Diffusion Approximation for Bayesian Inference on Chemical Reaction Systems

Project Report for the course
"Projects in Machine Learning and Artificial Intelligence"

Florian Stimberg

April 16, 2009

Contents

1 Introduction

In recent years diffusion processes inside of single cells have gotten much interest. Since at this microscopic scale only small number of molecules are involved, the process becomes highly stochastic. To account for this it is common to use stochastic differential equations (SDE) for modeling diffusion processes. Bayesian inference is then used to compute the hazard parameters of the chemical reactions and predict the process at times, where it wasn't observed or correct measurement error. In most cases exact inference isn't possible or computationally inefficient for non trivial models and different types of approximations have been used to overcome this problem.

This project tries to examine and expand the approach of Golightly and Wilkinson (2008) by approximating the transition densities with the Fokker-Planck equation and the Euler approximation. To minimize discretization errors, latent data points are introduced in between measurements. A Gibbs sampler with a Metropolis-Hastings step is then used to alternate between sampling from the posterior of the latent data and the parameters. A new approach of block sampling is introduced, which switches between sampling the latent data with even and uneven indexes.

This paper is structured as follows. Section 2 introduces the Lotka-Volterra Model, which is used as a running example. In Section 3 Stochastic Kinetics are explained and the diffusion approximation, which was used, is described in Section 4. The inference algorithm is illustrated in Section 5 (5.1 for the case of exact measurements, 5.2 for noisy measurements) and the alternating block update is introduced in 5.3. Section 6 shows the results from multiple simulations, which are discussed together with possible improvements for the future in Section 7.

2 The Lotka-Volterra Model

The Lotka-Volterra-Model is a simple but non trivial diffusion model. It consists of two species Y_1,Y_2 called prey and predator and the following reactions:

$$R_1 \quad : \quad Y_1 \rightarrow 2Y_1 \tag{1}$$
$$R_2 \quad : \quad Y_1 \rightarrow \emptyset \tag{2}$$
$$R_3 \quad : \quad Y_2 \rightarrow 2Y_2 \tag{3}$$
$$R_4 \quad : \quad Y_2 \rightarrow \emptyset \tag{4}$$

representing reproduction and death of the species. It is important to know, that the probability of R_2 and R_3 depends on Y_1 and Y_2 (see Section 3 for

details). Figure 1 shows an example of a simulation using a Lotka-Volterra Model.

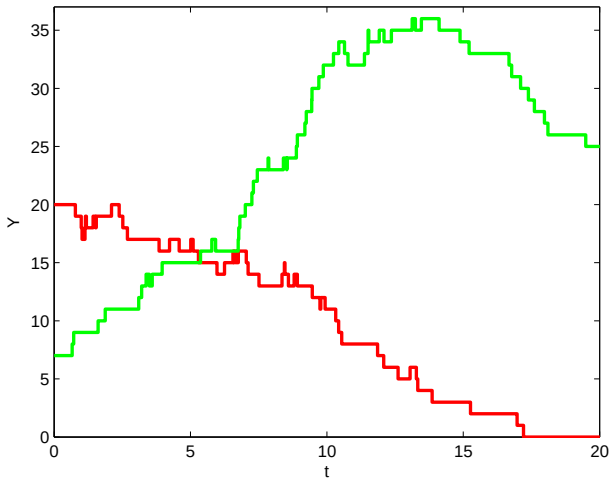

Figure 1: *Plot of a Lotka-Volterra simulation. The red line represents the prey population and the green line displays the predator population.*

3 Stochastic Kinetics

3.1 General Case

While molecules move around randomly through Brownian motion a reaction $R : Y_1 + Y_2 \rightarrow Y_3$ occurs if a molecule of type Y_1 hits a molecule of type Y_2. It is assumed that our system is kept well stirred, in a constant volume and is in thermal equilibrium. Gillespie (1992) showed, that under this circumstances the chance of a reaction happening is constant. It is furthermore assumed, that the law of mass action applies, which means that the chance of a reaction of type R is proportional to $Y_1 Y_2$, while here Y_i represents the number of molecules of species Y_i. Note, that this ambiguous notation is used throughout the rest of this paper.

If we have a system with k species and r reactions we can represent the reactions in two $r \times k$ matrices U and V, where u_{ij} is the number of molecules of species j needed for reaction R_i to happen and v_{ij} represents how much molecules of species j are products of reaction R_i.

The hazard of reaction R_i is denoted by $h_i(Y, c_i)$, where $Y = (Y_1, ..., Y_k)'$ is the current number of molecules for each species and c_i is called the rate

constant of the reaction. Since this hazard is zero if there aren't enough reactants for a reaction it is sufficient to use only one matrix $A = V - U$, called the net effect reaction matrix, to describe the reactions. For our purpose we try to solve the "chemical master equation", which describes the time evolution of the probability distribution over the system's states.

$$\frac{\partial}{\partial t}P(Y;t) = \sum_{i=1}^{r} \left(h_i(Y - A_i, c_i)P(Y - A_i;t) - h_i(Y, c_i)P(Y;t)\right) \tag{5}$$

where $P(Y;t)$ is the probability, that the system is in state $Y = (Y_1, Y_2, \ldots, Y_k)$ at time t. For a detailed derivation of the master equation see Gillespie (1992).

