Stat Comput (2012) 22:513–526
DOI 10.1007/s11222-011-9244-1

Rate estimation in partially observed Markov jump processes with measurement errors

Michael Amrein · Hans R. Künsch

Received: 13 July 2010 / Accepted: 3 March 2011 / Published online: 26 March 2011

Abstract We present a simulation methodology for Bayesian estimation of rate parameters in Markov jump processes arising for example in stochastic kinetic models. To handle the problem of missing components and measurement errors in observed data, we embed the Markov jump process into the framework of a general state space model. We do not use diffusion approximations. Markov chain Monte Carlo and particle filter type algorithms are introduced which allow sampling from the posterior distribution of the rate parameters and the Markov jump process also in data-poor scenarios. The algorithms are illustrated by applying them to rate estimation in a model for prokaryotic auto-regulation and the stochastic Oregonator, respectively.

Keywords Bayesian inference · General state space model · Markov chain Monte Carlo methods · Markov jump process · Particle filter · Stochastic kinetics

1 Introduction

It is generally accepted that many important intracellular processes, e.g. gene transcription and translation, are intrinsically stochastic, because chemical reactions occur at discrete times as results from random molecular collisions (McAdams and Arkin 1997; Arkin et al. 1998). These stochastic kinetic models correspond to a Markov jump process and can thus be simulated using techniques such as the Gillespie algorithm (Gillespie 1977) or—in the time-inhomogeneous case—Lewis' thinning method (Ogata 1981). Many of the parameters in such models are uncertain or unknown, therefore one wants to estimate them from times series data. One possible approach is to approximate the model with a diffusion and then to perform Bayesian (static or sequential) inference based on the approximation (see Golightly and Wilkinson 2005, 2006, 2008, 2009). This gives more flexibility to generate the proposals (see Durham and Gallant 2002), but it is difficult to quantify the approximation error. Depending on the application, it might be preferable to work with the original Markov jump process. This possibility is mentioned in Wilkinson (2006), Chap. 10, and Boys et al. (2008) demonstrate in the case of the simple Lotka-Volterra model that this approach is feasible in principle. But in more complex situations it is difficult to construct a Markov chain Monte Carlo (MCMC) sampler with good mixing properties. The key problems in our view are to construct good proposals for the latent process on an interval when the values at the two end points are fixed and the process is close to the boundary of the state space, and to construct reasonable starting values for the process and the parameters, in particular when some of the components are observed with small or zero noise. We propose here solutions for both of these problems that go beyond Wilkinson (2006), Chap. 10, and Boys et al. (2008) and thus substantially enlarge the class of models that are computationally tractable.

The rest of the paper is organized as follows. In Sect. 2, we describe the model, establish the relation to stochastic kinetics and introduce useful notation and densities. In Sect. 3, we motivate the Bayesian approach and present the base frame of the MCMC algorithm. Section 4 describes in detail certain aspects of the algorithm, mainly the construction of proposals for the latent Markov jump process. In Sect. 5, the particle filter type algorithm to initialize values for the parameters and for the latent Markov jump process

M. Amrein (✉) · H.R. Künsch
Seminar für Statistik, ETH Zürich, Rämistrasse 101, 8092 Zürich, Switzerland
e-mail: amrein@stat.math.ethz.ch

is presented. In Sect. 6, we look at two examples. First, the stochastic Oregonator (see Gillespie 1977) is treated in various scenarios, including some data-poor ones, to show how the algorithm works. Then, we turn to a model for prokaryotic auto-regulation introduced in Golightly and Wilkinson (2005) and reconsidered in Golightly and Wilkinson (2009). Finally, conclusions are given in Sect. 7.

2 Setting and definitions

2.1 Model

Consider a Markov jump process

$$\mathcal{Y} = \{y_t = (y_t^1, \dots, y_t^p)^T : t \geq t_0\}$$

on a state space $\mathcal{E} \subset \mathbb{N}_0^p$ with jump vectors $A_i \in \mathbb{Z}^p$ for $i \in \{1, \dots, r\}$ and possibly time dependent transition intensities $\mu_i(t, y) = \theta_i \cdot h_i(t, y)$:

$$\mathrm{P}[y_{t+\delta} = y + A_i | y_t = y] = \mu_i(t, y)\delta + o(\delta) \quad (\delta > 0).$$

We denote the total transition intensity by

$$\mu_0(t, y) = \sum_{i=1}^{r} \mu_i(t, y).$$

We assume that the functions $h = \{h_i\}_{i \in \{1,2,\dots,r\}}$, called the standardized transition intensities, the jump matrix A with columns A_i and the initial distribution f_0 of y_{t_0} are known. The goal is to estimate the hazard rates $\theta = (\theta_1, \dots, \theta_r)$ from partial measurements $x_0, x_1, \dots, x_n$ of the process at discrete time points $0 = t_0 < t_1 < \cdots < t_n$. Unobserved components are set to `na` and we assume

$$x_l | \mathcal{Y} = x_l | y_{t_l} \sim g_\eta(.|y_{t_l}),$$

where $g_\eta(x_l | y_{t_l})$ is a density with respect to some σ-finite measure (with possibly unknown) nuisance parameter η. We specify this more precisely in the examples in Sect. 6.

If a row in the matrix A a is a linear combination of the others, say

$$A_{lj} = \sum_{i \neq l}^{p} \lambda_i A_{ij} \quad \forall j \in \{1, \dots, r\},$$

then

$$y_t^l - \sum_{i \neq l}^{p} \lambda_i y_t^i = \text{const} \quad \forall t.$$

Throughout the article, we assume these conservation constants to be known. Therefore, we can remove y^l from the system and assume in the following that $\operatorname{rank}(A) = p \leq r$.

This framework can be regarded as a general state space model: x_0, $x_1, \dots$ is an observed times series which is derived from the unobservable Markov chain y_{t_0}, $y_{t_1}, \dots$ (see Künsch 2000 or Doucet et al. 2001).

For computational reasons, we further assume that we can easily evaluate the time-integrated standardized transition intensities

$$H_i(s, t, y) := \int_s^t h_i(u, y) du.$$

Models of the above form arise for example in the context of stochastic kinetics. Consider a biochemical reaction network with r reactions $R_1, R_2, \dots, R_r$ and p species $Y^1, Y^2, \dots, Y^p$, i.e.,

$$R_i : \; v_{i1} Y^1 + \cdots + v_{ip} Y^p \longrightarrow u_{i1} Y^1 + \cdots + u_{ip} Y^p$$

for $i = 1, \dots, r$. Let y_t^j denote the number of species Y^j at time t, $y_t = (y_t^1, \dots, y_t^p)^T$, $V = (v_{ij})$ and $U = (u_{ij})$. Then, according to the mass action law, we can describe $\{y_t : t \geq t_0\}$ as a Markov jump process with jump matrix $A = (U - V)^T$ and standardized reaction intensities

$$h_i(y) = \prod_{j, v_{ij} \geq 1} \binom{y^j}{v_{ij}} \quad \text{where } \binom{y}{v} = 0 \text{ when } y < v.$$

For further details, see e.g. Gillespie (1977), Golightly and Wilkinson (2005) or Golightly and Wilkinson (2009). We will use in the following terminology from this applications: We will call the jump times reaction times and classify a jump as one of the r possible reaction types.

2.2 Additional notation and formulae for densities

A possible path $y_{[a,b]}$ on an interval $[a, b]$ in our model is uniquely characterized by the total number of reactions n_{tot}, the initial state y_a, the successive reaction times $a < \tau_1 < \cdots < \tau_{n_{tot}} \leq b$ and the reaction types (or indices) $r_1, r_2, \dots, r_{n_{tot}} \in \{1, \dots, r\}$. The states at the reaction times are then obtained as

$$y_{\tau_k} = y_a + \sum_{i=1}^{k} A_{r_i}.$$

We write for simplicity y_k instead of y_{τ_k}. Furthermore, r_{tot}^i is the total number of reactions of type i and r_{tot} is the vector with components r_{tot}^i. All these quantities depend on the interval $[a, b]$. If this interval is not clear from the context, we write $n_{tot}([a, b])$, $\tau_k([a, b])$, etc.

The density ψ_θ of $y_{[a,b]}$ given y_a (with respect to the Lebesque measure for the reaction times and the counting measure for the reaction types) is well known, see e.g.

Wilkinson (2006), Chap. 10. Defining $\tau_0 = a$, $\tau_{n_{tot}+1} = b$ and $y_{\tau_0} = y_0 = y_a$, it is given by

$$\begin{aligned}&\exp\left(-\sum_{i=1}^{r}\theta_i\int_a^b h_i(s,y_s)ds\right)\\&\quad\cdot\prod_{k=1}^{n_{tot}}\theta_{r_k}h_{r_k}(\tau_k,y_{k-1})\\&=\exp\left(-\sum_{k=1}^{n_{tot}+1}\sum_{i=1}^{r}\theta_i H_i(\tau_{k-1},\tau_k,y_{k-1})\right)\\&\quad\cdot\prod_{k=1}^{n_{tot}}\theta_{r_k}h_{r_k}(\tau_k,y_{k-1}).\end{aligned}$$

In the time-homogeneous case, i.e., $h_i(t,y) = h_i(y)$, we have $H_i(\tau_{k-1},\tau_k,y_{k-1}) = h_i(y_{k-1})\delta_k$ with $\delta_k = \tau_k - \tau_{k-1}$. Therefore

$$\delta_k|\tau_{k-1}, y_{k-1} \sim \text{Exp}(\mu_0(y_{k-1})) \tag{1}$$

and

$$\text{P}[r_k = i|\tau_{k-1}, y_{k-1}] = \frac{\mu_i(y_{k-1})}{\mu_0(y_{k-1})}, \tag{2}$$

and we can exactly simulate the Markov jump process using the Gillespie algorithm (see Gillespie 1977) or some faster versions thereof (see Gibson and Bruck 2000). Replacing $h_i(y_{k-1})$ by $h_i(\tau_{k-1}, y_{k-1})$ in (1) and (2), this can be done "approximately" in the inhomogeneous case. An exact simulation algorithm based on a thinning method is described in Ogata (1981).

