Bayesian Prediction for Stochastic Process Models in Reliability

by

Simone Hermann

Dissertation

zur Erlangung des akademischen Grades eines Doktors
der Naturwissenschaften (rer. nat.) der Fakultät
Statistik der Technischen Universität Dortmund

technische universität
dortmund

Vorgelegt: 17. Oktober 2016

Tag der mündlichen Prüfung: 21. Dezember 2016

Erstgutachter: Prof. Dr. Christine Müller

Zweitgutachter: Prof. Dr. Katja Ickstadt

Kommissionsvorsitz: Prof. Dr. Jörg Rahnenführer

Contents

1. Introduction

The following work has been developed in the project B5 “Statistical methods for damage processes under cyclic load” of the Collaborative Research Center “Statistical modeling of nonlinear dynamic processes” (SFB 823) of the German Research Foundation (DFG). The aim of the project is to consider material fatigue from a statistical point of view, whereby modeling and forecasts are desired. Since Bayesian analysis is a strong tool for prediction, this thesis provides full Bayesian inference, estimation as well as prediction, for models that can be used to predict material fatigue in various materials. In this thesis, we will be mainly concerned with two different materials.

Firstly, Virkler et al. (1979) made experiments with aluminum alloy, where 68 specimen were set under cyclic load to investigate crack length propagation. Since aluminum alloy is a homogeneous material, the crack length propagation process appears to be continuous without any irregularities and jumps.

Secondly, Reinhard Maurer, Guido Heeke and Jens Heinrich conducted fatigue experiments on prestressed concrete under a low cyclic stress level within the SFB 823 to investigate the long life fatigue strength. Since prestressed concrete is a combination of concrete and tension wires, which break over time, this crack width process exhibits irregular jumps.

A model often used for crack growth in the engineering literature, is the deterministic process defined by the Paris-Erdogan equation

$$\frac{da}{dN} = C(\Delta K)^m, \tag{1.1}$$

where a denotes the crack length and N the number of cycles corresponding to the crack length a. C and m are material dependent constants. ΔK denotes the stress intensity factor range that depends on the square root of a. That leads to an autoregressive process whose differences increase with the growth of the process itself. For further information see, for example, Sobczyk and Spencer (1992).

Although all the methods described in the following can be applied in general, for specific modeling in application on the presented data sets, the process (1.1) will be taken as basis.

Firstly, for modeling of continuous crack propagation processes, a general diffusion process will be presented based on a stochastic differential equation (SDE). For generalization, we refrain from solving the SDE to an explicit solution and approximate the SDE with the widely used Euler-Maruyama approximation, see, for example, Fuchs (2013).

Secondly, for the jump process of the non-continuous crack propagation, a non-homogeneous Poisson process (NHPP) will be used.

Thirdly, both components are combined in a jump diffusion process for modeling of the crack propagation process. This process will be defined by a SDE as well.

For the respective models, in most cases, Bayesian estimation is straightforward or presented in existing literature. We will encounter two models with an arising filtering problem and present a solution which makes it possible to estimate the model parameters. Firstly, a hidden diffusion model will be considered and a particle Gibbs sampler for the filtering of the unobserved diffusion will be applied. Secondly, the Poisson process within the jump diffusion will be assumed to be unobserved, where we propose a simple filtering step within the Gibbs sampler. Note that for the underlying prestressed concrete data set, the counting process is observed.

Little research has been done in the field of prediction for stochastic processes without existing explicit solution of the SDE, both for general diffusion and for jump diffusion. In the case of an explicit solution, a distribution of the process in each time point is available and can be taken for a simple Bayesian prediction. This is explained, for example, for the Ornstein-Uhlenbeck and the Cox-Ingersoll-Ross process in a mixed effects model in Dion, Hermann and Samson (2016) and implemented in the R package mixedsde, see Dion, Samson and Hermann (2016).

If the process is approximated by the Euler-Maruyama scheme, this prediction can become inaccurate for large time differences. A common solution in the Bayesian community is to sample trajectories with samples from the posterior distribution and take these for prediction, namely the pointwise mean for a point prediction or quantiles for prediction intervals, respectively. The analogous frequentist solution is sampling of trajectories with the point estimates of the parameters. Both procedures will be compared to the methods presented in this thesis.

In our project within the SFB 823, prediction for the far future is desired, and, in the following, we present two Bayesian prediction methods for stochastic processes based on the Euler approximation, one for trajectories of the process and one for a predictive distribution for the process in one time point.

The thesis is structured as follows. In the next section, the model is presented, used sampling algorithms are explained, a short sensitivity analysis with motivation of prior distributions is performed and, as main part of the thesis, Bayesian prediction for stochastic processes based on the transition density, which can be the one resulting from the Euler approximation, is presented. Afterwards, the implication of different time distances for the Euler approximation is investigated.

Sections 3 and 4, joint in Chapter I, consider the general diffusion process. In Section 4, the diffusion is hidden in a regression model, where the estimation is complicated through a filtering problem.

In Chapter II, firstly, Section 5 introduces the NHPP and secondly, Section 6 joins diffusion and counting process into a jump diffusion. Thirdly, Section 7 presents a non-linear regression model including the NHPP, which is a result of the interdisciplinary work in the project with engineers, see Heeke, Hermann et al. (2015).

Section 8 describes the package BaPreStoPro (**Ba**yesian **Pre**diction of **Sto**chastic **Pro**cesses), developed as part of this thesis. All the methods described in this thesis are implemented in R, see R Core Team (2015), and joint in the package BaPreStoPro, see Hermann (2016a).

2. Foundations for Modeling, Sampling and Prediction

The goal of statistics is to reconstruct real phenomena with a specific uncertainty in a model to allow some kind of inference, for example, for the future. A wide class of models is the one of Markov processes, which have the property that the current state only depends on the previous state. One example would be the crack length in a specific material, which could be observed every day or every minute or, as in our cases, over a certain number of load cycles. In the first case one would assume a discrete-time Markov process. For measurements every minute or over load cycles, a continuous-time one would be also suitable. In the case of non-equidistant time points, a continuous-time choice would be suitable as well. The examples in this thesis are all concerned with crack growth in different materials, for example aluminum alloy, a homogeneous material, or prestressed concrete, a non-homogeneous material. All have in common that the current crack size only depends on the crack size before and on the increment, which by its nature is a Markov process. To be flexible with the observation time distances, we will assume a continuous-time process. In the case of homogeneous material, we can assume a continuous process. This will be done by a diffusion driven by a Wiener process. In the case of a non-homogeneous material, we observe irregular jumps, which make a non-continuous process suitable. Here, we will assume a jump diffusion process. To join all the presented models in one representation, we consider the model described in the following section.

2.1. Main Model

In this thesis, we consider the one-dimensional stochastic process $\{Y_t, t \in [0,T]\}$ on the time interval $[0,T]$ defined by the stochastic differential equation (SDE) given by

$$dY_t = b(\phi, t, Y_t)\, dt + s(\gamma, t, Y_t)\, dW_t + h(\eta, t, Y_t)\, dN_t, Y_0 = y_0(\phi), \tag{2.1}$$

where $\{W_t,\ t \in [0,T]\}$ denotes a Brownian motion on $[0,T]$, $\{N_t,\ t \in [0,T]\}$ denotes a non-homogeneous Poisson process with cumulative intensity function $\mathbb{E}[N_t] = \int_0^t \lambda_\xi(u)\, du = \Lambda_\xi(t)$ which has to be bounded on $[0,T]$. Both, Brownian motion and the Poisson process, have to be stochastically independent. y_0 will be assumed to be constant in some cases, but for the hierarchical and the hidden diffusion model, this will be a function of the location parameter ϕ. We could define the process also on the time interval $[0,\infty)$. But, our observation time interval is naturally bounded, since we observe material fatigue until failure. Strictly speaking, T in this context is a random variable, but we neglect this fact and assume T large enough that observation time points $0 \le t_0 < t_1 < ... < t_n \le T$ are located in the interval $[0,T]$.

Stochastic differential equations as in (2.1) are well-investigated, see, e.g., Øksendal and Sulem (2005); Shimizu and Yoshida (2006); Klebaner (2005); Protter (2005); Fuchs (2013). Throughout this thesis, we consider a parametric estimation approach, this means, functions b, s and h are assumed to be known. Under the Lipschitz conditions

$$\begin{aligned}
(b(\phi, t, x) - b(\phi, t, y))^2 &\le C_1 (x-y)^2,\\
(s(\gamma, t, x) - s(\gamma, t, y))^2 &\le C_2 (x-y)^2,\\
(h(\eta, t, x) - h(\eta, t, y))^2 &\le C_3 (x-y)^2
\end{aligned}$$

and the linear growth conditions

$$b(\phi,t,y)^2 \leq D_1(1+y^2),\ s(\gamma,t,y)^2 \leq D_2(1+y^2),\ h(\eta,t,y)^2 \leq D_3(1+y^2)$$

for all $t \in [0,T], x, y \in \mathbb{R}$, C_i and $D_i, i = 1,2,3$, positive constants, SDE (2.1) has a unique strong solution, see Ikeda and Watanabe (1989) for further details.

The resulting process is closely related to the class of Lévy processes, see, e.g., Protter (2005). For general functions b, s, h and Λ_ξ, the process defined by the SDE (2.1) does not have stationary increments, in contrary to a classical Lévy process. According to the definition in Cont and Tankov (2004) and Kluge (2005), our process fits in the class of time-inhomogeneous Lévy Processes.