Gillespie (1977) introduced an algorithm, which allows to sample from the master equation. Let's consider the hazard of *any* reaction happening, which is

$$h_0(Y, \Theta) = \sum_{i=1}^{r} h_i(Y, c_i) \tag{6}$$

where $\Theta = (c_1, \ldots, c_r)$. This means, that the time to the next reaction is exponentially distributed with the parameter

$$\lambda = \exp(h_0(Y, \Theta)) \tag{7}$$

The Gillespie algorithm first samples from (7) to decide when the next reaction happens and then it determines what type of reaction occurs. By using (6), it is easy to see, that the probability of reaction i is

$$\frac{h_i(Y, c_i)}{\sum_{i=1}^{r} h_i(Y, c_i)} \tag{8}$$

It is notable that this algorithm truly generates continuous time samples and therefore gives exact samples from the master equation.

3.2 Example: The Lotka-Volterra Model

As described in Section 2, the Lotka-Volterra Model has four reactions. When computing their hazards it is important to remember that R_2 and R_3 depend on the number of prey and predator, although this isn't represented in their reaction equation. With this in mind it is obvious that the hazards are

$$h_1(Y, c_1) = c_1 Y_1 \tag{9}$$
$$h_2(Y, c_2) = c_2 Y_1 Y_2 \tag{10}$$
$$h_3(Y, c_3) = c_3 Y_1 Y_2 \tag{11}$$
$$h_4(Y, c_4) = c_4 Y_2 \tag{12}$$

and the net effect matrix is

$$A' = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix} \tag{13}$$

4 Diffusion Approximation

4.1 General Case

Because for most cases, finding an exact solution to the master equation isn't tractable we use the Fokker-Planck equation as an continuous approximation of the master equation. As shown in Van Kampen (2007) this can be done by a second-order Taylor expansion of the master equation, if we assume, that $P(Y,;t)$ changes slowly with Y and the Markov process' jumps are small.

For a process $Y(t) = (Y_1, \ldots, Y_k)$ the Fokker-Planck equation is

$$\frac{\partial}{\partial t}P(Y;t) = -\sum_{i=1}^{k}\frac{\partial}{\partial Y_i}(\mu_i(Y)P(Y;t)) + \frac{1}{2}\sum_{i=1}^{k}\sum_{j=1}^{k}\frac{\partial^2}{\partial Y_i \partial Y_j}(\beta_{ij}(Y)P(Y;t)) \tag{14}$$

with

$$\mu_i(Y) = \lim_{\Delta t \to 0}\frac{1}{\Delta t}E[(Y_i(t+\Delta t) - Y_i(t))|Y(t) = Y] \tag{15}$$

and

$$\beta_{ij}(Y) = \lim_{\Delta t \to 0}\frac{1}{\Delta t}Cov[(Y_i(t+\Delta t) - Y_i(t)), (Y_j(t+\Delta t) - Y_j(t))|Y(t) = Y] \tag{16}$$

for $i, j = 1, \ldots, k$.

Thus the Itô diffusion for (14) is

$$dY(t) = \mu(Y)dt + \beta^{\frac{1}{2}}(Y)dW(t), \tag{17}$$

where $\mu(Y) = (\mu_1(Y), \ldots, \mu_k(Y))'$ is known as drift, $\beta^{\frac{1}{2}}(Y)$ is called the diffusion matrix and needs to satisfy $\beta^{\frac{1}{2}}(Y)(\beta^{\frac{1}{2}}(Y))' = [\beta_{ij}] = \beta(Y)$. dW(t) represents the increment of k-dimensional Brownian motion.

Golightly and Wilkinson (2005) showed that for our system this leads to the stochastic differential equation

$$dY(t) = \mu(Y, \Theta)dt + \beta^{\frac{1}{2}}(Y, \Theta)dW(t) \tag{18}$$

where the drift and diffusion are

$$\mu(Y, \Theta) = A'h(Y, \Theta) \tag{19}$$

$$\beta(Y, \Theta) = A'diag(h(Y, \Theta))A \tag{20}$$

with A being the net effect matrix, $h(Y, \Theta) = (h_1(Y, c_1), \ldots, h_r(Y, c_r))'$ and $\Theta = (c_1, \ldots, c_r)$ is the parameter vector.