We write the density of all observations in $[a,b]$ as

$$g_\eta(x_{[a,b]}|y_{[a,b]}) = \prod_{l,a\le t_l\le b} g_\eta(x_l|y_{t_l}),$$

where the empty product is interpreted as 1. The joint density $p(y_{[t_0,t_n]}, x_{[t_0,t_n]}|\theta,\eta)$ of $y_{[t_0,t_n]}$ and $x_{[t_0,t_n]}$ (given the parameters θ and η) is then

$$f_0(y_0)\cdot\psi_\theta(y_{[t_0,t_n]}|y_0)\cdot g_\eta(x_{[t_0,t_n]}|y_{[t_0,t_n]}). \tag{3}$$

3 Bayesian estimation and Monte Carlo methods

The maximum likelihood estimator is too complicated to compute because we are not able to calculate the marginalisation of the density in (3) over $y_{[t_0,t_n]}$ explicitly. It seems easier to combine a Bayesian approach with Monte Carlo methods, that is to sample from the posterior distribution of the parameters and the underlying Markov jump process $y_{[t_0,t_n]}$ given the data (see Robert and Casella 2004). This has also the additional advantage that prior knowledge about the reaction rates can be used. Assuming θ and η to be independent a priori, the joint distribution of $y_{[t_0,t_n]}$, $x_{[t_0,t_n]}$, θ and η has the form

$$p(y_{[t_0,t_n]}, x_{[t_0,t_n]}, \theta, \eta) = p(y_{[t_0,t_n]}, x_{[t_0,t_n]}|\theta,\eta)\cdot p(\theta)\cdot p(\eta).$$

We want to simulate from the conditioned density

$$p(y_{[t_0,t_n]}, \theta, \eta|x_{[t_0,t_n]}),$$

which yields also samples from $p(\theta,\eta|x_{[t_0,t_n]})$ using a marginalisation over $y_{[t_0,t_n]}$. The standard approach to do this is iterating between blockwise updates of the latent process $y_{[t_0,t_n]}$ on subintervals of $[t_0,t_n]$ with Metropolis-Hastings steps, updates of θ and updates of η (see e.g. Gilks et al. 1996, Chap. 1; Boys et al. 2008 or Golightly and Wilkinson 2009).

As in Boys et al. (2008), we choose independent Gamma distributions with parameters α_i and β_i as priors for θ_i:

$$p(\theta) \propto \prod_{i=1}^{r}\theta_i^{\alpha_i-1}\exp(-\beta_i\theta_i).$$

We write this distribution as $\Gamma_r(\alpha,\beta)$ where α and β are vectors of dimension r. Conditionally on $y_{[t_0,t_n]}$,$x_{[t_0,t_n]}$ and η, the components θ_i have then again independent Gamma distributions, more precisely

$$\begin{aligned}\theta|y_{[t_0,t_n]}, x_{[t_0,t_n]}, \eta &\sim \theta|y_{[t_0,t_n]}\\&\sim \Gamma_r\left(\tilde\alpha(y_{[t_0,t_n]}), \tilde\beta(y_{[t_0,t_n]})\right),\end{aligned} \tag{4}$$

with

$$\tilde\alpha_i(y_{[t_0,t_n]}, \alpha_i) = \alpha_i + r_{tot}^i$$

and

$$\begin{aligned}\tilde\beta_i(y_{[t_0,t_n]}, \beta_i) &= \beta_i + \int_{t_0}^{t_n} h_i(s,y_s)ds\\&= \beta_i + \sum_{k=1}^{n_{tot}+1} H_i(\tau_{k-1},\tau_k,y_{k-1}).\end{aligned}$$

Choosing a suitable prior for η depends heavily on the error distribution, so we refer to the examples in Sect. 6.

We propose the following algorithm, which will be explained in more detail in the next sections. The generation of initial values $y_{[t_0,t_n]}^{(0)}$, $\theta^{(0)}$ and $\eta^{(0)}$ will be discussed in Sect. 5. The choice of the set $\mathcal{I}_{[t_0,t_n]}$ of overlapping subintervals $[a,b]\subset[t_0,t_n]$ for updating y will be discussed in Sect. 4.4.

Algorithm 1 (Simulation from $y_{[t_0,t_n]}, \theta, \eta|x_{[t_0,t_n]}$)
For $m = 1, 2, \ldots, M$:

1. Set $y_{[t_0,t_n]} = y^{(m-1)}_{[t_0,t_n]}$, $\theta = \theta^{(m-1)}$, $\eta = \eta^{(m-1)}$. Update $y_{[a,b]}$ for all $[a,b] \in \mathcal{I}_{[t_0,t_n]}$ sequentially in a random order by proposing $y^{new}_{[a,b]}$ as described in Sects. 4.1, 4.2, 4.5 and 4.6 and replacing $y_{[a,b]}$ by $y^{new}_{[a,b]}$ with probability $\alpha(y^{new}_{[a,b]}|y_{[a,b]}, \theta, \eta)$ (see (12)). Set $y^{(m)}_{[t_0,t_n]} = y_{[t_0,t_n]}$.
2. Simulate $\theta^{(m)} \sim \Gamma_r(\tilde{\alpha}(y^{(m)}_{[t_0,t_n]}), \tilde{\beta}(y^{(m)}_{[t_0,t_n]}))$.
3. Generate $\eta^{(m)}$ given $y^{(m)}_{[t_0,t_n]}$ in a suitable fashion.

4 Simulating a path given parameters and observations

We assume now that θ and η are fixed and we want to modify $y_{[a,b]}$ on subintervals $[a,b]$ of $[t_0,t_n]$. First we consider the case $t_0 < a < b < t_n$ where the values y_a and y_b remain unchanged. The boundary cases will be discussed in Sect. 4.6. Exact methods to simulate from a continuous time Markov chain conditioned on both endpoints are reviewed and discussed in Hobolth and Stone (2009). The rejection method is too slow in our examples, and direct sampling and uniformization require finite state space and eigendecompositions of the generator matrix. This would require truncating the state space and is too time-consuming in our examples. Hence we use a Metropolis-Hastings procedure. Our proposal distribution q first generates a vector of new total reaction numbers r^{new}_{tot} on $[a,b]$ and then, conditioned on r^{new}_{tot}, generates a value $y^{new}_{[a,b]}$.

4.1 Generating new reaction totals

Because the values y_a and y_b are fixed, we must have that

$$Ar^{new}_{tot} = y_b - y_a = Ar_{tot} \Leftrightarrow A(r^{new}_{tot} - r_{tot}) = 0. \tag{5}$$

If $p = r \Leftrightarrow \text{rank}(A) = r$, A is invertible and the reaction totals remain unchanged. Otherwise it is known that $\text{kernel}(A) := \{x \in \mathbb{Z}^r : A \cdot x = 0\}$ forms a lattice and can be written as $\{a_1 \cdot v_1 + \cdots + a_d \cdot v_d : a_1, \ldots, a_d \in \mathbb{Z}\}$ with $d = r - p$ and basis vectors $v_l \in \mathbb{Z}^r$, $l \in \{1, 2, \ldots, d\}$ (note that these vectors are not unique). Appendix describes how to compute a basis vector matrix

$$V(A) = (v_1, \ldots, v_d).$$

This enables us to generate a vector r^{new}_{tot} which respects (5) in a simple way:

$$r^{new}_{tot} = r_{tot} + V(A) \cdot Z, \; Z \sim q^Z_\iota, \tag{6}$$

where q^Z_ι is a symmetric proposal distribution q^Z_ι on $\mathbb{Z}^d$, i.e., $q^Z_\iota(z) = q^Z_\iota(-z)$, with parameter ι. If r^{new}_{tot} has a negative component, we stop and set $y^{new}_{[a,b]} = y_{[a,b]}$.

4.2 Generating a new path given the reaction totals

The new path $y^{new}_{[a,b]}$ depends only on y_a and the new reaction totals r^{new}_{tot}, and not on the old path $y_{[a,b]}$. The constraint $y^{new}_b = y_b$ is satisfied automatically by our construction of r^{new}_{tot}. Therefore our algorithm simply generates a path on $[a,b]$ with given initial value and given reaction totals, and we can omit the superscripts *new*.

A first possibility is to generate the path according to r independent inhomogeneous Poisson processes with intensities

$$\lambda_i(t) = \mu_i(a, y_a)\frac{b-t}{b-a} + \mu_i(b, y_b)\frac{t-a}{b-a}, \tag{7}$$

conditioned on the totals $r^{new,i}_{tot}$ as in Boys et al. (2008) for the simple Lotka-Volterra reaction system. In many cases this leads to proposals which approximate the true jump process nicely. But in situations where the standardized reaction intensities h_i depend strongly on y, this proposal often generates paths that are impossible under the model. This is typically the case when the number of molecules of some species is small. In order to address this problem, we constructed the following proposal which first decides the order in which the reactions take place, that is we first generate r_k for $k = 1, 2, \ldots, n_{tot}$. In a second step, we generate the reaction times τ_k. In the first step, we take into account both the possibility of a reaction of a given type to occur at the current state of the process and the remaining number of reactions S^i_k of type i after time τ_k that still have to occur in order to reach the prescribed total. In the second step, we take the values of the intensities into account. In order to make the description of the algorithm easier to read, we mention that t^*_k is a first guess for τ_{k-1} (needed only if the intensities are time-inhomogeneous). Also remember that $y_k = y_{\tau_k}$.