In this thesis, the parameters in equation (2.1) have the following dimensions: $\phi \in \mathbb{R}^p$ will be assumed to be a vector, $\gamma \in (0,\infty)$ and $\eta \in \mathbb{R}$ will be assumed to be scalars, whereby extensions to a multi-dimensional vector would be easy to implement. Function s depends on γ because in most cases it will be $s(\gamma,t,y) = \gamma$ or $s(\gamma,t,y) = \gamma y$, but for inference γ^2, i.e. the parameter of the resulting variance, will be estimated.

During this thesis, we consider different specific cases of the process, for example, the continuous diffusion process, i.e. $h(\eta,t,y) = 0$, or the Poisson process itself, i.e. $h(\eta,t,y) = 1$, $b(\phi,t,y) = s(\gamma,t,y) = 0$, $y_0 = 0$.

If the SDE (2.1) has an explicit solution, one has a transition density that goes in for the estimation. But in the case that the process has no explicit representation, the SDE can be approximated in time points $0 \leq t_0 < t_1 < ... < t_n \leq T$. A widely used approximation scheme is the Euler-Maruyama, or short Euler, approximation and is given by

$$\begin{aligned}
&Y_0 = y_0(\phi), \qquad\qquad (2.2)\\
&Y_i = Y_{i-1} + b(\phi,t_{i-1},Y_{i-1})\,\Delta_i + s(\gamma,t_{i-1},Y_{i-1})\sqrt{\Delta_i}\,\zeta_i + h(\eta,t_{i-1},Y_{i-1})\Delta N_i,\\
&\zeta_i \sim \mathcal{N}(0,1) \text{ i.i.d.}, \quad \Delta N_i \sim \text{Pois}\left(\Lambda_\xi(t_i) - \Lambda_\xi(t_{i-1})\right),\\
&\Delta_i = t_i - t_{i-1},\ i = 1,...,n,
\end{aligned}$$

with i.i.d. meaning independent and identically distributed. For jump diffusions, see Platen and Bruti-Liberati (2010) or for ordinary diffusions, i.e. $h(\eta,t,y) = 0$, Fuchs (2013). Conditional on the jump term, the approximated variables have the normal transition density

$$p(Y_i|Y_{i-1},\phi,\gamma,\eta,\Delta N_i) = \\ \frac{1}{\sqrt{2\pi\Delta_i}s(\gamma,t_{i-1},Y_{i-1})}\exp\left(-\frac{(Y_i - Y_{i-1} - b(\phi,t_{i-1},Y_{i-1})\,\Delta_i - h(\eta,t_{i-1},Y_{i-1})\Delta N_i)^2}{2s^2(\gamma,t_{i-1},Y_{i-1})\Delta_i}\right).$$

In most cases, the true underlying transition density cannot be assumed to be normal. A good overview of possibly resulting problems of biased estimations can be seen in Sørensen (2004) or Beskos et al. (2006) for the case $h(\eta,t,y) = 0$.

The main scope of this thesis is prediction. For the different specific cases of the process in (2.1), prediction procedures will be presented. In the case where the Euler approximation is used, the prediction is also based on the same discretized variables as used in the estimation which circumvents the problem of possibly biased estimations.

In the following, we use notation $\theta = (\phi, \gamma^2, \eta, \xi)$ for the parameter vector, or parts of it in the respective model, and $Y_{(n)} = (Y_0, Y_1, ..., Y_n)$ for the vector of Euler approximated variables.

2.2. Sampling Algorithms

In Bayesian estimation, the posterior distributions often are not explicitly available. Therefore, sampling algorithms may to be used for approximation. In the following, the algorithms used in this thesis, are described. The first one is the most popular in the class of Markov Chain Monte Carlo (MCMC) algorithms.

2.2.1. Metropolis Hastings (MH) Algorithm

We want to sample from the posterior distribution

$$p(\theta|Y_{(n)}) \propto p(Y_{(n)}|\theta) \cdot p(\theta).$$

At first, choose a proper proposal density $q(\theta^*|\theta)$. A minimal necessary condition is that

$$\bigcup_{\theta \in \text{supp } p(\cdot|Y_{(n)})} \text{supp } q(\cdot|\theta) \supset \text{supp } p(\cdot|Y_{(n)})$$

with supp denoting the support of a function. In words, each point of the posterior's support has to be available by the Markov chain based on that proposal density. The formal condition would also be true for the density of the Dirac measure at θ. Therefore, this is only the minimal necessary condition, the proposal density has to be chosen wisely for a good approximation of the posterior distribution.

At second, choose a starting value θ_0^* with $p(\theta_0^*|Y_{(n)}) > 0$. For $k = 1, ..., K$ draw $\theta^* \sim q(\cdot|\theta_{k-1}^*)$ and set $\theta_k^* = \theta^*$ with acceptance probability

$$\rho(\theta^*, \theta_{k-1}^*) = \min\left\{\frac{p(Y_{(n)}|\theta^*) \cdot p(\theta^*)}{p(Y_{(n)}|\theta_{k-1}^*) \cdot p(\theta_{k-1}^*)} \cdot \frac{q(\theta_{k-1}^*|\theta^*)}{q(\theta^*|\theta_{k-1}^*)},\ 1\right\}$$

and $\theta_k^* = \theta_{k-1}^*$ with probability $1-\rho(\theta^*, \theta_{k-1}^*)$. Then, the stationary distribution of the Markov chain $\{\theta_k^*,\ k = 1, ..., K\}$ is equal to the posterior $p(\theta|Y_{(n)})$, see Robert and Casella (2004), p. 272.

In the special case of a symmetric proposal density, i.e. $q(\theta^*|\theta_{k-1}^*) = q(\theta_{k-1}^*|\theta^*)$ the acceptance probability reduces to

$$\rho(\theta^*, \theta_{k-1}^*) = \min\left\{\frac{p(Y_{(n)}|\theta^*) \cdot p(\theta^*)}{p(Y_{(n)}|\theta_{k-1}^*) \cdot p(\theta_{k-1}^*)},\ 1\right\}.$$

One example is the density of the $\mathcal{N}(\theta^*_{k-1}, sd^2)$ distribution with mean equal to the last iteration step θ^*_{k-1} and a fixed standard deviation sd. This is also known as symmetric random walk, see Robert and Casella (2004), p. 206.

A crucial step of the algorithm is the choice of the proposal density. Theoretically, the minimal necessary condition given above suffices. But in practice, a good approximation of the posterior density is of interest. If the acceptance probability $\rho(\theta^*, \theta^*_{k-1})$ is large, many samples are accepted, but the new one moves only little from the old iteration step and the support of the distribution will be explored slowly. On the other hand, if the acceptance probability is small, only few candidates are accepted, which is also not suitable for a continuous distribution. Rosenthal (2011) calculated an optimal acceptance rate of the MH algorithm, which is 0.234. To reach this, an adaptive algorithm might be a solution, which will be presented below.

The choice of the starting point is also an important step of the algorithm. If it is not well chosen, the chain needs many iterations to reach the stationary distribution and all these samples are skipped, this is called burn-in phase. We will make an example to see this more clear.

Example 1
We consider process (2.1) *with* $h(\eta, t, y) = 0$, $s(\gamma, t, y) = \gamma y$ *and* $b(\phi, t, y) = \phi y$. *It is approximated by*

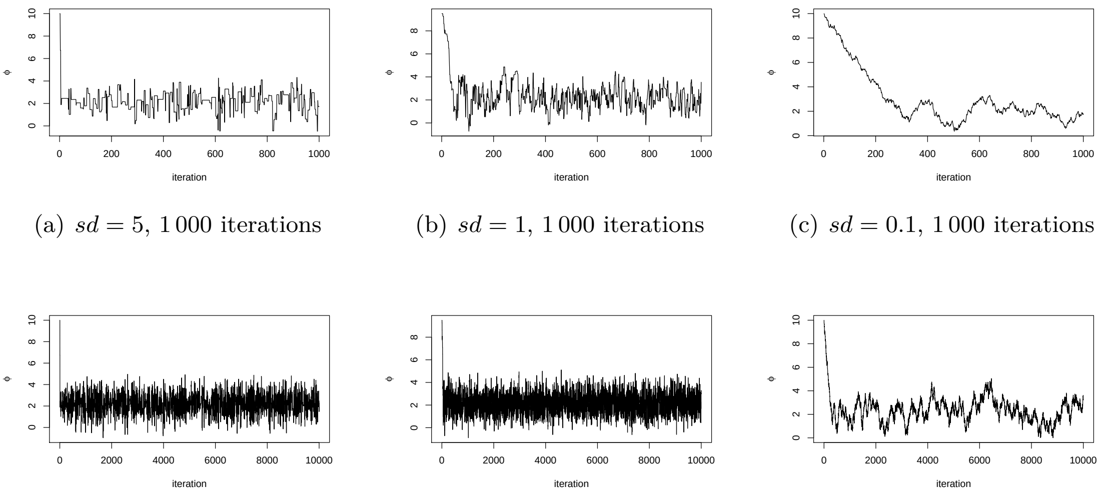
$$Y_i = Y_{i-1}(1 + \phi\,\Delta_i) + \gamma Y_{i-1}\sqrt{\Delta_i}\,\zeta_i,\ \ i = 1, ..., n,$$

with $Y_0 = y_0$ *and* $\zeta_i \sim \mathcal{N}(0, 1)$. *We consider the process with* $y_0 = 1, \phi = 2,\ \gamma = 1$ *in* $t_0 = 0, t_1 = 0.02, ..., t_{50} = 1$.