4.2 Example: The Lotka-Volterra Model

For the Lotka-Volterra Model (19) and (20) can be applied using the net effect matrix from (13) and the hazards from (9) to (12). This leads to

$$\mu(Y,\Theta) = \begin{pmatrix} c_1 Y_1 - c_2 Y_1 Y_2 \\ c_3 Y_1 Y_2 - c_4 Y_2 \end{pmatrix} \tag{21}$$

$$\beta(Y,\Theta) = \begin{pmatrix} c_1 Y_1 + c_2 Y_1 Y_2 & 0 \\ 0 & c_3 Y_1 Y_2 + c_4 Y_2 \end{pmatrix}. \tag{22}$$

5 Inference

Given a number of observations we want to conduct inference on the path between measurements and the parameters Θ. This is done by approximating a solution for (18). The first step is discretizing (18) by the Euler approximation

$$\Delta Y(t) = \mu(Y(t),\Theta)\Delta t + \beta^{\frac{1}{2}}(Y(t),\Theta))\Delta W(t) \tag{23}$$

with $\Delta W(t) \sim N(0, I\Delta t)$. Naturally this introduces an discretization error, which can be reduced by introducing latent data points in between observations. For the sake of simplicity we assume the observations are at evenly spaced times and insert $m-1$ latent data points between two neighboring observations. If we have l observations this leads to $(l-1)(m-1)$ missing values $Y_i(t_j)$, which are simulated. The simulated data together with the observations is called the augmented data $\hat{Y}$ and we can merge it into a matrix

$$\hat{Y} = \begin{pmatrix} Y_1(t_0) & Y_1(t_1) & \dots & Y_1(t_m) & \dots & Y_1(t_n) \\ Y_2(t_0) & Y_2(t_1) & \dots & Y_2(t_m) & \dots & Y_2(t_n) \\ \vdots & \vdots & & \vdots & & \vdots \\ Y_k(t_0) & Y_k(t_1) & \dots & Y_k(t_m) & \dots & Y_k(t_n) \end{pmatrix} \tag{24}$$

where $n = lm$.

5.1 Exact Measurements

In this section we assume that the observations aren't altered by noise and therefore won't be changed during the simulation. In this case the joint posterior of the augmented data and the parameters is

$$\pi(Y,\Theta|D) \propto \pi(\Theta)\left[\prod_{i=0}^{n-1} \pi(\hat{Y}^{i+1}|Y^i,\Theta)\right], \tag{25}$$

where Y^i denotes the ith column of $\hat{Y}$, $\pi(\Theta)$ is the prior density over the parameter values and

$$\pi(Y^{i+1}|Y^i,\Theta) = |\beta_i^{-1}|^{\frac{1}{2}} \exp\left\{-\frac{1}{2}(\Delta Y^i - \mu_i \Delta t)'(\Delta t \beta_i)^{-1}(\Delta Y^i - \mu_i \Delta t)\right\} \tag{26}$$

with $\Delta Y^i = Y^{i+1} - Y^i$, $\mu_i = \mu(Y^i, \Theta)$, $\beta_i = \beta(Y^i, \Theta)$.

To sample from (25), an Gibbs-Sampling algorithm is used by alternating between sampling from the latent data, conditional on the parameters and vice versa. Direct sampling from these conditionals is impossible and because of this an Metropolis-Hastings step is used for both conditionals. To get high acceptance rates a single-site update is used for sampling the latent data, conditional on the parameters. This means for each column Y^i (where i is not a multiple of m, thus not a measurement) a proposal is made and accepted accordingly to the Metropolis-Hastings algorithm. The conditionals to sample from are:

$$\pi(Y^i|Y^{i-1}, Y^{i+1}, \Theta, D) \propto \pi(Y^i|Y^{i-1},\Theta)\pi(Y^{i+1}|Y^i,\Theta) \tag{27}$$

$$\text{for } i = 1, \ldots, n-1 \tag{28}$$

$$\pi(\Theta|\hat{Y}, D) \propto \pi(\Theta)\left[\prod_{i=0}^{n-1} \pi(Y^{i+1}|Y^i,\Theta)\right] \tag{29}$$

In summary the employed algorithm is:

1. Set starting values for the parameters and initialize the latent data by linear interpolating between the adjacent observations.

2. For $i = 1, \ldots, n-1$ with i not being a multiple of m use a Metropolis-Hastings step to draw Y^i using a Gaussian density with mean $(Y^{i-1} + Y^{i+1})/2$ and variance $\frac{1}{2}\Delta t \beta(Y^{i-1}, \Theta)$ as proposal.

3. Use a Metropolis-Hastings step to draw a new Θ with a Gaussian random walk on $\log(\Theta)$ as proposal. This is done to ensure that all proposals are positive, which is necessary for the rate constants.

4. If enough cycles have been performed stop, otherwise return to step 2.

5.2 Noisy Measurements

To account for measurement errors, now the real process $X(t)$ is not observed directly, instead we observe $D(t) \sim X(t) + N(0, \Sigma)$, with $\Sigma = diag(\sigma_i^2)$. Because the noise covariance is unknown to the inference algorithm, it now

needs to learn the latent data and $\Theta = (c_1, \ldots, c_r, \sigma_1, \ldots, \sigma_k)$. Note that diagonal covariance was chosen for the sake of simplicity and the algorithm can cope with any given covariance matrix.