Algorithm 2 (Proposing path $y_{[a,b]}$ given r_{tot}, y_a)

1. Set $S^i_0 = r^i_{tot}$ for $i \in \{1, \ldots, r\}$ and $y_0 = y_a$.
2. For $k = 1, \ldots, n_{tot}$ do the following:
   Set $t^*_k = a + (b-a)(k-1)/n_{tot}$. If $\mu_l(t^*_k, y_{k-1}) = 0$ for all l with $S^l_{k-1} > 0$, stop. Otherwise, generate r_k with probabilities

   $$\text{P}[r_k = i] \propto \mathbb{I}_{(0,\infty)}(\mu_i(t^*_k, y_{k-1})) \cdot S^i_{k-1}. \tag{8}$$

   If $r_k = i$, set $S^i_k = S^i_{k-1} - 1$, $S^l_k = S^l_{k-1}$ for $l \neq i$ and $y_k = y_{k-1} + A_i$.
3. Generate $(\delta_k; k \in \{1, \ldots, n_{tot}+1\})$ according to a Dirichlet distribution with parameter $\alpha = (\alpha_k; k \in \{1, \ldots, n_{tot}+1\})$ where

   $$\alpha_k = \mu_0^{-1}(t^*_k, y_{k-1})\frac{\sum_l \mu_0^{-1}(t^*_l, y_{l-1})}{\sum_l \mu_0^{-2}(t^*_l, y_{l-1})}, \tag{9}$$

   and set $\tau_k = \tau_{k-1} + (b-a)\delta_k$ for $k = 1, \ldots, n_{tot}$.

The algorithm stops in step 2 when we can no longer reach the state y_b on a possible reaction path using the available remaining reactions. This means that an impossible path is proposed which has acceptance probability 0.

The heuristics behind the steps in the above algorithm is the following. The probabilities (8) are an attempt to ensure that a reaction of type i at the current state is possible according to the law of the process and we nevertheless reach the prescribed reaction total. Of course, there are many other possibilities to define the probabilities in (8), e.g. the geometric mean

$$\sqrt{S^i_{k-1}\mu_i(t^*_k, y_{k-1})}.$$

Empirically, we found that the above variant leads to good acceptance rates in the examples in Sect. 6.

The Dirichlet distribution in (9) is used to approximate the distribution of independent $\text{Exp}(\mu_0(t^*_k, y_{k-1}))$ waiting times δ_k conditioned on the event that their sum is equal to $b-a$ based on the following considerations. If all $\mu_0(t^*_k, y_{k-1})$ are equal, the conditional first two moments are

$$\mathrm{E}\left[\delta_k \,\middle|\, \sum_l \delta_l = b-a\right] = (b-a)\frac{\mathrm{E}[\delta_k]}{\sum_l \mathrm{E}[\delta_l]} \tag{10}$$

and

$$\begin{aligned}\mathrm{Var}&\left[\delta_k \,\middle|\, \sum_l \delta_l = b-a\right]\\ &= (b-a)^2\\ &\quad\times\left(\frac{\mathrm{Var}(\delta_k)+\mathrm{E}[\delta_k]^2}{\sum_l \mathrm{Var}(\delta_l)+(\sum_l \mathrm{E}[\delta_l])^2} - \left(\frac{\mathrm{E}[\delta_k]}{\sum_l \mathrm{E}[\delta_l]}\right)^2\right),\end{aligned} \tag{11}$$

and moreover the conditional distribution is Dirichlet with parameters $\alpha_k = 1$, scaled by $b-a$, see e.g. Bickel and Doksum (1977), Sect. 1.2. In the general case, we use a Dirichlet distribution as approximation and determine the parameters such that the expectation matches the right-hand side of (10) for all k. This implies that

$$\alpha_k \propto \mu_0^{-1}(t^*_k, y_{k-1}).$$

Finally, the proportionality factor is determined such that the sum of the variances matches the sum of the right-hand sides of (11). Note that there is an exact simulation method (see Fearnhead and Meligkotsidou 2004), but it is computationally much more expensive.

4.3 Acceptance probability of a new path

By construction, the proposal density $q(y^{new}_{[a,b]}|y_{[a,b]}, \theta)$ has the form

$$q(y^{new}_{[a,b]}|y_a, r^{new}_{tot}, \theta)q(r^{new}_{tot}|r_{tot}).$$

Because of the symmetry of q^Z_t, we have

$$q(r^{new}_{tot}|r_{tot}) = q(r_{tot}|r^{new}_{tot}).$$

So it will cancel out in the acceptance probability and we do not need to consider it.

Next, $q(y_{[a,b]}|y_a, r_{tot}, \theta)$ is equal to

$$\prod_{k=1}^{n_{tot}} \frac{\mathbb{I}_{(0,\infty)}(\mu_i(t^*_k, y_{k-1}))S^i_{k-1}}{\sum_{l=1}^r \mathbb{I}_{(0,\infty)}(\mu_l(t^*_k, y_{k-1}))S^l_{k-1}} \times \frac{f^{\text{Dir}}_\alpha((\tau_k - \tau_{k-1})/(b-a) : k \in \{1, \ldots, n_{tot}+1\})}{(b-a)^{n_{tot}}}$$

where f^{Dir}_α is the density of the Dirichlet distribution with parameter α from (9).

Hence, according to the Metropolis-Hastings recipe, the acceptance probability $\alpha(y^{new}_{[a,b]}|y_{[a,b]}, \theta, \eta)$ is

$$\min\left\{1, \frac{\psi_\theta(y^{new}_{[a,b]}|y_a)g_\eta(x_{[a,b]}|y^{new}_{[a,b]})q(y_{[a,b]}|y_a, r_{tot}, \theta)}{\psi_\theta(y_{[a,b]}|y_a)g_\eta(x_{[a,b]}|y_{[a,b]})q(y^{new}_{[a,b]}|y_a, r^{new}_{tot}, \theta)}\right\}. \tag{12}$$

4.4 Choice of the subintervals $[a, b]$

To ensure that the process can be updated on the whole interval $[t_0, t_n]$, we have to choose a suitable set of subintervals $\mathcal{I}_{[t_0,t_n]}$ for which we apply the above updating algorithms. As a general rule, one can say that they should be overlapping. Also it is often useful to include subintervals which do not lead to a change of the process at the observation times $t_1 < t_2 < \cdots < t_n$. In such situations, the terms $g_\eta(x_{[a,b]}|y^{new}_{[a,b]})$ and $g_\eta(x_{[a,b]}|y_{[a,b]})$ are equal and therefore cancel out in the acceptance probability.

In cases where the observations are complete and noise-free, we need only the subintervals of the form $[t_{l-1}, t_l]$. However, because it is sometimes a non-trivial problem to find a realization of the Markov jump process which matches all observations, we found that it is sometimes useful to include a tiny noise in the model and to choose also subintervals with a t_l as interior point. By this trick we can often obtain realizations that match all observation by the above updating algorithms.

In general, good choices of the subintervals can be very dependent on the given situation. The standard one is to let $\mathcal{I}_{[t_0,t_n]}$ consists of all intervals of the form $[t_{l-1}, t_l]$ and $[(t_{l-1}+t_l)/2, (t_l+t_{l+1})/2]$.

4.5 Updating latent components

Let us assume that at the time t_l the j-th component is not observed, i.e., $x_l^j = \mathtt{na}$, but the others are. Especially when working with small or no noise, updating with the above proposal on $[a, b]$ where $a < t_l < b$ is problematic because of the following reason: Assume y_l already matches x_l nicely on the observed components. Then a new proposal can only be accepted when y_l^{new} matches the observed components, too. Thus not only the acceptance rate is low, but more severely, y_l^j remains usually unchanged although we do not have any information on y_l^j.

To circumvent this problem, we construct a proposal on the interval $[a, b]$, $a < t_l < b$, which generates simultaneously new reactions totals on the intervals $[a, t_l]$ and $[t_l, b]$, $r_{tot,a}^{new}$ and $r_{tot,b}^{new}$, respectively, so that the values y_{t_a}, y_{t_b} and the observed values of y_l remain fixed, but y_l^j can change. Similarly to (5), we consider solutions of

$$A_{-j,.}(r_{tot,a}^{new} - r_{tot,a}) = 0$$

where $A_{-j,.}$ denotes the reaction matrix without the j-th row. Because of the assumption that $\operatorname{rank}(A) = p \le r$, $\operatorname{kernel}(A_{-j,.})\backslash\operatorname{kernel}(A)$ is non-empty. Thus we can draw v from $\operatorname{kernel}(A_{-j,.})\backslash\operatorname{kernel}(A)$ in a symmetric manner, so that the reaction totals

$$r_{tot,a}^{new} = r_{tot,a} + v \quad \text{and} \quad r_{tot,b}^{new} = r_{tot,b} - v \tag{13}$$

fulfill the above requirements. To propose the jump processes on the two intervals $[a, t_l]$ and $[t_l, b]$, we can use the techniques from Sect. 4.2. The calculation of the acceptance probability is similar to the one described in Sect. 4.3. The symmetric proposal distribution for the vector v from $\operatorname{kernel}(A_{-j,.})\backslash\operatorname{kernel}(A)$ is specified for the examples considered in Sect. 6.

