSA version specialised for steady state problems.

7 Final thoughts

The main drawback with ESSA is obviously that it can not solve problems that do not have a steady state. Even for problems that have a steady state, it can sometimes be interesting to see how the system evolves in time, which can not be done with ESSA either. Because of the reduction of trajectories used is only applicable for finding steady state solutions, but also accounts for most of the speedup, it would not be possible to get even close to these results for non steady state problems. However, each trajectory in ESSA is also faster, and this is something that might be used in the development of an alternative method for non steady state problems.

There are also improvements that can be made to the ESSA developed in this project. The main drawback with ESSA apart from not being able to solve

non steady state problems is that it is vulnerable to parameter settings such as how many trajectories are to be used from the beginning, the length of the time window when determining in steady state etc. The characteristics of the problem to be solved determines which parameter setup is suitable. A possible opening for improvement is thus to make the ESSA more automatic. This might be done by adding steps that gathers information about the problem and trying in this way to choose good parameters automatically instead of manually.

Furthermore, a large portion of this project has been about researching different methods, and apart from the methods that proved successful, a great deal can be learned from the experiments that were not successful. In order to develop methods that can solve larger problems, it is based on our results probably best to direct further research into the development of improved SSA algorithms that can take large coarse steps when far from the target, and fine steps when close. SSA is also a good choice for further development since it is well suited for parallelization, and the hardware development today is more directed towards more CPU cores than higher speed.

References

[1] Andreas Hellander Daniel T. Gillespie and Linda R. Petzold. Perspective: Stochastic algorithms for chemical kinetics. *The Journal of Chemical Physics*, Vol. 138, May 2013.

[2] Daniel T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, Vol. 22 pp.403-434, December 1976.

[3] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, Vol. 81 pp.2340-2361, December 1977.

[4] Eric D. Siggia Michael B. Elowitz, Arnold J. Levine and Peter S. Swain. Stochastic gene expression in a single cell. *Science*, Vol. 297 pp.1183-1186, August 2002.