The additional noise term leads to a modified posterior:

$$\pi(\hat{Y}, \Theta|D) \propto \pi(\Theta)\pi(Y^0) \left[\prod_{i=0}^{n-1} \pi(Y^{i+1}|Y^i, \Theta)\right] \left[\prod_{i\in\{0,m,\ldots,n\}} \pi(D^i|\hat{Y}^i, \Theta)\right] \tag{30}$$

where $D = \{D^1, D^m, \ldots, D^n\}$ represents the noisy measurements.

For the parameter values we sample from the following conditional

$$\pi(\Theta|Y, D) \propto \pi(\Theta) \left[\prod_{i=0}^{n-1} \pi(Y^{i+1}|Y^i, \Theta)\right] \left[\prod_{i\in\{0,m,\ldots,n\}} \pi(D^i|Y^i, \Theta)\right]. \tag{31}$$

Since $\sigma_1, \ldots, \sigma_k$, like the rate constants, need to be positive, we use a Gaussian random walk on the logarithms as a proposal density for all parameters.

When sampling the latent process conditional on the current parameter values it is now needed to sample on times, where observations were made, too. Therefore we have to distinguish four different cases:

- for Y^i, with i not being a multiple of m the full conditional is

$$\pi(Y^i|Y^{i-1}, Y^{i+1}, \Theta, D) \propto \pi(Y^i|Y^{i-1}, \Theta)\pi(Y^{i+1}|Y^i, \Theta) \tag{32}$$

- for Y^i, with i being a multiple of m, but not 0 or n

$$\pi(Y^i|Y^{i-1}, Y^{i+1}, \Theta, D) \propto \pi(Y^i|Y^{i-1}, \Theta)\pi(Y^{i+1}|Y^i, \Theta)\pi(D^i|Y^i, \Theta) \tag{33}$$

- for Y^0

$$\pi(Y^0|Y^1, \Theta, D) \propto \pi(Y^0)\pi(Y^1|Y^0, \Theta)\pi(D^0|Y^0, \Theta) \tag{34}$$

- for Y^n

$$\pi(Y^n|Y^{n-1}, \Theta, D) \propto \pi(Y^n|Y^{n-1}, \Theta)\pi(D^n|Y^n, \Theta) \tag{35}$$

As a proposal an Gaussian random walk update for the first and last measurement is used. For the rest of the simulations the proposal density is the same as for the exact measurements in Section 5.1.

5.3 Alternating Block Updates

While the methods described in Section 5.1 and 5.2 were taken from Golightly and Wilkinson (2008) with small adaptations, the following approach tries to enhance the speed of computation and give better results than the former methods. The idea to the alternating block updates comes from the fact, that the conditionals in (32) and (33), as well as the used proposal, only depend on the neighboring data points Y^{i-1} and Y^{i+1}. The sampling from the latent data, conditional on the parameters is now split into two parts. First the algorithm samples for all Y^i with even i, then for odd i. Inside both of these steps, all iterations can be done in parallel because they only depend on the latent data from the other step. This should enable large speed improvements on multi-core systems.

6 Results

6.1 Standard Single-Site Update

Figure 2 and 3 show the result of a standard run of the algorithm. Like all following results the data was simulated with starting values $Y_{=}(20, 7)'$, parameters $c_1 = 0.05, c_2 = 0.01, c_3 = 0.01, c_4 = 0.05, \sigma_1^2 = \sigma_2^2 = 1$ and the simulation ended if $t \geq 20$. The prior distribution over the parameters were

$$\pi(c_i) = 1 \text{ for } 0.0067 \approx e^{-5} \leq c_i \leq e^{-1} \approx 0.37$$
$$\pi(\sigma_i) = 1 \text{ for } 0.0067 \approx e^{-5} \leq \sigma_i \leq e^{1} \approx 2.7$$

An uninformed prior was used for the starting value Y^0. Simulations ran for 200,000 cycles each. The first 50,000 were dropped as burn-in and the rest was thinned by a factor of 1000. In order to even out a possible bias in a dataset, the posterior means of the parameters were averaged over the inference on three different datasets.

It is notable that the algorithm tries to balance out the observations error in Figure 2, but fails to predict the latent data between observations better than the initial linear interpolation.