4.6 Updating the path at a border

In the cases $b = t_n$ or $a = t_0$ we also want to change the values of y_{t_n} and y_{t_0}, respectively (unless f_0 is a Dirac measure). We recommend to propose first a change in r_{tot}^{new}, that is

$$r_{tot}^{new} = r_{tot} + r', \qquad r' \sim q_{l'}^{r'}, \tag{14}$$

where $q_{l'}^{r'}$ is a symmetric distribution on $\mathbb{Z}^r$. Then either y_a or y_b remains unchanged and the other value follows from $y_b - y_a = Ar_{tot}^{new}$. The rest can be done again with Algorithm 2. If $y_{t_0}^{new} \neq y_{t_0}$, the factor $f_0(y_{t_0}^{new})/f_0(y_{t_0})$ is needed additionally in the acceptance probability (12).

If one wants to change only some components of y_{t_0} or y_{t_n}, respectively, the same ideas as in Sect. 4.5 can be used. For more details, see the examples in Sect. 6.

5 Initialisation of η, θ and $y_{[t_0,t_n]}$

The form of the trajectories of the underlying Markov jump process depends strongly on the parameter θ and the value at t_0. So just choosing $\eta^{(0)}$ and $\theta^{(0)}$ and then simulating $y_{[t_0,t_n]}^{(0)}$ leads usually to processes which match the observed data badly. It then takes very many iterations in the algorithm until we obtain processes that are compatible with the data.

In our experience, generating the starting values by Algorithm 3 below leads to substantial increases in computational efficiency. It is inspired by the particle filter: Instead of reweighting and resampling, we just select the most likely particle, perform a number of Metropolis-Hastings steps (similarly to Gilks and Berzuini 2001) and propagate with the Gillespie algorithm.

An additional trick can bring further improvement. Because the speed of the techniques described depends heavily on the number of reactions in the system, one wants to ensure that the initial value $y_{[t_0,t_n]}$ for Algorithm 1 has rather too few than too many reactions. We can achieve this with a simple shrinkage factor ν between 0 and 1 for θ during the initialisation, that is replacing θ after simulation with $\nu \cdot \theta$. This acts like a penalisation on the reaction numbers: It does not affect the probabilities in (2) for the time-homogeneous case, but makes the system slower, resulting in fewer reactions. But even in the inhomogeneous case, it may be useful.

Algorithm 3 (Generating starting values)

1. Choose $\eta^{(0)}$.
2. Simulate $S^{\{1\}}$ i.i.d. starting values $y_{t_0}^s \sim p(y_{t_0}|x_{t_0})$ and generate $y_{(t_0,t_1]}^s$ for $s \in \{1, 2, \ldots, S^{\{1\}}\}$ using the Gillespie algorithm with the normalized standardized reaction intensities $\mathbb{I}_{\{h_i>0\}}$ $(i = 1, \ldots, r)$ and equal hazard rates $1/(t_1 - t_0)$. Set $y_{[t_0,t_1]}^{\{1\}} = y_{[t_0,t_1]}^{s'}$, where $s' = \operatorname{argmax}_s\{g_{\eta^{(0)}}(x_{[t_0,t_1]}|y_{[t_0,t_1]}^s)\}$. Simulate $\theta^{\{1\}} \sim \Gamma_r(\tilde{\alpha}(y_{[t_0,t_1]}^{\{1\}}), \tilde{\beta}(y_{[t_0,t_1]}^{\{1\}}))$.
3. For $l = 1, \ldots, n-1$:
   a) Use $M^{\{l\}}$ steps of Algorithm 1 on $[t_0, t_l]$ with shrinkage factor ν and starting values $y_{[t_0,t_l]}^{\{l\}}$ and $\theta^{\{l\}}$ to generate $y_{[t_0,t_l]}^{\{l+1\}}$ and $\theta^{\{l+1\}}$.
   b) Generate $S^{\{l\}}$ paths $y_{[t_0,t_{l+1}]}^s$ which are independent continuations of $y_{[t_0,t_l]}^{\{l+1\}}$ on $(t_l, t_{l+1}]$, based on the Gillespie algorithm with $\theta^{\{l+1\}}$ and set $y_{[t_0,t_{l+1}]}^{\{l+1\}} = y_{[t_0,t_{l+1}]}^{s'}$, where s' is equal to $\operatorname{argmax}_s\{g_{\eta^{(0)}}(x_{l+1}|y_{t_{l+1}}^s)\}$.
4. Set $\theta^{(0)} = \theta^{\{n\}}$ and $y_{[t_0,t_n]}^{(0)} = y_{[t_0,t_n]}^{\{n\}}$.

So to propagate to the process on the interval $(t_l, t_{l+1}]$ (for $l = 1, \ldots, n-1$), we use $\theta^{\{l+1\}}$ which should roughly follow the distribution of θ given $x_{[t_0,t_l]}$ and $\eta^{(0)}$, because of step 3.a).

6 Examples

6.1 Stochastic Oregonator

First we consider the stochastic Oregonator to illustrate the algorithms. It is a highly idealized model of the Belousov-Zhabotinskii reactions, a non-linear chemical oscillator. It has 3 species and the following 5 reactions:

$$\begin{aligned} &R_1: Y^2 &&\longrightarrow Y^1\\ &R_2: Y^1+Y^2 &&\longrightarrow \emptyset\\ &R_3: Y^1 &&\longrightarrow 2Y^1+Y^3\\ &R_4: 2Y^1 &&\longrightarrow \emptyset\\ &R_5: Y^3 &&\longrightarrow Y^2 \end{aligned} \tag{15}$$

For further details, see Gillespie (1977). Following Sect. 2.1, the process $\{y_t : t \geq t_0\}$, where $y_t = (y_t^1, y_t^2, y_t^3)^T$ and y_t^i is the number of species Y^i at time t, is a Markov jump process with standardized reaction intensities

$$h(y) = (y^2, y^1y^2, y^1, y^1(y^1-1)/2, y^3)^T$$

and the jump matrix

$$A := \begin{pmatrix} 1 & -1 & 1 & -2 & 0\\ -1 & -1 & 0 & 0 & 1\\ 0 & 0 & 1 & 0 & -1 \end{pmatrix}.$$

As starting distribution f_0 for y_{t_0}, we use the uniform distribution on $\{0,\dots,K\}^3$ with $K = 25$. The measurement errors are normally distributed with precision η, that is

$$g_\eta(x,y) = \prod_{j:x^j\neq \mathtt{na}} \frac{\sqrt{\eta}}{\sqrt{2\pi}} \exp\left(-\frac{\eta}{2}(x^j-y^j)^2\right). \tag{16}$$

In Fig. 1, a sample trajectory for

$$\theta = (0.1, 0.1, 2.5, 0.04, 1),$$

simulated with the Gillespie algorithm, is shown, observed every 1/3 units of time during a time period of 20 units.

If we choose a Gamma(α,β) prior for η, then the full conditional posterior distribution of η is again a Gamma distribution with parameters $\tilde{\alpha}^\eta(x_{[t_0,t_n]},\alpha)$ equal to

$$\alpha + \frac{1}{2}\#\{(l,j)\in\{1,\dots,n\}\times\{1,\dots,r\} : x_l^j \neq \mathtt{na}\}$$

and

$$\tilde{\beta}^\eta(y_{[t_0,t_n]}, x_{[t_0,t_n]}, \beta) = \beta + \frac{1}{2}\sum_{(l,j):x_l^j\neq \mathtt{na}} \left(x_l^j - y_{t_l}^j\right)^2,$$

where na denotes an unobserved value. This yields a simple way to perform step 3. in Algorithm 1.

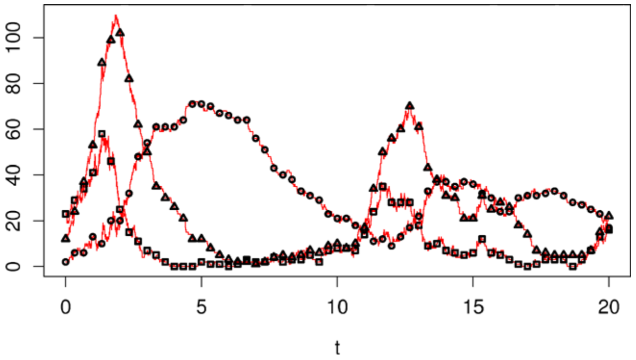

Fig. 1 Sample trajectory of the Oregonator Markov jump process at observation times $0, \frac{1}{3}, \frac{2}{3}, \dots, 20$: y_t^1 (*squares*), y_t^2 (*circles*) and y_t^3 (*triangles*). The *thin lines* indicate the process between the observation times

We now want to estimate the parameters and the Markov jump process given in Fig. 1 with total reaction numbers

$$r_{tot} = (76, 417, 518, 92, 508)^T \tag{17}$$

from observations at the times

$$\mathbb{T} = \left\{0, \frac{1}{3}, \frac{2}{3}, \dots, 20\right\}.$$

We analyze the following situations:

Full observation (F): We observe every species at the time points $\mathbb{T}$. (a): exact, i.e., $\eta=\infty$. (b): with error. We choose $\eta = 1$ and estimate it, too.

Species i latent for $i \in \{1,2,3\}$ (Si): We observe only species Y^j for $j \neq i$ at the time points $\mathbb{T}$. (a): exact, i.e., $\eta = \infty$. (b): with known precision $\eta = 1$.

We use near uniform $\Gamma(1, 0.01) = \mathrm{Exp}(0.01)$ priors on the parameters θ, so that the mode of the posterior should be near to the maximum likelihood estimator. When η is unknown, we use a $\Gamma(2, .2)$ prior. It is also rather flat, but ensures that the precision is greater than 0.