With prior distribution $\phi \sim \mathcal{N}(2, 4)$ and fixed $\gamma^2 = 1$ for a simple MH algorithm for ϕ, the posterior is proportional to

$$p(\phi|Y_{(n)}, \gamma^2) \propto \prod_{i=1}^{n} p(Y_i|Y_{i-1}, \phi, \gamma^2) \cdot p(\phi) \tag{2.3}$$
$$= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\gamma^2 Y_{i-1}^2 \Delta_i}} \exp\left(-\frac{1}{2\gamma^2 Y_{i-1}^2 \Delta_i}\left\{Y_i - Y_{i-1}(1 + \phi\Delta_i)\right\}^2\right) \frac{1}{\sqrt{2\pi \cdot 4}} \exp\left(-\frac{1}{2\cdot 4}(\phi - 2)^2\right)$$
$$= \prod_{i=1}^{n} \frac{1}{0.2\sqrt{\pi Y_{i-1}^2}} \exp\left(-\frac{1}{0.04 Y_{i-1}^2}\left\{Y_i - Y_{i-1}(1 + 0.02\phi)\right\}^2\right) \frac{1}{2\sqrt{2\pi}} \exp\left(-\frac{1}{8}(\phi - 2)^2\right).$$

We start the algorithm with $\phi^*_0 = 10$ and draw the new candidate from the normal distribution with mean equal to the last sample and compare proposal standard deviations $sd \in \{5, 1, 0.1\}$. In Figure 1, we see the resulting Markov chains. The acceptance rate with the large standard deviation is 0.23 which is close to the optimal one calculated in Rosenthal (2011). The first 1 000 chain iterations can be seen in Figure 1(a) and there are constant phases in which no new candidate is accepted. In 1(b), the first 1 000 iterations with standard deviation equal to 1 are displayed. The acceptance rate is 0.67 which is larger than the optimal one. But with a close look on the chain, this result would be also good for the approximation of the posterior distribution. The stationary distribution is reached after 100 samples and the distribution support is well filled. With a look on Figures 1(d) and 1(e) displaying 10 000 iterations of the respective chain, the difference is very small. Another picture appears in Figures 1(c) and 1(f) which show the chains for the small proposal standard deviation 0.1. The acceptance rate is 0.96 which is large. To get an equally good approximation of the posterior as for the two other chains, much more iterations would have to be drawn and a high thinning rate would be necessary.



(a) $sd = 5$, 1 000 iterations

(b) $sd = 1$, 1 000 iterations

(c) $sd = 0.1$, 1 000 iterations

(d) $sd = 5$, 10 000 iterations

(e) $sd = 1$, 10 000 iterations

(f) $sd = 0.1$, 10 000 iterations

Figure 1: Markov chains, based on different proposal variances, with stationary distribution (2.3) and starting value $\phi_0^* = 10$.

In the following, $\theta_1^*, ..., \theta_K^*$ will always be understood as the resulting samples from the posterior, already reduced by the burn-in period and the thinning rate.

In the case of a high-dimensional vector θ, it might be difficult to find a proper proposal density for the MH algorithm to update all components simultaneously. In this case, the Gibbs sampler can be more suitable.

2.2.2. Gibbs Sampler

For a high dimensional parameter vector $\theta = (\theta_1, ..., \theta_d)$, where each $\theta_j, j = 1, .., d$, can be also a vector itself, the idea is to sample iteratively from the full conditional posterior distributions of the components $p(\theta_j | Y_{(n)}, \theta_1, ..., \theta_{j-1}, \theta_{j+1}, ..., \theta_d),\ j = 1, ..., d$.

Choose starting values $\theta^*_{2,0}, ..., \theta^*_{d,0}$ and for $k = 1, ..., K$ draw

$$\begin{aligned}
\theta^*_{1,k} &\sim p(\theta_1|Y_{(n)}, \theta^*_{2,k-1}, ..., \theta^*_{d,k-1}),\\
\theta^*_{2,k} &\sim p(\theta_2|Y_{(n)}, \theta^*_{1,k}, \theta^*_{3,k-1}, ..., \theta^*_{d,k-1}),\\
&\vdots\\
\theta^*_{d-1,k} &\sim p(\theta_{d-1}|Y_{(n)}, \theta^*_{1,k}, ..., \theta^*_{d-2,k}, \theta^*_{d,k-1}),\\
\theta^*_{d,k} &\sim p(\theta_d|Y_{(n)}, \theta^*_{1,k}, ..., \theta^*_{d-1,k}).
\end{aligned}$$

The resulting Markov chain $\{(\theta^*_{1,k}, ..., \theta^*_{d,k}),\ k = 1, ..., K\} = \{\theta^*_k,\ k = 1, ..., K\}$ has stationary distribution $p(\theta|Y_{(n)})$, see Robert and Casella (2004), p. 372, or, for comparison, Carlin and Louis (2009).

2.2.3. Metropolis within Gibbs Algorithm

In the case of not explicitly available full conditional posteriors, one step of the Gibbs sampler can be conducted by a MH algorithm. The original work of Metropolis et al. (1953) introduced what we now call Metropolis within Gibbs algorithm. Here, we restrict to the case of only one component $\theta_j, j \in \{1, ..., d\}$, to be sampled by a MH step. Of course, this can be done for several components. In addition, for notation simplicity, we assume the jth component to be independent from the others in their prior distribution. This means $p(\theta) = p(\theta_j)p(\theta_1, ..., \theta_{j-1}, \theta_{j+1}, ..., \theta_d)$. We choose a proper proposal density $q(\cdot|\theta^*_{1,k}, ..., \theta^*_{j-1,k}, \theta^*_{j,k-1}, ..., \theta^*_{d,k-1})$ for θ_j similar to the MH algorithm and starting values $\theta^*_{2,0}, ..., \theta^*_{d,0}$, if $j = 1$ also $\theta^*_{1,0}$. For $k = 1, ..., K$ draw

$$\begin{aligned}
\theta^*_{1,k} &\sim p(\theta_1|Y_{(n)}, \theta^*_{2,k-1}, ..., \theta^*_{d,k-1}),\\
&\vdots\\
\theta^*_{j-1,k} &\sim p(\theta_{j-1}|Y_{(n)}, \theta^*_{1,k}, ..., \theta^*_{j-2,k}, \theta^*_{j,k-1}, ..., \theta^*_{d,k-1}),
\end{aligned}$$

$\theta^*_j \sim q(\theta_j|\theta^*_{1,k}, ..., \theta^*_{j-1,k}, \theta^*_{j,k-1}, ..., \theta^*_{d,k-1})$ and accept $\theta^*_{j,k} = \theta^*_j$ with probability

$$\begin{aligned}
\rho(\theta^*_j, \theta^*_{j,k-1}) = \min\Bigg\{1,\ &\frac{p(Y_{(n)}|\theta^*_{1,k}, ..., \theta^*_{j-1,k}, \theta^*_j, \theta^*_{j+1,k-1}, ..., \theta^*_{d,k-1}) \cdot p(\theta^*_j)}{p(Y_{(n)}|\theta^*_{1,k}, ..., \theta^*_{j-1,k}, \theta^*_{j,k-1}, ..., \theta^*_{d,k-1}) \cdot p(\theta^*_{j,k-1})}\\
&\cdot \frac{q(\theta^*_{j,k-1}|\theta^*_{1,k}, ..., \theta^*_{j-1,k}, \theta^*_j, \theta^*_{j+1,k-1}, ..., \theta^*_{d,k-1})}{q(\theta^*_j|\theta^*_{1,k}, ..., \theta^*_{j-1,k}, \theta^*_{j,k-1}, ..., \theta^*_{d,k-1})}\Bigg\}
\end{aligned}$$

and $\theta^*_{j,k} = \theta^*_{j,k-1}$ with probability $1 - \rho(\theta^*_j, \theta^*_{j,k-1})$,

$$\begin{aligned}
\theta^*_{j+1,k} &\sim p(\theta_{j+1}|Y_{(n)}, \theta^*_{1,k}, ..., \theta^*_{j,k}, \theta^*_{j+2,k-1}, ..., \theta^*_{d,k-1}),\\
&\vdots\\
\theta^*_{d,k} &\sim p(\theta_d|Y_{(n)}, \theta^*_{1,k}, ..., \theta^*_{d-1,k}).
\end{aligned}$$

See for further details Robert and Casella (2004), p. 392.

2.2.4. Adaptive MCMC

As seen in the description of the MH algorithm in Section 2.2.1, the quality of the algorithm strongly depends on the proposal density. The remaining question is how to choose a suitable proposal variance. This variance controls the approximation quality of the distribution of interest. We have seen the effect in Figure 1. This problem is tackled in Rosenthal (2011) with an adaptive MH step using the proposal distribution $\mathcal{N}(\theta^*_{k-1}, e^{2l})$ with l as the logarithm of the standard deviation and θ^*_{k-1} the previous iteration state. This parameter is chosen so that the acceptance rate is close to the optimal, which is 0.234 for a classical MH algorithm and 0.44 for the Metropolis within Gibbs sampler, see Rosenthal (2011). The idea is to divide the chain into batches, for example of 50 iterations. After every batch, the current acceptance rate is calculated. If it is outside a specific range it is adapted, whereby we will choose [0.3,0.6] in the following calculations. Rosenthal (2011) propose to add, resp. subtract, an adoption amount $\delta(N) = \min(0.1, N^{-1/2})$ to, resp. from, l after the Nth batch.