As a measure of quality we use the square error between the true parameter values and the posterior mean. Figure 4 and shows the dependence on m for each rate constant alone, while Figure 5 shows the dependence for all rate constants by using the Euclidean distance between the vector of the true rate constants and the posterior mean vector. For the each parameter on its own the dependence cannot be clearly seen, but it is obvious that the Euclidean distance declines with m. Because of this for further analysis only

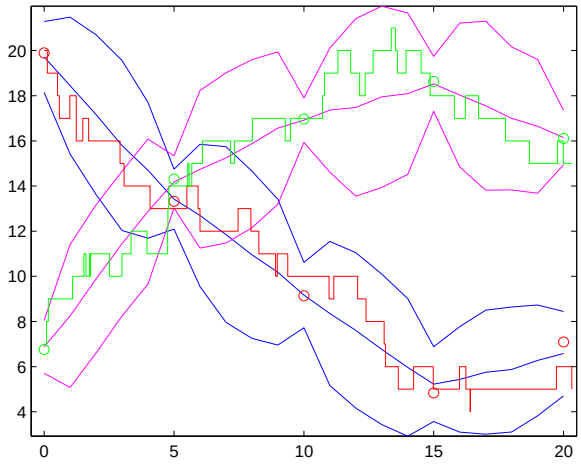

Figure 2: *Plot of the posterior of the latent data. Circles represent measurements, the green and red lines are the real process data, while the magenta and blue lines are the mean of the simulation, with the double standard derivation added as confidence interval.*

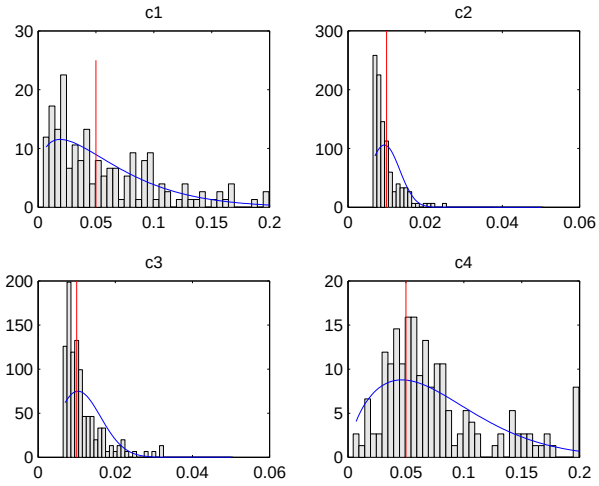

Figure 3: *Plot of the posterior of the parameters. The red line shows the real value of the parameter, the blue line is a Weibull distribution fitted to the histogram data with maximum likelihood.*

the Euclidean distance is used. The link between m and the quality of the prediction can be seen even more clearly, when σ is set higher (Fig. 6), this is probably due to the very low error of the posterior mean for $\sigma = 1$. Another important fact is, that the algorithm doesn't improve significantly for $m \geq 5$.

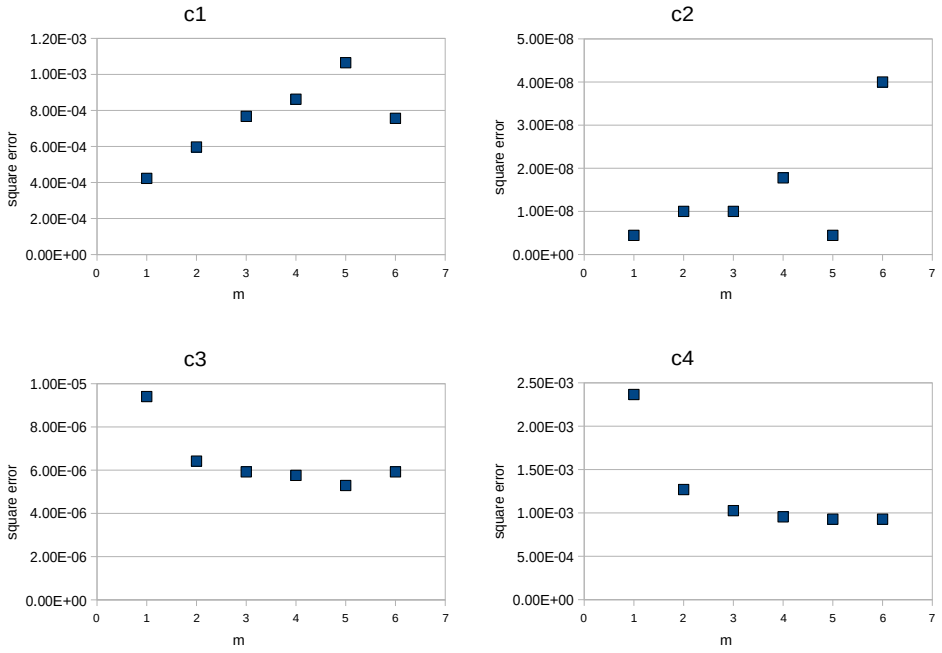

Figure 4: *Dependence on m for $\sigma = 1$ and 5 measurements, for each rate constant*

Not surprisingly more measurements also improve the results, which can be seen in Figure 7, despite an outlier for 15 measurements. But it has to be considered, that under realistic conditions, it might not be easy to increase the number of measurements. On the other hand, increasing m is just a matter of computational power.

Likewise, it is obvious that a higher standard derivation of the noise, reduces the accuracy of the algorithm. This can be seen in Figure 8.