The scenarios S1, S2 and S3 are expected to be very difficult for the MCMC algorithm because of the additional complication of having to mix over the uncertainty of the latent species.

6.1.1 *Specifications of the algorithm*

We specify the proposal distributions and further details in our algorithm as follows. A basis vector matrix is given by

$$V(A) = \begin{pmatrix} 1 & -1 & 0 & 1 & 0\\ 0 & 1 & 1 & 0 & 1 \end{pmatrix}^T$$

and we simulate Z in (6) as follows

$$Z = (2B_s - 1)\cdot(B_c\cdot B_1, (1-B_c)\cdot B_2)^T$$

where $B_s \sim \text{Bin}(1, 1/2)$, $B_c \sim \text{Bin}(1, 1/3)$, $B_1 \sim \text{Bin}(1, 9/10)$ and $B_2 \sim \text{Bin}(6, 1/2)$. We need the larger variance in the second component to ensure good mixing properties since simulation shows that the total numbers of reaction 2, 3 and 5 vary over a much bigger range than reactions 1 and 4. Further, we use the standard set of subintervals described in Subsect. 4.4.

For the initialisation (Algorithm 3), we use $M^{\{l\}}$ and $S^{\{l\}}$ around 100 to 200, slight shrinking and $\eta = 8$ when unknown. We use the standard set of subintervals described in Sect. 4.4.

In the scenarios with exact observation (a), we add a tiny normal noise to the model ($\sigma^2 = 1/\eta = 10^{-4}$). By this trick, we obtain underlying jump processes that match all observations after several iterations of Algorithm 1 as mentioned in Sect. 4.4.

In the scenarios with the latent components (S1, S2 and S3), we replace the standard update from above on intervals $[(t_{l-1} + t_l)/2, (t_l + t_{l+1})/2]$ containing an observation every second MCMC iteration with the special update described in Sect. 4.5 on $[t_{l-1}, t_{l+1}]$, so that the latent component mixes better. To find a suitable distribution to simulate the vector v in (13), we look exemplarily at the scenario S1. First, we need the integer solutions to

$$A_{-1,\cdot}x = 0.$$

With the techniques from the appendix, we find the basis vectors $v_1 = (1, -1, 0, 0, 0)^T$, $v_2 = (0, 0, 0, 1, 0)^T$ and $v_3 = (0, 1, 1, 0, 1)^T$. Because the last one is already in the kernel of A, we can restrict ourselves to v_1 and v_2 for the proposal of v, i.e., we choose $\pm v_1$ or $\pm v_2$ with equal probability 1/4.

For the new reaction number at the beginning on the interval $[t_{n-1}, t_n]$ or at the end on the interval $[t_{n-1}, t_n]$, we want updates which change only one component of y_{t_0} or y_{t_n}, respectively, to get better acceptance. In order to do this for the first component, one can use the same proposal as above and add the resulting vector to the total reaction number to get the new one.

6.1.2 Results

First, we analyze average acceptance rates for the different scenarios separately for updates which do not change the values of the process at the observation times, i.e., updates on intervals $[t_{l-1}, t_l]$ (A), updates on intervals $[(t_{l-1} + t_l)/2, (t_l + t_{l+1})/2]$ (B) with the standard technique and updates on intervals $[t_{l-1}, t_{l+1}]$ with the special update for the scenarios with latent components described in Sect. 4.5 (C). An overview is given in Table 1.

We see that it is a good idea to include the intervals with no changes of the process at observation times since the acceptance rate is much higher on these. When observation is exact (a) in the scenarios with latent components, the need of using the special update (C) is evident, for noisy observations (b), it is not that important.

Table 1 Average acceptance rates in % for the different Oregonator scenarios and process updates on intervals $[t_{l-1}, t_l]$ (A), $[(t_{l-1} + t_l)/2, (t_l + t_{l+1})/2]$ (B) and $[t_{l-1}, t_{l+1}]$ when latent components occur with the technique from Sect. 4.5 (C)

		A	B	C
F	a	17.1	0.3	
	b	17.0	2.6	
S1	a	13.3	0.5	5.9
	b	9.9	3.1	3.3
S2	a	16.9	0.5	7.8
	b	17.1	4.7	7.5
S3	a	16.6	0.5	7.5
	b	17.2	4.6	7.8

Here, we proposed the underlying jump process according to Algorithm 2. The performance of the proposal based on independent Poisson processes (7) was comparable. But between times 3 to 6, where at least one component is very small, Algorithm 2 is clearly better (acceptance rates are up to 2.5 times that big), as expected from the heuristics given in Sect. 4.2.

In Figs. 2 and 3, we show the trace plots for the parameters in scenario Fb and S1a. On the whole, mixing seems satisfactory, although not optimal for some parameters in the scenario S1a with the latent species. Nevertheless, we can produce good estimates as we will see below. In addition, the initialisation process yields starting values which are already very close to the true values. As expected, it reduces the burn-in considerably and thus makes parallelization (if needed) more efficient.

To analyze if our chains are long enough and compare the algorithm in the different scenarios, we calculate effective sample sizes. The effective sample size (ESS) gives the size of an i.i.d. sample with the same variance as the current sample and thus indicates the loss of the efficiency due the use of the Markov chain (see e.g. Robert and Casella 2004, Sects. 12.3.5 and 12.6). Note that the estimation of the ESS is a delicate issue, so the values should be interpreted only as an indication of the order. In Table 2, we estimated the ESS per 10^5 iterations of the MCMC scheme (Algorithm 1) with the function **effectiveSize** from the package **coda** in the language for statistical computing R (see R Development Core Team 2010). This function fits an AR(p) process to the traces of each parameter. The value of the asymptotic variance is then given by a well-known formula.

We observe that the ESS for the parameters in the scenarios S1, S2 and S3 are much smaller than in the case of full observation. Especially for the reaction rates corresponding to standardized transition intensities which depend on the

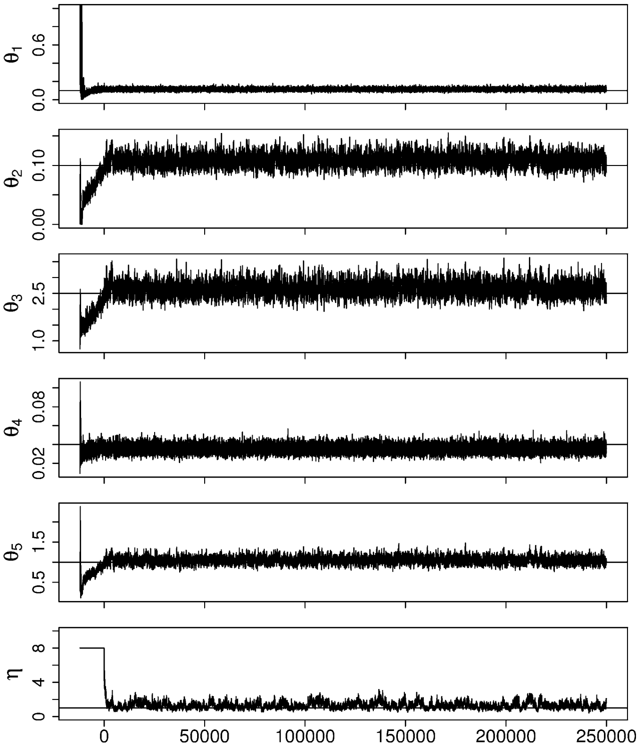

Fig. 2 Traces for the parameters in scenario Fb for the Oregonator example. The origin on the abscissa marks the last iteration of the initialisation (Algorithm 3). True values are indicated with a *horizontal line*

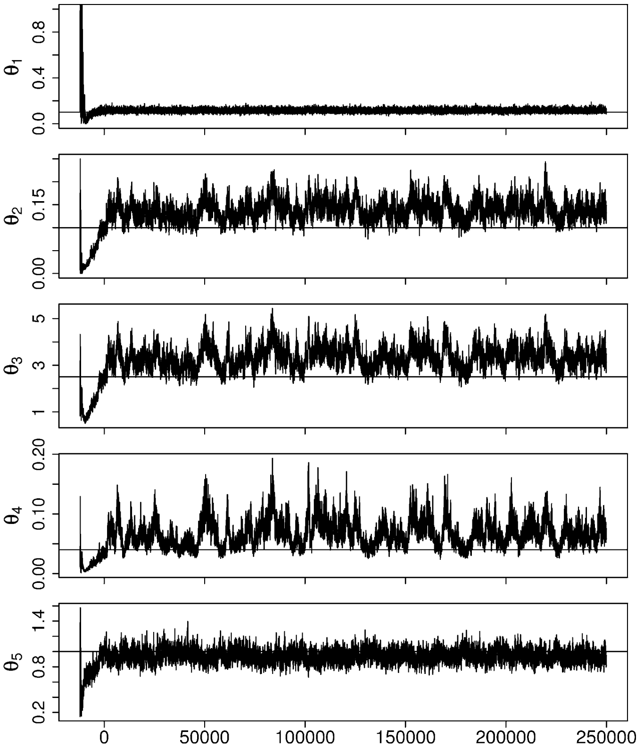

Fig. 3 Traces for the parameters in scenario S1a for the Oregonator example. The origin on the abscissa marks the last iteration of the initialisation (Algorithm 3). True values are indicated with a *horizontal line*

Table 2 ESS per 10^5 iterations (Oregonator)

		θ_1	θ_2	θ_3	θ_4	θ_5	η
F	a	13055	1050	1062	11338	1053	
	b	9861	783	791	7887	792	150
S1	a	4389	192	138	73	498	
	b	3442	84	62	45	268	
S2	a	62	57	598	1344	552	
	b	56	53	507	992	482	
S3	a	1157	354	328	1041	326	
	b	514	369	355	455	169	