The presented methods consider sampling from the posterior distribution of θ. The MH algorithm and the Gibbs sampler are very suitable in the case of easy and fast computable densities. We will present a predictive distribution which also will not be explicitly available and its calculation is computationally costly. In the case of expensive computation of the interesting density, the following algorithms are more suitable. Because we sample from the predictive density, we switch the notation from θ to y.

2.2.5. Rejection Sampling

One possibility to draw random samples from a distribution with density $p(y)$ without Markov chains is to choose a so called envelope (or proposal) density $g(y)$, where it is easy to draw from, and accept or reject the samples. This envelope density has to hold the condition that there is some constant A so that $p(y) \leq Ag(y)$ for all $y \in \operatorname{supp} p(\cdot)$.

To draw one sample of a distribution with density $p(y)$, conduct the following steps.

(i) Generate a realization u from the uniform distribution $\mathcal{U}(0,1)$ and independently one sample y^* from $g(y)$.

(ii) If $u \leq \frac{p(y^*)}{Ag(y^*)}$, accept y^* as a sample from distribution with density p otherwise, reject the candidate.

One challenge of this algorithm is the choice of density $g(y)$. One possibility is to choose a candidate area $[y_l, y_u]$, which should include the support of $p(y)$, and set $g(y) = \frac{1}{y_u - y_l}\mathbb{1}(y)$ equal to the density of the uniform distribution on the candidate area. With $A = (y_u - y_l) \cdot \max_{y \in [y_l, y_u]} p(y)$, this density fulfills the condition for the algorithm. In Figure 2, we see an illustration of the algorithm. The black line marks the interesting density and in gray, we see the rejection area. If this is large, the algorithm gets inefficient, because many samples are rejected. The smaller the gray area is, the more efficient is the algorithm.

Another possibility is to fix a vector of points $y_l = y_1 < y_2 < ... < y_C = y_u$. Then, calculate $p(y_1), ..., p(y_C)$ and

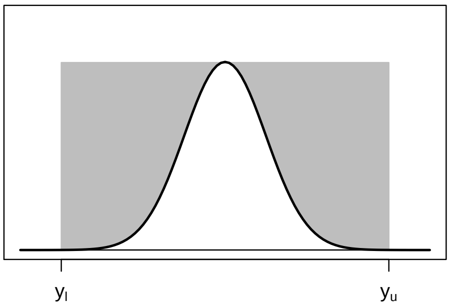


Figure 2: Illustration of rejection sampling, black line: interesting density, in gray: rejection area.

(i) generate a realization u from the uniform distribution $\mathcal{U}(0, \max_{j=1}^{C} p(y_j))$ and independently one index $j \sim \mathcal{U}\{1, ..., C\}$.

(ii) If $u \leq p(y_j)$, accept y_j as a sample from the distribution with density p otherwise, reject the candidate.

In the case of many samples to draw from the same distribution with density p, the second procedure can be much faster. We will investigate the runtimes later on. See Devroye (1986) for further details of the algorithm.

2.2.6. Inversion Method

Another possibility to draw random samples from a continuous distribution with distribution function $F(y) = \int_{-\infty}^{y} p(z)\, dz$ is the inversion method. For $U \sim \mathcal{U}(0, 1)$, the random variable $F^{-1}(U)$ has the distribution of interest. In many cases, this inverse function F^{-1} is not calculable. However, there is one possibility to fix an interval $[y_l, y_u]$ and to choose a vector of points $y_l = y_1 < y_2 < ... < y_C = y_u$. Then, one calculates $F(y_1), ..., F(y_C)$ and for a realization u from the uniform distribution, $\min\{y \in \{y_1, ..., y_C\} | F(y) \geq u\}$ can be seen as a sample from F. In the case of many samples to draw from the same distribution F this is a very suitable choice.

In the case of only few, or one, sample to draw, we present the binary search algorithm. Here, we also fix an interval $[y_l, y_u]$ and a realization u from the uniform distribution and in each step $F(\frac{y_l+y_u}{2})$ is compared to u.

This means, to draw one sample of distribution F, we choose a fineness degree fn and conduct the following.

(i) Generate a realization u from the uniform distribution $\mathcal{U}(0, 1)$.

(ii) As long as $y_u - y_l >$ fn:

if $F(\frac{y_l+y_u}{2}) < u$: $y_l = \frac{y_l+y_u}{2}$,

else $y_u = \frac{y_l+y_u}{2}$.

(iii) Return $\frac{y_l+y_u}{2}$ as sample of distribution F.

In Figure 3, we see an illustration of the binary search algorithm.

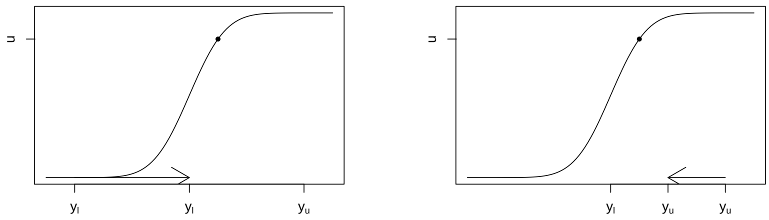


Figure 3: Illustration of inversion method with binary search.

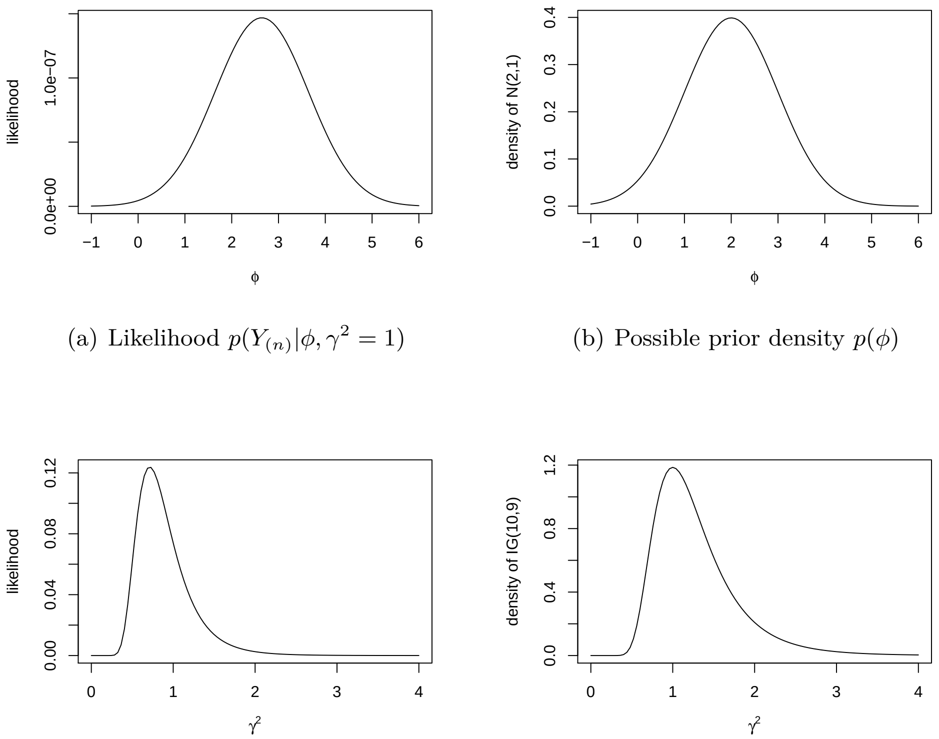
The problem of finding a suitable interval $[y_l, y_u]$ is the same for rejection sampling and inversion method. The only difference is that rejection sampling is very sensitive for runtimes when the interval is too large, because many candidates are rejected. See for further details of the algorithm, for example, Devroye (1986).

For both sampling methods, rejection sampling as well as inversion method, we have presented two versions. At first, one sample can be drawn by running the firstly presented steps (i) and (ii) of the rejection sampling, see Section 2.2.5, or steps (i)-(iii) of the inversion method. This can be repeated as often as desired. This is fast, if only few samples have to be drawn but can take long for a large number of samples. At second, based on a fixed candidate area, the distribution function, respectively the density, is calculated for a vector of candidates, among which a random sample is drawn. To draw only few samples, this can be inefficient, because it takes long to calculated the distribution function for a large vector of candidates. But to draw a large amount of samples, this is much faster than drawing each sample separately without candidate area. We will investigate this effect later on.

2.3. Sensitivity Analysis

In an informative Bayesian analysis, one usually is interested in the influence of the prior parameters. In reality, it will often be the case that no prior knowledge is available and, therefore, it is important to know what happens to the estimations, if the prior parameters are not correctly chosen. We again use Example 1. In Figure 4, we first motivate the choice of the normal distribution for the location parameter and the inverse gamma (IG) distribution for the variance parameter.