The noise variance σ_1^2 and σ_2^2 was learned by the algorithm and clearly the predicted values depend on the true parameters used for the simulation (see Figure 9). Apart from this slight dependence, the algorithm doesn't succeed in estimating the real values.

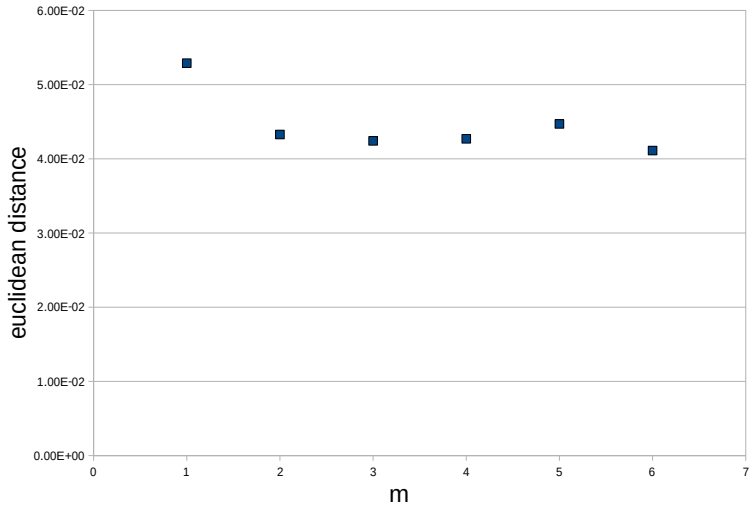

Figure 5: *Dependence on m for $\sigma = 1$ and 5 measurements*

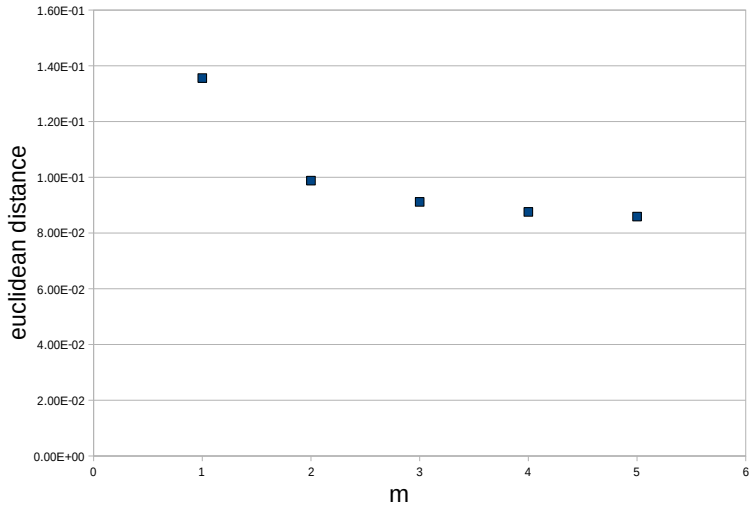

Figure 6: *Dependence on m for $\sigma = 2$ and 5 measurements*

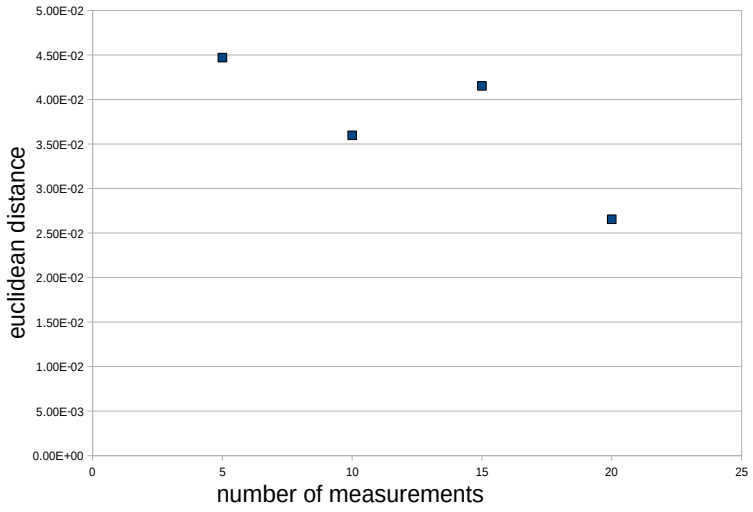

Figure 7: *Dependence on the number of measurements for* $\sigma = 1$ *and* $m = 5$