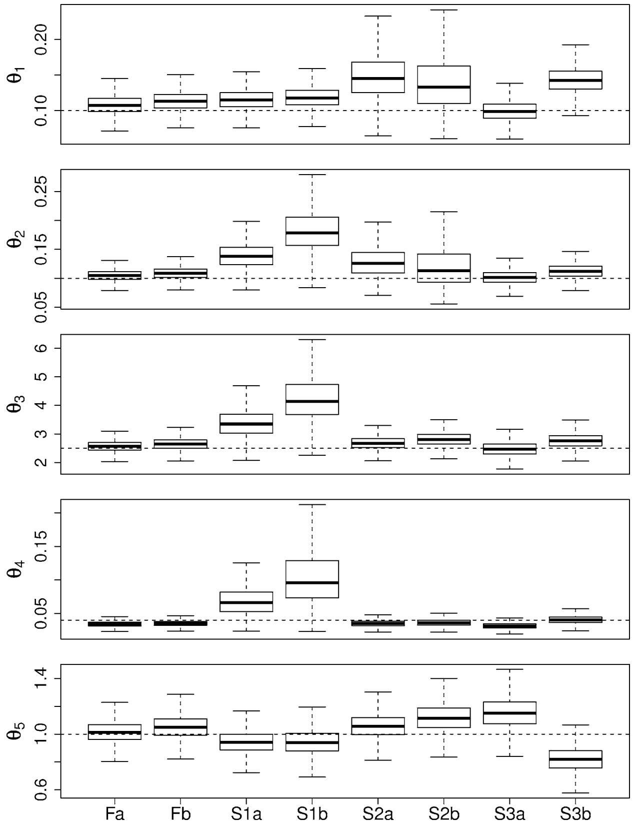

Fig. 4 Box plots of the parameters θ generated with the MCMC Algorithm 1 for the different scenarios (Oregonator model). True values are indicated with a *horizontal dotted line*

latent component. For example in scenario S1, Y^1 is not observed, leading to a loss in terms of mixing for reactions rates θ_2, θ_3 and θ_4. Or in scenario S2, where the ESS of θ_1 and θ_2 is very low. The reason for this is that the algorithm has to mix over the latent components (see Fig. 5). Nevertheless, we are able to produce reasonable estimates, as shown below.

Figure 4 shows box plots of the simulated parameters θ in the different scenarios. All calculations are based on 200,000 iterations of Algorithm 1.

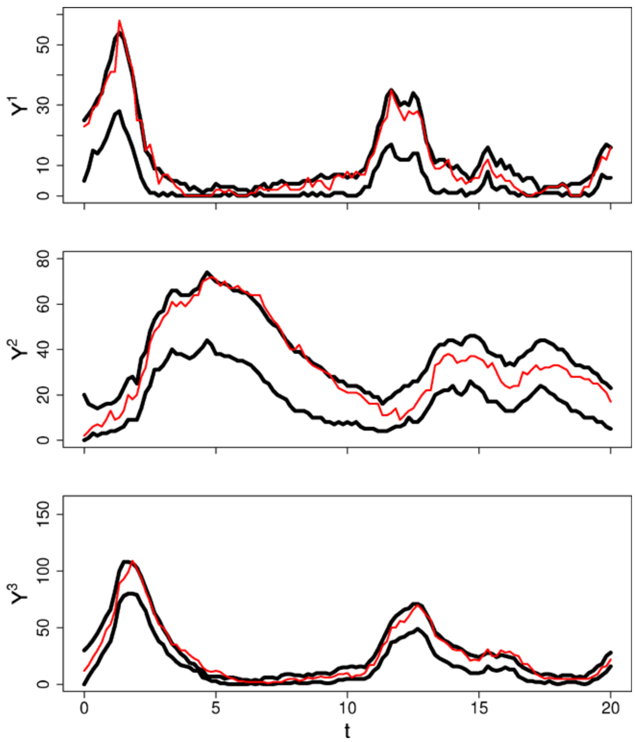

Fig. 5 Pointwise 95% credible bands (indicated by the *thick lines*) of the totally latent components for the Oregonator model in the scenarios S1a (*top*), S2a (*middle*) and S3a (*bottom*), respectively. The true values are shown as *thin line*

In the scenarios with full observation (F), the median is always near the true value. When a component is missing, the true value is mostly in regions where the posterior is high. In the scenario S1, the posteriors of θ_2, θ_3 and θ_4 seem rather spread out. But this are exactly the reaction rates depending directly on the unobserved component.

For the parameter η in scenario Fb, the first quartile, the median and the third quartile are 1.04, 1.27 and 1.53, respectively (true value is 1.00).

Finally, Fig. 5 displays pointwise 95% credible bands of the latent components in the process for the scenarios S1a, S2a and S3a. For comparison, we also indicate the true values of the latent component with a thin line. We can see that they mostly lie within our credible bands which shows that our algorithm can reliably recover the unobserved process component. Note that the values for Y^1 in scenario S1a are rather underestimated, leading to lower values for the standardized reaction intensities depending on y^1. This explains the overestimation of the directly associated parameters θ_2, θ_3 and θ_4 (see Fig. 4).

6.2 Prokaryotic auto-regulation

We look at the simplified model for prokaryotic auto-regulation introduced in Golightly and Wilkinson (2005) and reconsidered in Golightly and Wilkinson (2009). It is described by the following set of 8 chemical reactions.

$$\begin{array}{lll}
R_1: \mathrm{DNA}+\mathrm{P}_2 & \longrightarrow & \mathrm{DNAP}_2\\
R_2: \mathrm{DNAP}_2 & \longrightarrow & \mathrm{DNA}+\mathrm{P}_2\\
R_3: \mathrm{DNA} & \longrightarrow & \mathrm{DNA}+\mathrm{RNA}\\
R_4: \mathrm{RNA} & \longrightarrow & \mathrm{RNA}+\mathrm{P}\\
R_5: 2\mathrm{P} & \longrightarrow & \mathrm{P}_2\\
R_6: \mathrm{P}_2 & \longrightarrow & 2\mathrm{P}\\
R_7: \mathrm{RNA} & \longrightarrow & \emptyset\\
R_8: \mathrm{P} & \longrightarrow & \emptyset
\end{array}$$

In this system, the sum $\mathrm{DNAP}_2 + \mathrm{DNA}$ remains constant, and we assume that this constant K is known and equal to 10 in our simulation. Therefore it is enough to consider the four species $y = (y^1, y^2, y^3, y^4)^T = (\mathrm{RNA}, \mathrm{P}, \mathrm{P}_2, \mathrm{DNA})^T$, where RNA, P, P_2 and DNA are now interpreted as numbers of the corresponding species. According to the mass action law, the standardized transition intensities are

$$h(y) = \left(y^4 y^3, K - y^4, y^4, y^1, y^2(y^2-1)/2, y^3, y^1, y^2\right)^T$$

and the jump matrix is given by

$$A := \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0\\ 0 & 0 & 0 & 1 & -2 & 2 & 0 & -1\\ -1 & 1 & 0 & 0 & 1 & -1 & 0 & 0\\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

As starting distribution, we assume that the number of DNA molecules is uniformly distributed on $\{0, \dots, K\}$ and the other species are initially 0. Following Golightly and Wilkinson (2009), we again use normally distributed measurement errors, see (16). The update for η (step 4. in Algorithm 1), can then be done using Gamma distributions. Also we consider different scenarios in a similar manner to the last example.

Full observation (F): Observation of every species. (a): exact, i.e., $\eta = \infty$. (b): with error. We choose $\eta = 4$ and estimate it, too.

DNA is latent (L): Observation of species RNA, P and P_2. (a): exact, i.e., $\eta = \infty$. (b): with error. We choose $\eta = 4$ and estimate it, too.

The true values of the parameters are

$$\theta = (0.1, 0.7, 0.6, 0.085, 0.05, 0.2, 0.2, 0.015)$$

and we observe the process every 0.5 units of time on the interval $[0, 50]$. The total reaction numbers for the true Markov jump process are

$$r_{tot} = (192, 190, 122, 53, 116, 99, 117, 7)^T.$$

As reported in Golightly and Wilkinson (2005) and Golightly and Wilkinson (2009), ratios of the parameters θ_1/θ_2 and θ_5/θ_6, connected to the reversible reaction pairs R_1, R_2

and R_5, R_6, respectively, are more precise than the individual rates. We found a similar behavior also for θ_3/θ_7 and θ_4/θ_8. This is related to the fact that adding or subtracting an equal number of the corresponding reaction between two consecutive observation times does not change the values of the Markov jump chain at these time points, making it rather difficult to tell how many of these reaction events should be there from discrete observations only. This implies also that there is a strong positive dependence in the posterior between these pairs of parameters.