We compare three sample sizes $n \in \{20, 100, 500\}$ between $t_0 = 0$ and $t_n = 1$ and each two wrong prior means with three different prior variances. For $\phi \sim \mathcal{N}(m_\phi, V_\phi)$ we run the estimations with



(a) Likelihood $p(Y_{(n)}|\phi, \gamma^2 = 1)$

(b) Possible prior density $p(\phi)$

(c) Likelihood $p(Y_{(n)}|\phi = 2, \gamma^2)$

(d) Possible prior density $p(\gamma^2)$

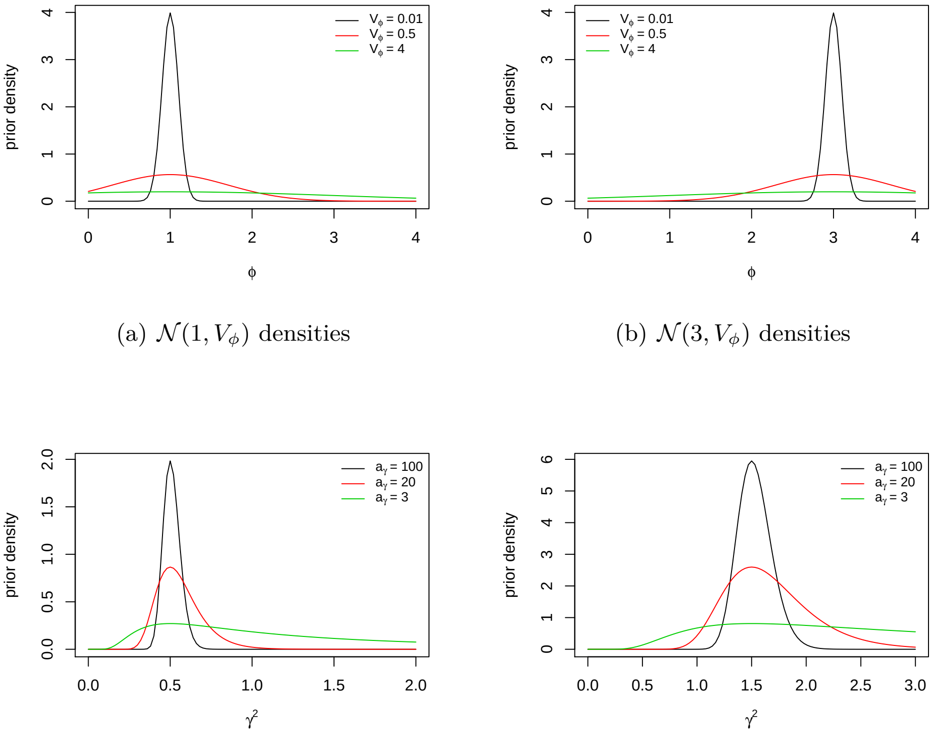
Figure 4: Motivation of prior distribution families for Example 1; top: comparison of likelihood with respect to ϕ (a) and density of a suitable prior $\mathcal{N}(2, 1)$ distribution (b), bottom: comparison of likelihood with respect to γ^2 (c) and density of a suitable prior IG$(10, 9)$ distribution (d).

$m_\phi \in \{1, 3\}$ whereby the parameter used for the simulation is $\phi = 2$ and $V_\phi \in \{0.01, 0.5, 4\}$ to compare large, medium and small influence of the prior distribution. For $\gamma^2 \sim \text{IG}(a_\gamma, b_\gamma)$ – with chosen value $\gamma^2 = 1$ in the simulation – we compare the prior means 0.5 and 1.5 and, therefore, choose $a_\gamma \in \{3, 20, 100\}$ and $b_\gamma \in \{(a_\gamma - 1) \cdot 0.5, (a_\gamma - 1) \cdot 1.5\}$. This can be seen with a look on the expectation of the IG(a_γ, b_γ):

$$\mathbb{E}[\gamma^2] = \frac{b_\gamma}{a_\gamma - 1}, \quad Var(\gamma^2) = \frac{b_\gamma^2}{(a_\gamma - 1)^2 (a_\gamma - 2)}$$

for $a_\gamma > 2$. This means a large variance for a_γ larger but close to 2 and a small variance for large a_γ. In Figure 5 we see the densities of the prior distributions.

For each parameter setting, we sample from the posterior with a Metropolis within Gibbs sampler. Details to the estimation itself are given in Section 3. For the sensitivity analysis of ϕ, we fix the prior parameters of γ^2 to $a_\gamma = 3$ and $b_\gamma = 2$ which leads to prior mean and standard deviation equal to the chosen value for $\gamma^2 = 1$. In Figure 6, we see the resulting posterior densities. In the left column, the prior mean is $m_\phi = 1$ and in the right column, it is $m_\phi = 3$.



(a) $\mathcal{N}(1, V_\phi)$ densities

(b) $\mathcal{N}(3, V_\phi)$ densities

(c) $\text{IG}(a_\gamma, (a_\gamma - 1) \cdot 0.5)$ densities

(d) $\text{IG}(a_\gamma, (a_\gamma - 1) \cdot 1.5)$ densities

Figure 5: Prior densities for ϕ with $V_\phi \in \{0.01, 0.5, 4\}$ (top) and γ^2 with $a_\gamma \in \{3, 20, 100\}$ (bottom).

For all considered sample sizes, the results are very similar. The short dotted line marks the chosen value $\phi = 2$ for simulation. The results are mirrored with respect to that line. In all cases, we can see that the very small variance, illustrated by the black line, can fix the posterior to the prior mean location. But even the red line, $V_\phi = 0.5$, leads to a biased estimation. The green line, $V_\phi = 4$, leads to good results. Only in the case of sample size $n = 20$, which is very small for a process with autoregressive variance, the posterior location is not exactly correct. By comparison of the left and right plots, it becomes clear that this is a result of only one simulation, since both posterior densities have the same shape and the same location despite of the different prior locations. The credibility interval would include the value $\phi = 2$.

In the sensitivity analysis of γ^2, we fix $m_\phi = 2$ and $V_\phi = 4$ with the same reason as above. In Figure 7 we see the resulting posterior densities. In the left column, we compare each of the prior variances $\frac{b_\gamma^2}{(a_\gamma-1)^2(a_\gamma-2)} \in \{0.0026, 0.0139, 0.25\}$ and in the right column $\frac{b_\gamma^2}{(a_\gamma-1)^2(a_\gamma-2)} \in \{0.023, 0.125, 2.25\}$. We see that for all cases the small prior variance, i.e. $a_\gamma = 100$, leads to biased estimates. For the small sample size as well as for $n = 100$, also the red line is influenced by the prior distribution. But in all cases,

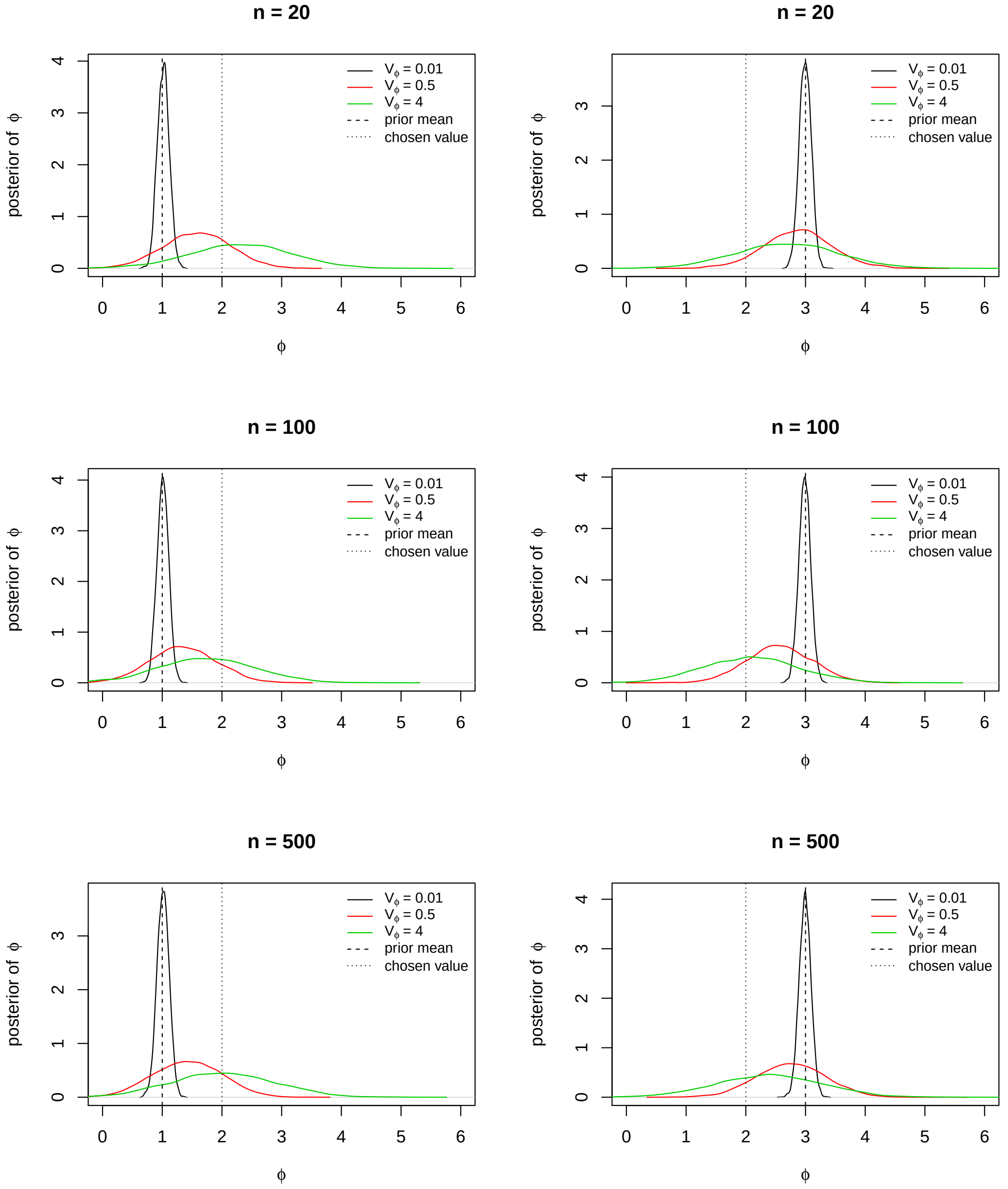


Figure 6: Posterior densities of ϕ with the corresponding prior densities in Figures 5(a) and 5(b), left: prior mean $m_\phi = 1$, right: prior mean $m_\phi = 3$.

the green line, i.e. $a_\gamma = 3$, shows good results. In the middle row, the posterior location lies not at the chosen parameter $\gamma^2 = 1$. But comparison of the support of the left and the right plot, both calculated with the same simulated realization of sample $Y_{(n)}$, shows that this is due to the fact of only one simulation similar to the upper row of Figure 6.