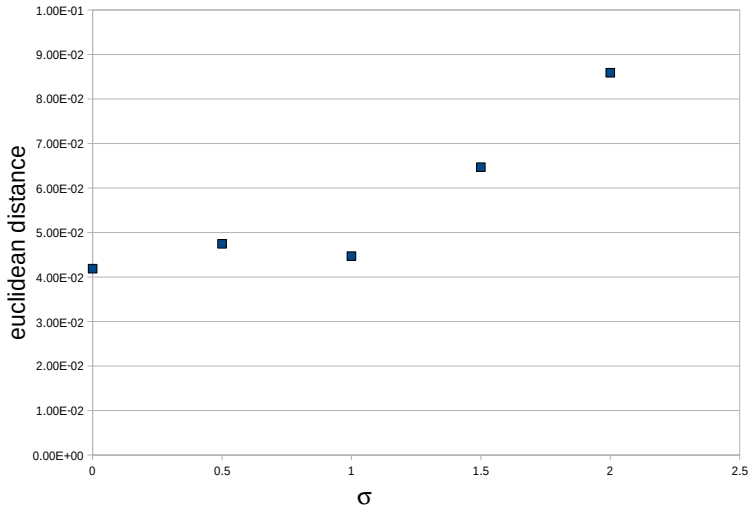

Figure 8: *Dependence on* σ *for* $m = 5$ *and 5 measurements.*

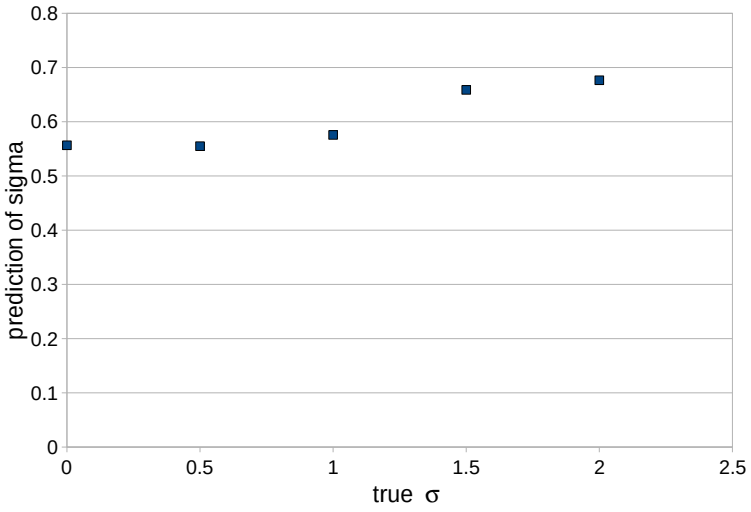

Figure 9: *Prediction of σ for $m = 5$ and 5 measurements.*

6.2 Alternating Block Updates

The method of alternating between sampling from the odd and even indexed data described in Section 5.3 promises faster computation on modern multi-core processor systems. Naturally this shouldn't come at the expense of quality. Figure 10 shows the results of this algorithm alongside with the standard approach. For higher m it seems like the error rises, but this could just be coincidence, since only two out of three parameter configurations show this behavior. Additionally the results for $m = 5$ are still pretty accurate as Figure 11 and 12 show. One advantage over the standard approach seem to be the better prediction of the latent process. Figure 11 shows that in between observations the algorithm predicts the process with curves, while single-site updating showed not much more then the initial linear interpolation.

Unfortunately the advantage of using a multi-core processor couldn't be explored in detail, but even without this, the algorithm is considerably faster ($\approx 10\%$ for $m = 10$) than the standard approach because it can be vectorized more easily. This advantage could be lost when using a software which isn't optimized for vectorization (MATLAB® was used for this project).

7 Discussion

The results showed, that even with very few observations good predictions can be made. With higher m and more measurements the error can be

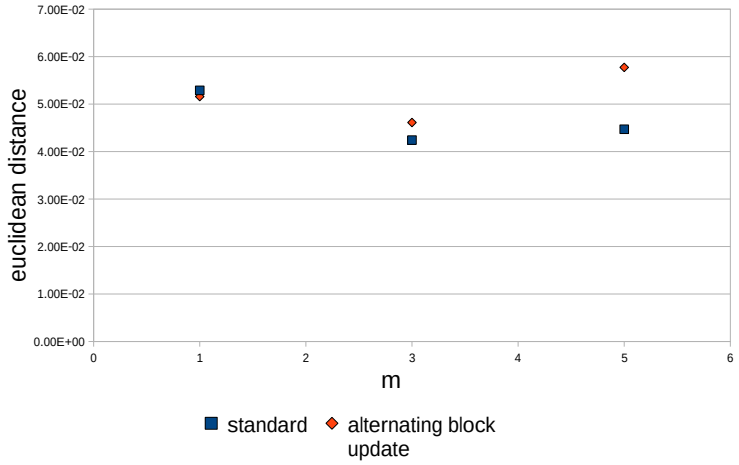

Figure 10: *Comparison of the prediction error for single-site update and alternating block update($\sigma = 1, m = 5$ and 5 measurements).*

further reduced, but values for m over 5 don't seem to improve the algorithm notably. This can be due to the rather simple Lotka-Volterra model because Golightly and Wilkinson (2005) used a slightly more complex model and registered considerably better results with $m = 10$.

The new approach of an alternating block update showed a significant speed improvement with a promising option for multi-core processor systems. Also this algorithm made better predictions of the latent process but may be slightly less accurate for inference on the parameters.