Therefore we analyse the MCMC algorithm when working with the following reparameterization:

$$\begin{aligned} &\rho_1 = \theta_1 + \theta_2, \qquad \rho_3 = \theta_3 + \theta_7, \qquad \rho_5 = \theta_4 + \theta_8, \\ &\rho_7 = \theta_5 + \theta_6, \qquad \rho_2 = \frac{\theta_1}{\theta_1 + \theta_2}, \qquad \rho_4 = \frac{\theta_3}{\theta_3 + \theta_7}, \\ &\rho_6 = \frac{\theta_4}{\theta_4 + \theta_8}, \qquad \rho_8 = \frac{\theta_5}{\theta_5 + \theta_6}. \end{aligned} \tag{18}$$

For ρ_l $(l = 1, 3, 5, 7)$ we use $\Gamma(a, b)$ priors and for ρ_k $(k = 2, 4, 6, 8)$ Beta(d, e) priors. For updating e.g. (ρ_1, ρ_2), we factor the joint density of (ρ_1, ρ_2) given $y_{[t_0,t_n]}$ as $p(\rho_1|\rho_2)p(\rho_2)$. Then $p(\rho_1|\rho_2)$ is a $\Gamma(a + r^1_{tot} + r^2_{tot}, b + \rho_2 I_1 + (1 - \rho_2)I_2))$ density, and $p(\rho_2) \propto$

$$(b + \rho_2 I_1 + (1 - \rho_2)I_2)^{-(a+N_1+N_2)} \rho_2^{d+N_1-1} (1 - \rho_2)^{e+N_2-1},$$

with $I_i = \sum_{k=1}^{n_{tot}+1} h_i(y_{k-1})\delta_k$ $(i = 1, 2)$. The factor

$$(\beta + \rho_2 I_1 + (1 - \rho_2)I_2)^{-(a+N_1+N_2)}$$

is approximated by piecewise linear upper bounds, so we can simulate from $p(\rho_2)$ using an adaptive accept-reject-method with mixtures of truncated Beta distributions as proposals. After we have generated ρ is this way, we can easily find the corresponding θ using the inverse transform of (18). Thus we have now two ways to update θ. The standard variant (S) and the one via the transformation (18) from above (T). We will see in Sect. 6.2.2 that T has somewhat better mixing properties.

In the scenario F, we use near uniform $\Gamma(1, 0.1)$ priors for θ as before. When working with ρ, we also use $\Gamma(1, 0.1)$ priors for ρ_l $(l = 1, 3, 5, 7)$ and $\mathrm{B}(1, 1)$ priors, i.e., uniform priors on $[0, 1]$, for ρ_k $(k = 2, 4, 6, 8)$, so that the mode of the posterior should be near to the maximum likelihood estimator.

In the scenario L, when working with the above priors, we have the problem that sometimes reaction 1 or 2 are removed from the system, i.e., the corresponding rates are estimated to be 0 or at least very small. Since we assume that these occur, we use for θ_1, θ_2 and ρ_1 $\Gamma(2, 5)$ and for ρ_2 Beta$(1.2, 1.2)$ priors instead. This provides the prior information that corresponding rate parameters are unlikely to be near zero and are around 0.1 to 1.

6.2.1 Specifications of the algorithm

The basis vector matrix is given by

$$V(A) = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}^T$$

and for q_ι^Z we choose $q_\iota^Z(\pm\vec{e}_i) = 0.1$ for $i \in \{1, 2, 3, 4\}$ and $q_\iota^Z(\vec{0}) = 0.2$ (see (6)).

To get the new total reaction number for the update at the beginning, i.e., on the interval $[t_0, t_1]$, we have to respect that $y^1_{t_0} = y^2_{t_0} = y^3_{t_0} = 0$. We therefore only want to change the fourth component of y_{t_0}. So

$$A_{-4,.}(r^{new}_{tot}(y_{[t_0,t_1]}) - r_{tot}(y_{[t_0,t_1]})) = 0.$$

With the techniques from the appendix, we find the same basis vectors as in $V(A)$ plus the vector $v_5 = (0, -1, 0, 2, 1, 0, 0, 0)^T$. So we use (14) where q_ι^R draws not only from $\pm v_5$, but also from some other specific vectors in kernel$(A_{-4,.})\setminus$ kernel(A), e.g. $v_5 + V(A)_1$. For scenario L, where the DNA is latent, we construct a proposal for v in (13) in the same way, since we also want to change the fourth component only.

6.2.2 Results

We analyze each of the four different scenarios Fa, Fb, La and Lb with both methods to generate the rate parameters, i.e., method S (standard) and T (using the transformation). As before, we first look at acceptance rates, see Table 3. Acceptance rates for proposals which update process values at observation times are much lower. In the scenarios where DNA is latent (L), the update according to (13) is not as important as in the previous example, since acceptance rates on intervals $[(t_{l-1} + t_l)/2, (t_l + t_{l+1})/2]$ are similar.

Compared to the Poisson process proposal, Algorithm 2 was slightly better on average, with clear advantage when DNA is on a low level, i.e., 0 or 1.

In Fig. 6, we show the trace plots of the initialisation and 250,000 iterations of Algorithm 1 for the scenario FbS. Stationary behaviour seems to be achieved a few iterations after the initialisation. Once again, Algorithm 3 is of utter utility.

To analyze the gain of the reparameterization, we once again compute the ESS. Results are given in Table 4. For most parameters, the variant with the transformation (T) yields an improvement in terms of ESS and thus has better mixing properties. The ESS usually decreases comparing a scenario with observation errors to the same one without.

Figure 7 shows box plots of the parameters in the different scenarios using the transformation to generate the proposals for θ, based on 100,000 iterations of the MCMC algorithm, which should be enough considering Table 4. We

Table 3 Average acceptance rates in % for the different scenarios and process updates on intervals $[t_{l-1}, t_l]$ (A), $[(t_{l-1}+t_l)/2, (t_l+t_{l+1})/2]$ (B) and $[t_{l-1}, t_{l+1}]$ when latent components occur with the technique from Sect. 4.5 (C)

	A	B	C
FaS	39.4	2.3	
FaT	39.3	2.4	
FbS	39.8	4.4	
FbT	39.3	4.6	
LaS	41.3	2.2	2.7
LaT	40.8	2.4	2.4
LbS	43.6	5.6	2.3
LbT	42.7	6.3	1.7

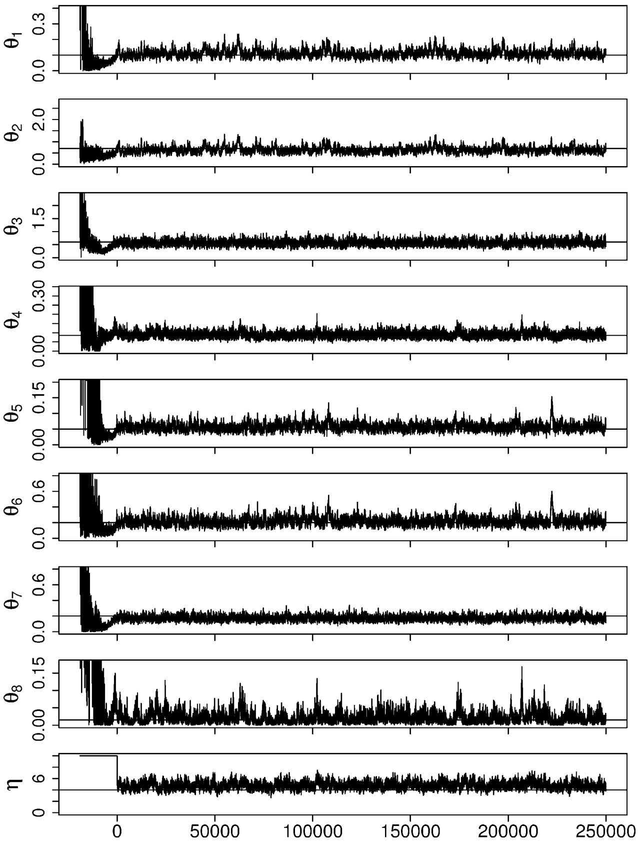

Fig. 6 Traces of (θ, η) for the prokaryotic auto-regulation model in scenario FbS. The origin on the abscissa marks the last iteration of the initialisation (Algorithm 3). True values are indicated with a *horizontal line*

see that in the scenarios where every species is observed (F), posterior medians are near the true values. When the DNA is latent (L), true values are in regions where the posterior density is high.

The medians of the posterior for η in the scenarios FbT and LbT are 4.75 and 4.85, respectively.

Table 4 ESS per 10^5 iterations (prokaryotic auto-regulation)

	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	η
FaS	162	154	1507	21093	406	384	1463	6130	
FaT	189	197	1609	23804	459	428	1718	6671	
FbS	176	160	700	533	303	300	668	251	316
FbT	188	180	731	645	355	343	757	261	319
LaS	118	143	836	351	306	292	1696	259	
LaT	212	217	661	403	505	463	1414	268	
LbS	100	192	271	133	280	267	394	116	158
LbT	205	269	330	122	337	343	421	121	139

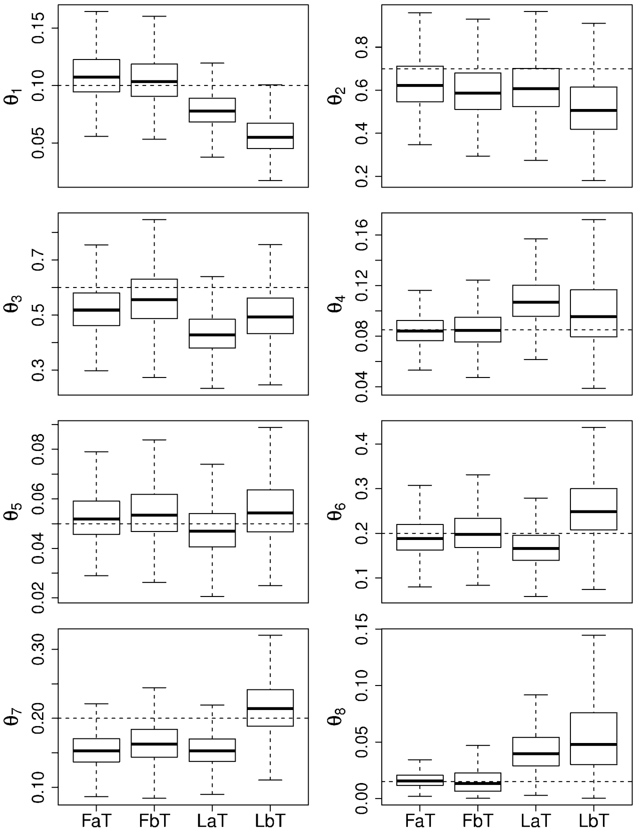

Fig. 7 Box plots of the parameters θ generated with the MCMC Algorithm 1 for the different scenarios. True values are indicated with a *horizontal dotted line*

Finally, we compare the pointwise 95% credible bands of the latent component, that is the number of DNA molecules, in the scenarios LaT and LbT based on 100’000 iterations in Fig. 8. In the case with observation error (LbT), there seems to be a slight overestimation, whereas for exact observation (LaT), the true underlying process component lies nicely within our credible bands.