In summary, if no expert knowledge is available, the prior parameters will be chosen so that

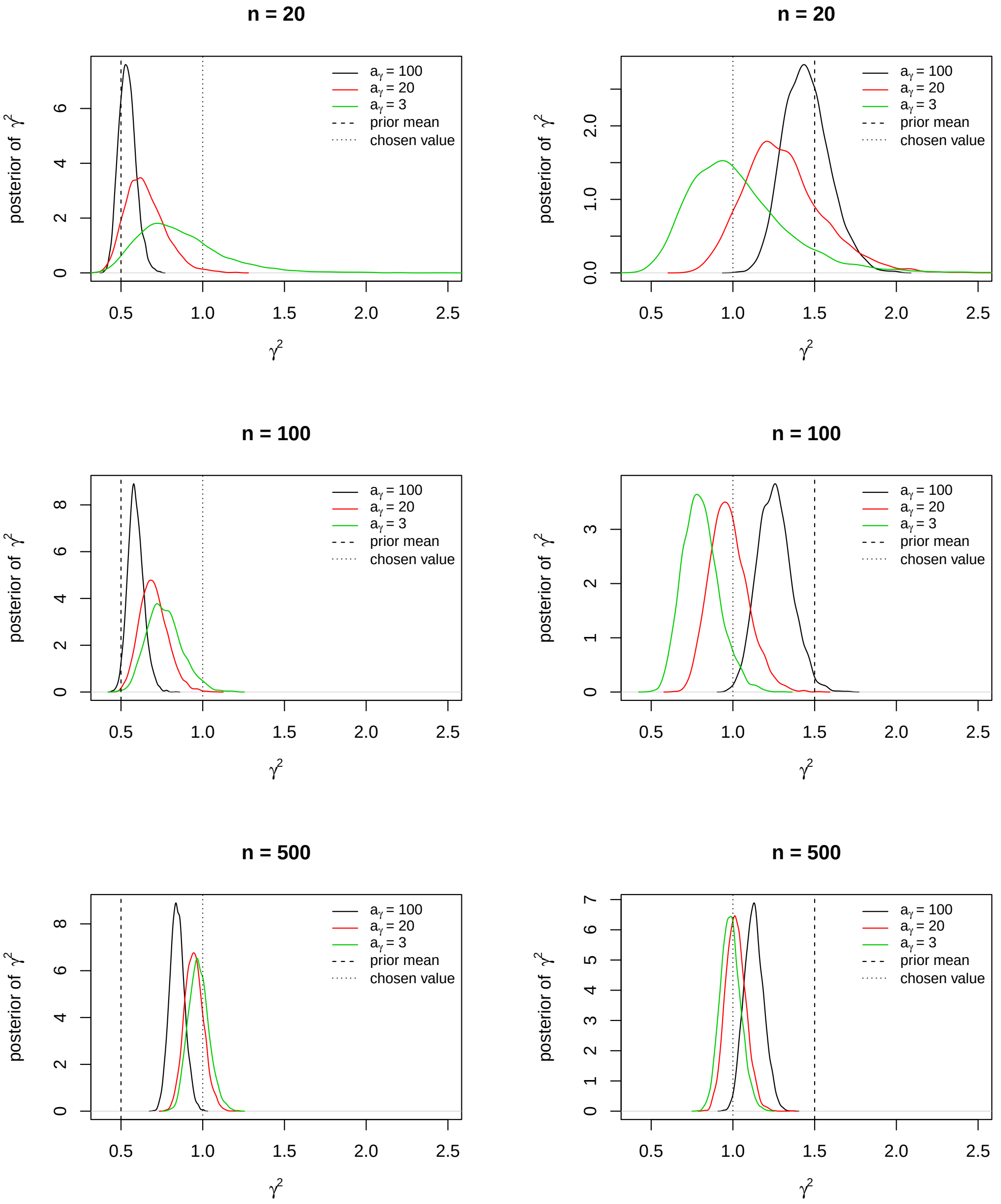


Figure 7: Posterior densities of γ^2 with the corresponding prior densities in Figures 5(c) and 5(d), left: prior mean $\frac{b_\gamma}{a_\gamma - 1} = 0.5$, right: prior mean $\frac{b_\gamma}{a_\gamma - 1} = 1.5$.

the expected value and the standard deviation are equal to each other, which would mean for the above example: $V_\phi = \phi^2$ and $a_\gamma = 3$. This will be done in the following investigations.

2.4. Bayesian Prediction

In Meeker and Escobar (1998), two different prediction problems are presented. Firstly, observations of one unit or individual are available and predictions for future observations for a new individual are of interest. This case is referred to as new-sample prediction. Secondly, future observations for the current individual have to be predicted. This is called within-sample prediction. We will use these names in the following as well.

There are several ways to predict a new observation variable Y^* at time t^*. In the frequentist estimation, in many cases, a predictive distribution is not calculable and, therefore, trajectories are simulated from the model with point estimates plugged in, see, for example, Chiquet et al. (2009). In Vidoni (2004), a predictive density for discrete-time stochastic processes is calculated, but prediction is only made for the next observation Y_{n+1}. In the Bayesian approach, it is common to take the samples from the posterior distribution, $\theta_1^*, ..., \theta_K^*$, and to simulate with each of them one trajectory. Altogether, K trajectories provide prediction samples, see, for example, Weinberg et al. (2007). If the density $p(Y^*|\theta)$ is explicitly available as it is, for example, in the case of an explicit solution of the SDE (2.1), the predictive distribution can be just calculated by a sampling procedure using the density $p(Y_{t^*}|\theta)$. This is presented in Blanke and Bosq (2015) for the Poisson and the Ornstein-Uhlenbeck (OU) process, in Jokiel-Rokita et al. (2014) for doubly stochastic Poisson processes, and in Dion, Hermann and Samson (2016) for the OU and the Cox-Ingersoll-Ross process within a mixed effects model. For a non-linear regression model see, for example, Yuan and Ji (2015); Robinson and Crowder (2000), which would be similar.

In the following, a novel approach to predict stochastic processes, based on the Euler approximation, is presented. This procedure is valuable for all processes defined by (2.1), including the estimation uncertainty as well as possible uncertainties from latent variables. We will first present an algorithm leading to a pointwise distribution of Y^* and afterwards, a second algorithm yielding a vector of process variables.

In Bayesian prediction, one naturally has a predictive distribution and can derive the point prediction or prediction intervals. The predictive distribution for a point of interest Y^* is given by

$$p(Y^*|Y_{(n)}) = \int p(Y^*|Y_{(n)}, \theta) \cdot p(\theta|Y_{(n)})\, d\theta,$$

which often is not analytically available. But, if we have samples $\theta_1^*, ..., \theta_K^* \sim p(\theta|Y_{(n)})$ from the posterior distribution, we can approximate the predictive density by

$$\begin{aligned} p(Y^*|Y_{(n)}) &= \int p(Y^*|Y_{(n)}, \theta) \cdot p(\theta|Y_{(n)})\, d\theta \\ &\approx \frac{1}{K} \sum_{k=1}^{K} p(Y^*|Y_{(n)}, \theta_k^*). \end{aligned}$$

In the case of within-sample prediction, the Markov property $p(Y^*|Y_{(n)}, \theta) = p(Y^*|Y_n, \theta)$ can be used. For a new-sample prediction, Y^* is a point of a new series and $p(Y^*|Y_{(n)}, \theta) = p(Y^*|\theta)$.

In the following, two algorithms for prediction are presented. The first algorithm considers the predictive distribution of one point of interest that can lie far away from the last observed value. In the case of a new series, this is equivalent to predicting a point that lies in large distance to the starting point. Since we base the model on the SDE (2.1), the Euler approximation can perform poorly for large time differences. This problem will be tackled. A second algorithm is presented to sample a trajectory vector. Sampling from an arbitrary multivariate distribution is not self-evident. Both algorithms can be used equally for the prediction of a new series as well as for prediction of the further development of the series. We declare the specific points with *(ia)* and *(ib)* in the algorithms.