The posterior mean was used for most of the analysis, but Figure 3 and 12 show that the posterior maximum is more accurate in most cases and some of the results could be affected by this. Aside from this the Euclidean distance reduces the effect of the parameters, which have smaller values. This could be counteracted by a weighted error, with e.g. the reciprocal of the true value as weight.

Since the simple single-site update used in this paper is known to face correlation problems with high m (see Golightly and Wilkinson (2008)) more cycles and a higher thinning factor might have been more appropriate but this would have lead to very long computational times. Additionally a significant increase of the autocorrelation couldn't be observed for the domain of m used in this report.

For future research a broader prior over the parameters should be used to see how dependent the algorithm is on this information. Additionally

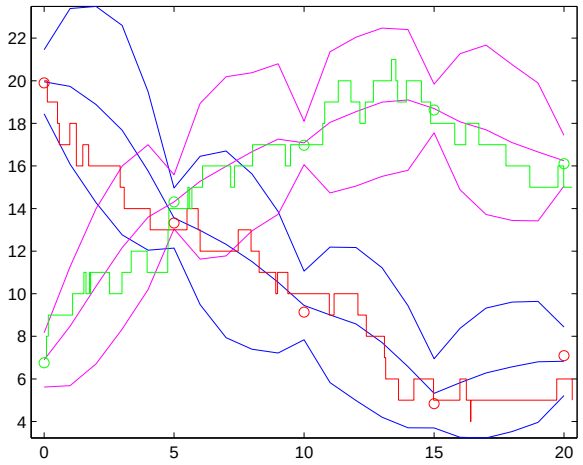

Figure 11: *Plot of the posterior of the latent data from the alternating block update algorithm. Circles represent measurements, the green and red lines are the real process data, while the magenta and blue lines are the mean of the simulation, with the double standard derivation added as confidence interval.*

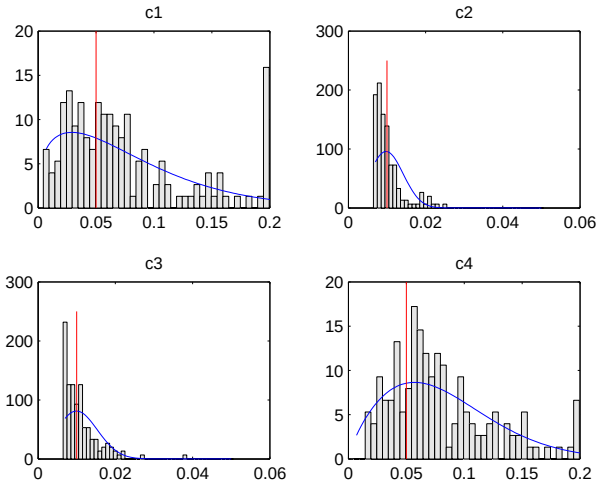

Figure 12: *Plot of the posterior of the parameters from the alternating block update algorithm. The red line shows the real value of the parameter, the blue line is a Weibull distribution fitted to the histogram data with maximum likelihood.*

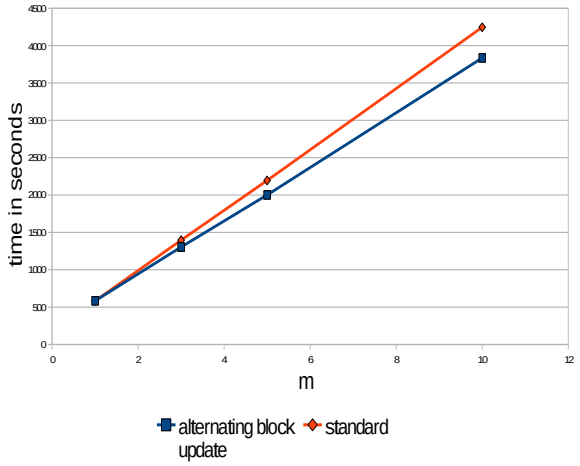

Figure 13: *Comparison of the computation time for single-site update and alternating block update*

priors, which aren't constant could be applied to study the outcome. Another interesting experiment would be to use a fairly high number of observations and strong observation noise. A wide field for further research would be to compare the effects of different proposals for the latent data points. Golightly and Wilkinson (2008) use the modified diffusion bridge but a proposal, which incorporates not only the neighboring data might deliver better results.

References

Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

Daniel T. Gillespie. A rigorous derivation of the chemical master equation. *Physica A: Statistical and Theoretical Physics*, 188(1-3):404–425, September 1992.

A. Golightly and D. J. Wilkinson. Bayesian inference for stochastic kinetic models using a diffusion approximation. *Biometrics*, 61(3):781–788, 2005.

A. Golightly and D. J. Wilkinson. Bayesian inference for nonlinear multivariate diffusion models observed with error. *Computational Statistics & Data Analysis*, 52(3):1674–1693, Jan 2008.

N. G. Van Kampen. *Stochastic Processes in Physics and Chemistry, Third Edition (North-Holland Personal Library)*. North Holland, April 2007.