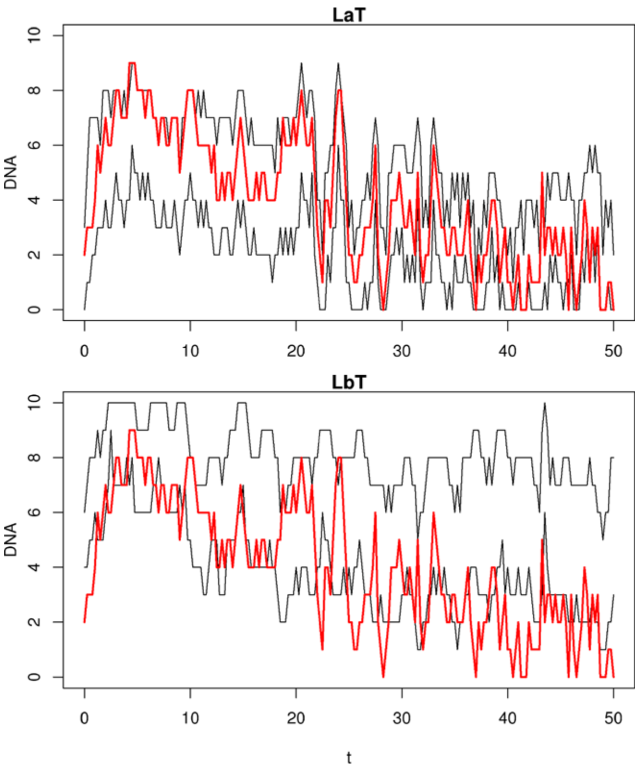

Fig. 8 95% credible bands (indicated by the *thin lines*) of the latent DNA in the scenarios LaT (*top*) and LbT (*below*). True values are shown as *thick line*

7 Conclusions

In this paper, we have presented a technique to infer rate constants and latent process components of Markov jump processes from time series data using fully Bayesian inference and Markov chain Monte Carlo algorithms. We have used a new proposal for the Markov jump process and—exploiting the general state space framework—a filter type initialisation algorithm to render the problem computationally more tractable. Even in very data-poor scenarios in our examples, e.g. one species is completely unobserved, we have been able to estimate parameter values and processes and the true values are contained in the posterior credible bands.

The techniques are generic to a certain extent, but as our examples have shown, they have to be adapted to the situation at hand, which makes their blind application rather difficult. Clearly, the speed of our algorithm scales with the number of jump events, so it is less suitable in situations with many jumps. In such a situation, using the diffusion approximation is recommended. However, we believe that the statement "It seems unlikely that fully Bayesian inferential techniques of practical value can be developed based on the original Markov jump process formulation of stochastic kinetic models, at least given currently available computing hardware" in the introduction of Golightly and Wilkinson (2009) is too pessimistic.

Appendix: Integer solutions of homogeneous linear equations

Let $A \in M_{p\times r}(\mathbb{Z})$ be an integer $p \times r$ matrix. We want to determine the set

$$\mathbb{L} = \{x \in \mathbb{Z}^r : Ax = 0\}. \tag{19}$$

Obviously, it is enough to consider only linear independent rows of A, so we assume $\operatorname{rank}(A) = p \le r$. The case $p = r$ is then trivial, so $p < r$. The main idea is to transform the matrix A into the so called Hermite normal form. We denote with $\lfloor x \rfloor$ the largest integer smaller or equal x. For the following, see Newman (1972), pages 15 ff, or Cohen (1993), pages 66 ff.

Definition 1 (Hermite normal form) $H \in M_{p\times r}(\mathbb{Z})$ with rank s is in Hermite normal form if

1. $\exists i_1,\dots,i_s$ with $1 \le i_1 < \cdots < i_s \le p$ with $H_{i_j,j} \in \mathbb{Z}\setminus\{0\}$ for $1 \le j \le s$.
2. $H_{i,j} = 0$ for $1 \le i \le i_j - 1,\ 1 \le j \le s$.
3. The columns $s+1$ to r are 0.
4. $\lfloor H_{i_j,l}/H_{i_j,j} \rfloor = 0$ for $1 \le l < j \le s$.

Proposition 1 *For every* $A \in M_{p\times r}(\mathbb{Z})$ *exists a unique unimodular matrix* U $(U \in GL_r(\mathbb{Z}) := \{B \in M_{r\times r}(\mathbb{Z}) : \det(B) = \pm 1\})$, *such that* $H = AU$ *is in Hermite normal form.*

There exist many algorithms to calculate H and U, see e.g. Storjohann and Labahn (1996) or Jäger (2001).

The Hermite normal form allows us to determine the set (19). Because A is assumed to have maximal rank, by definition $H = (B, 0)$, where B is an invertible, lower triangular $p \times p$ matrix. For $y = U^{-1}x$ we have the equation $0 = Ax = AUy = (B, 0)y$, so y has zeroes in the first p positions and arbitrary integers in the remaining positions. Hence a basis vector matrix V for (19) is given by $v_i = u_{r+i}$. If necessary, one can reduce the length of the v_i by the Algorithm 2.3 in Ripley (1987).

References

Arkin, A., Ross, J., McAdams, H.H.: Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected escherichia coli cells. Genetics **149**, 1633–1648 (1998)

Bickel, P.J., Doksum, K.A.: Mathematical Statistics; Basic Ideas and Selected Topics. Holden-Day Inc., Oakland (1977)

Boys, R.J., Wilkinson, D.J., Kirkwood, T.B.: Bayesian inference for a discretely observed stochastic kinetic model. Stat. Comput. **18**(2), 125–135 (2008)

Cohen, H.: A Course in Computational Algebraic Number Theory. Springer, Berlin (1993)

Doucet, A., de Freitas, J.F.G., Gordon, N.J.: Sequential Monte Carlo Methods in Practice. Springer, New York (2001)

Durham, G., Gallant, R.: Numerical techniques for maximum likelihood estimation of continuous time diffusion processes. J. Bus. Econ. Stat. **20**, 279–316 (2002)

Fearnhead, P., Meligkotsidou, L.: Exact filtering for partially observed continuous time models. J. R. Stat. Soc. B **66**(3), 771–789 (2004)

Gibson, M.A., Bruck, J.: Efficient exact stochastic simulation of chemical systems with many species and many channels. J. Phys. Chem. A **104**(9), 1876–1889 (2000)

Gilks, W.R., Berzuini, C.: Following a moving target-Monte Carlo inference for dynamic Bayesian models. J. R. Stat. Soc. B **63**(1), 127–146 (2001)

Gilks, W.R., Richardson, S., Spiegelhalter, D.J.: Markov Chain Monte Carlo in Practice. Chapman and Hall, London (1996)

Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem. **81**, 2340–2360 (1977)

Golightly, A., Wilkinson, D.: Bayesian inference for nonlinear multivariate diffusion models observed with error. Comput. Stat. Data Anal. **52**(3), 1674–1693 (2008)

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

Golightly, A., Wilkinson, D.J.: Bayesian sequential inference for stochastic kinetic biochemical network models. J. Comput. Biol. **13**(3), 838–851 (2006)

Golightly, A., Wilkinson, D.J.: Markov chain Monte Carlo algorithms for SDE parameter estimation. In: Learning and Inference for Computational Systems Biology. MIT Press, Cambridge (2009)

Hobolth, A., Stone, EA: Simulation from endpoint-conditioned, continuous-time Markov chains on a finite state space, with applications to molecular evolution. Ann. Appl. Stat. **3**(3), 1204–1231 (2009)

Jäger, G.: Algorithmen zur Berechnung der Smith-Normalform und deren Implementation auf Parallelrechnern. PhD thesis, Universität Essen, Fachbereich 6 (Mathematik und Informatik) (2001)

Künsch, H.R.: Complex Stochastic Systems. Chapman & Hall/CRC, London (2000), Chap. 3

McAdams, H., Arkin, A.: Stochastic mechanisms in gene expression. Proc. Natl. Acad. Sci. USA **94**, 814–819 (1997)

Newman, M.: Integral Matrices. Academic Press, New York (1972)

Ogata, Y.: On Lewis' simulation method for point processes. IEEE Trans. Inf. Theory **27**(1), 23–31 (1981)

R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna (2010). http://www.R-project.org

Ripley, B.D.: Stochastic Simulation. Wiley, New York (1987)

Robert, C.P., Casella, G.: Monte Carlo Statistical Methods, 2nd edn. Springer Texts in Statistics. Springer, New York (2004)

Storjohann, A., Labahn, G.: Asymptotically fast computation of Hermite normal forms of integer matrices. In: Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation, ISSAC '96, pp. 259–266. ACM, New York (1996)

Wilkinson, D.J.: Stochastic Modelling for Systems Biology. Chapman & Hall, London (2006)