Let us assume, we want to predict the process in time point $t^* \gg t_n$, respectively $t^* \gg t_0$. For large time distances, the Euler approximation can be inaccurate. Therefore, the idea of the algorithm is to go in M steps to the interesting value and in each to include the predictive distribution of the last step. Therefore, we choose a sampling partition $\{t_0, t_n\} \ni \tau_0 < \tau_1 < ... < \tau_M = t^*$, for example

$$\tau_m = m \cdot \Delta^* + \tau_0,\ m = 0, ..., M, \quad \Delta^* := \frac{t^* - \tau_0}{M}.$$

Let $p(Y_{m+1}^*|Y_m^*, \theta)$ denote the transition density of the Euler approximated variables $Y_0^* \in \{y_0, Y_n\}, Y_1^*, ..., Y_M^* = Y^*$ in $\tau_0, \tau_1, ..., \tau_M$.

In the following, we denote with $\theta_1^*, ..., \theta_K^*$ samples from the posterior distribution $p(\theta|Y_{(n)})$ for the models without latent variable. In the specific models with latent variables, the hierarchical and the jump models, the first step is to predict the latent variables, the random effect in the hierarchical model and the Poisson process in the jump models. For the jump diffusion, prediction for the Poisson process variables $N_{\tau_1}, ..., N_{\tau_M}$ is needed as a first step, see for example Hermann et al. (2016a). Then, $\theta_1^*, ..., \theta_K^*$ contain the posterior samples of the parameters as well as the prediction samples of the Poisson process variables. For the hierarchical model, for new-sample predictions, samples of the predictive distribution for the random effects are drawn, details are given in Section 3.2. In the following, $\theta_k^*, k = 1, ..., K$, will denote the vector of posterior and predictive samples jointly. We sample from the pointwise predictive distribution by running the following algorithm.

Algorithm 1
Take samples $\theta_k^* \sim p(\theta|Y_{(n)}),\ k = 1, ..., K$, from the posterior distribution, respectively from the predictive distribution of latent variables.

(i) Set starting values

(ia) $Y_0^{*(k)} := Y_n, k = 1, ..., K$: within-sample prediction,

(ib) $Y_0^{*(k)} := y_0, k = 1, ..., K$: new-sample prediction.

(ii) For $m = 1, ..., M$ draw K samples

$$Y_m^{*(1)}, ..., Y_m^{*(K)} \sim \frac{1}{K} \sum_{k=1}^{K} p\left(Y_m^*|Y_{m-1}^{*(k)}, \theta_k^*\right).$$

Sampling from the density can be conducted with inversion method or rejection sampling as described before. The advantage of this algorithm is that we can use a fixed vector of candidates and calculate the density in the second step only once and draw K samples vectorial, which makes it faster than methods that calculate the density separately for each sample as explained at the end of Section 2.2.6. We will see the different runtimes later on.

Algorithm 1 yields samples from the predictive distribution $p(Y_m^*|Y_{(n)}), m = 1, ..., M$, for each of the points $\tau_1, ..., \tau_M$, but no trajectories.

This can be seen as follows. For $M = 2$ it is

$$Y_1^{*(r)} \sim \frac{1}{K}\sum_{k=1}^{K} p\left(Y_1^*|Y_0^{*(k)}, \theta_k^*\right) \approx \int p(Y_1^*|Y_0^*, \theta)p(\theta|Y_{(n)})\, d\theta = p(Y_1^*|Y_{(n)}), \quad r = 1, ..., K,$$

for $Y_0^* \in \{y_0, Y_n\}$. In the second step it is

$$Y_2^{*(r)} \sim \frac{1}{K}\sum_{k=1}^{K} p\left(Y_2^*|Y_1^{*(k)}, \theta_k^*\right) \approx \int p(Y_2^*|Y_1^*, \theta)p(Y_1^*, \theta|Y_{(n)})\, d(Y_1^*, \theta) = p(Y_2^*|Y_{(n)})$$

for $r = 1, ..., K$. There are three approximations in one step as can be seen by

$$\begin{aligned} p(Y_2^*|Y_{(n)}) &= \int p(Y_2^*|Y_1^*, \theta)p(Y_1^*, \theta|Y_{(n)})\, d(Y_1^*, \theta) \\ &\approx \int p(Y_2^*|Y_1^*, \theta)p(Y_1^*|Y_{(n)})p(\theta|Y_{(n)})\, d(Y_1^*, \theta) && (2.4) \\ &\approx \frac{1}{K_1K_2}\sum_{k_1=1}^{K_1}\sum_{k_2=1}^{K_2} p\left(Y_2^*|Y_1^{*(k_1)}, \theta_{k_2}^*\right) && (2.5) \\ &\approx \frac{1}{K}\sum_{k=1}^{K} p\left(Y_2^*|Y_1^{*(k)}, \theta_k^*\right). && (2.6) \end{aligned}$$

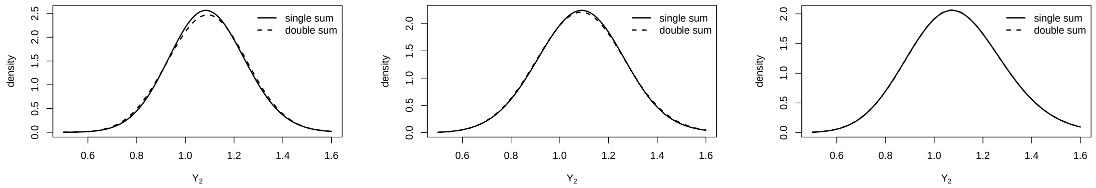
Approximation (2.4) is related to the fact that $p(Y_1^*, \theta|Y_{(n)}) = p(Y_1^*|\theta, Y_{(n)})p(\theta|Y_{(n)})$. In addition, it is

$$\begin{aligned} p(Y_1^*|\theta, Y_{(n)}) &= \frac{p(Y_1^*, \theta, Y_{(n)})}{p(\theta, Y_{(n)})} \\ &= \frac{p(Y_1^*, \theta, Y_{(n)})}{p(\theta|Y_{(n)})p(Y_{(n)})} \\ &= \frac{p(\theta|Y_1^*, Y_{(n)})}{p(\theta|Y_{(n)})}\frac{p(Y_1^*, Y_{(n)})}{p(Y_{(n)})} \\ &= \frac{p(\theta|Y_1^*, Y_{(n)})}{p(\theta|Y_{(n)})}p(Y_1^*|Y_{(n)}) \\ &\approx p(Y_1^*|Y_{(n)}). && (2.7) \end{aligned}$$

For large n, the term $\frac{p(\theta|Y_1^*, Y_{(n)})}{p(\theta|Y_{(n)})}$ is close to 1. In practice, the variables $Y_1^{*(k)}, k = 1, ..., K$, are even drawn from the predictive distribution based on $p(\theta|Y_{(n)})$ and, therefore, will not have additional information for the parameter.

Approximation (2.5) is explained by $p(Y_1^*|Y_{(n)}) \approx \frac{1}{K_1}\sum_{k_1=1}^{K_1} \mathbb{1}_{\{Y_1^{*(k_1)}\}}(Y_1^*)$ and $p(\theta|Y_{(n)}) \approx \frac{1}{K_2}\sum_{k_2=1}^{K_2} \mathbb{1}_{\{\theta_{k_2}^*\}}(\theta)$, which can be arbitrary good according to the choice of K_1 and K_2.

Approximation (2.6) is due to the fact that the expression (2.5) is computationally very costly. We will investigate this approximation through Example 1. In Figure 8, we see both densities, expression (2.5) and (2.6) for three different sample sizes $K = K_1 = K_2 \in \{10, 100, 1000\}$. For $K = 10$, a small difference is identifiable, but even for $K = 100$, the difference is small. In practice, even samples size $K = 1000$ is small for the approximation of the density, and there is no difference between the two curves. The runtimes for drawing K samples from the displayed densities are calculated ten times with an Intel(R) Core(TM) i5-3470 CPU (3.20GHz), 8 GB RAM, Windows 7 64bit personal computer. The average times are displayed in Table 1. For $K = 10$, the difference is negligible. But for growing samples size, this runtime saving decides over feasibility of the algorithm.



(a) $K = K_1 = K_2 = 10$ (b) $K = K_1 = K_2 = 100$ (c) $K = K_1 = K_2 = 1000$

Figure 8: Investigation of approximation (2.6) with Example 1; in dotted lines: $\frac{1}{K_1 K_2}\sum_{k_1=1}^{K_1}\sum_{k_2=1}^{K_2} p\left(Y_2^*|Y_1^{*(k_1)}, \theta_{k_2}^*\right)$, in solid lines: $\frac{1}{K}\sum_{k=1}^{K} p\left(Y_2^*|Y_1^{*(k)}, \theta_k^*\right)$, $K = K_1 = K_2 \in \{10, 100, 1000\}$.

Table 1: Overview of the runtimes in seconds for expression (2.5) and expression (2.6), averaged over 10 runs.

	$K = K_1 = K_2 = 10$	$K = K_1 = K_2 = 100$	$K = K_1 = K_2 = 1000$
expression (2.5)	0.038	1.694	199.136
expression (2.6)	0.021	0.040	0.187

Altogether, the approximations (2.4)-(2.6) are justified and $Y_2^{*(1)}, ..., Y_2^{*(K)}$ can be seen as samples from $p(Y_2^*|Y_{(n)})$. For arbitrary M, we can assume $Y_{M-1}^{*(k)} \sim p(Y_{M-1}^*|Y_{(n)}), k = 1, ..., K$, and with induction

$$Y_M^{*(r)} \sim \frac{1}{K}\sum_{k=1}^{K} p\left(Y_M^*|Y_{M-1}^{*(k)}, \theta_k^*\right) \approx \int p(Y_M^*|Y_{M-1}^*, \theta)p(Y_{M-1}^*, \theta|Y_{(n)})\, d(Y_{M-1}^*, \theta) = p(Y_M^*|Y_{(n)})$$

follows for $r = 1, ..., K$ with the same approximations as described before.

The following algorithm is made for the case that trajectories are the value of interest. Here, it is possible to draw not exactly M samples from the trajectory, but to choose a critical value

y_c and to stop drawing if it is reached. In addition, it is possible to draw L samples with $L \neq K$.

Algorithm 2
Take samples $\theta_k^* \sim p(\theta|Y_{(n)}),\ k = 1, ..., K$, from the posterior distribution, respectively from the predictive distribution of latent variables. For $l = 1, ..., L$ repeat the following steps.

(i) Set $m = 1$ and

(ia) $Y_0^{*(l)} := Y_n$: within-sample prediction,

(ib) $Y_0^{*(l)} := y_0$: new-sample prediction.

(ii) For $m = 1, 2, ...$ draw one sample

$$Y_m^{*(l)} \sim \frac{1}{K}\sum_{k=1}^{K} p\left(Y_m^*|Y_{m-1}^{*(l)}, \theta_k^*\right).$$

For notation simplicity, in the following, we restrict to the case of M trajectory entries.

The reason why this procedure leads to L independent samples of $(Y_1^*, ..., Y_M^*)$ can be seen as follows. At first, it is

$$\begin{aligned} p(Y_m^*|Y_{m-1}^{*(l)}, Y_{(n)}) &= \int p(Y_m^*|Y_{m-1}^{*(l)}, \theta)p(\theta|Y_{m-1}^{*(l)}, Y_{(n)})\, d\theta \\ &\approx \int p(Y_m^*|Y_{m-1}^{*(l)}, \theta)p(\theta|Y_{(n)})\, d\theta \\ &\approx \frac{1}{K}\sum_{k=1}^{K} p\left(Y_m^*|Y_{m-1}^{*(l)}, \theta_k^*\right), m = 1, ..., M. \end{aligned} \tag{2.8}$$

The first equation is just the Markov property $p(Y_m^*|Y_{m-1}^{*(l)}, Y_{(n)}, \theta) = p(Y_m^*|Y_{m-1}^{*(l)}, \theta)$. The last approximation is the same as in (2.5). Approximation (2.8) is the same as in (2.7) and, therefore, the same approximation as in (2.4). For large n, one sample $Y_{m-1}^{*(l)}$ does not effect the posterior. By the way, $Y_{m-1}^{*(l)}$ is itself drawn by the predictive distribution based on the same posterior samples. Therefore, even in finite sample sizes, the effect is negligible.

Afterwards, in iteration l, it is approximated

$$\begin{aligned} Y_1^{*(l)} &\sim p(Y_1^*|Y_{(n)}), \\ Y_2^{*(l)} &\sim p(Y_2^*|Y_1^{*(l)}, Y_{(n)}), \\ &\vdots \\ Y_M^{*(l)} &\sim p(Y_M^*|Y_{M-1}^{*(l)}, Y_{(n)}). \end{aligned}$$

Using the Markov property, the joint density is

$$p(Y_1^*, ..., Y_M^*|Y_{(n)}) = p(Y_1^*|Y_{(n)}) \cdot p(Y_2^*|Y_1^*, Y_{(n)}) \cdot ... \cdot p(Y_M^*|Y_{M-1}^*, Y_{(n)})$$

and we get L independent samples from $p(Y_1^*, ..., Y_M^*|Y_{(n)})$ with Algorithm 2.

To see the difference between the two sampling Algorithms 1 and 2, we will compare them in the setting of Example 1.

We run the following simulation study. 500 series with $n = 20$ are simulated to have a high estimation uncertainty. To simulate time-continuity, between each two time points, 99 equidistant points are inserted, the trajectories from the model are drawn, and afterwards, every 100th point is taken as observation. This means, for $t_0 = 0, t_1 = 0.05, ..., t_{20} = 1$ and $\Delta = \Delta_i = 0.05$, we simulate the series $\widetilde{Y}_0, \widetilde{Y}_1, ..., \widetilde{Y}_{2000}$ in $\widetilde{t}_0 = 0, \widetilde{t}_1 = 0.0005, ..., \widetilde{t}_{2000} = 1$ with $\widetilde{\Delta} = 0.0005$. Afterwards, $Y_0 = \widetilde{Y}_0, Y_1 = \widetilde{Y}_{100}, ..., Y_{20} = \widetilde{Y}_{2000}$ form the observed variables. See Iacus (2008); Phillips (1973) for further details. In Table 2, an overview of the various time distances of the Euler approximation can be found, that are used for simulation, estimation and prediction.

Table 2: Overview of the time distances used for the Euler approximation in the simulation of the data series, the observations variables, for prediction and for simulation of the new series.

	Δ_i for simulation	Δ_i of observation variables	Δ_i^* for prediction	Δ_i for new series
Figure 9	0.0005	0.05	0.05	0.0005
Figure 11	0.0005	0.05	0.05	0.05/0.0005
Figure 14	0.0005	0.05	0.05	0.05/0.005/ 0.0005/0.00005
Figure 15	0.000005	0.05	0.05	0.05/ 0.000005
Figure 16	0.02	0.02	see (2.11)	0.02
Figure 17	0.02	0.02	see (2.11)	0.02
Figure 18	0.000002	0.02	see (2.11)	0.000002

Afterwards, the parameters are estimated, information will be given in Section 3. With $K = 10\,000$ samples from the posterior distribution, we will compare four prediction methods. The first two are the ones presented in Algorithms 1 and 2. The third method is simple sampling from the model based on the point estimation, as often used in the frequentist approach. This procedure fits in the definition of naive prediction in Meeker and Escobar (1998) and will, therefore, be named *naive sampling* in the following. In detail, we run the following algorithm.

Algorithm 3 (Naive sampling)
Let $\widehat{\theta} = \frac{1}{K}\sum_{k=1}^{K}\theta_k^*$ be the approximated posterior mean. For $l = 1, ..., L$ repeat the following.

(i) Set $m = 1$ and

(ia) $Y_0^{*(l)} := Y_n$: within-sample prediction,

(ib) $Y_0^{*(l)} := y_0$: new-sample prediction.

(ii) For $m = 1, 2, ...$ draw one sample

$$Y_m^{*(l)} \sim p\left(Y_m^*|Y_{m-1}^{*(l)}, \widehat{\theta}\right).$$

The fourth one is straightforward sampling common in Bayesian analysis, where each of the K posterior samples is taken to simulate one trajectory from the model. All K trajectories would also form a prediction, where pointwise mean or quantiles could be used for point prediction or prediction intervals. We will name it *posterior sampling* in the following.

Algorithm 4 (Posterior sampling)
Take samples $\theta_k^* \sim p(\theta|Y_{(n)})$, $k = 1, ..., K$, from the posterior distribution, respectively from the predictive distribution of latent variables. For $l = 1, ..., L \leq K$ repeat the following.

(i) Set $m = 1$ and

(ia) $Y_0^{*(l)} := Y_n$: within-sample prediction,

(ib) $Y_0^{*(l)} := y_0$: new-sample prediction.

(ii) For $m = 1, 2, ...$ draw one sample

$$Y_m^{*(l)} \sim p\left(Y_m^* | Y_{m-1}^{*(l)}, \theta_l^*\right).$$

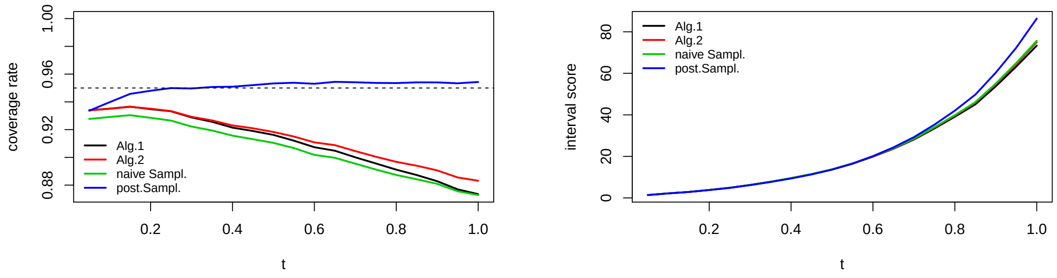
We need a criterion for the quality of the procedure. The coverage rate is an obvious criterion, since the level has to be kept. But the interval size is also interesting. Gneiting and Raftery (2007) propose an interval score as a combination of interval size, which should be small, and coverage rate, which should be equal to or larger than the level $1 - \alpha$. The score is given by

$$S(l, u, y) = u - l + \frac{2}{\alpha}(l - y)\mathbb{1}_{(-\infty,l)}(y) + \frac{2}{\alpha}(y - u)\mathbb{1}_{(u,\infty)}(y). \tag{2.9}$$

l denotes the lower and u the upper bound of the credible or prediction interval and y the true, i.e. the observed, value. In the case of a non-covering interval, the deviance from the respective bound is highly penalized by the deviance itself multiplied with $\frac{2}{\alpha}$. During this thesis, we will use $\alpha = 0.05$, this means, the difference between the observed value and the interval bound is penalized with $\frac{2}{\alpha} = 40$.

For the prediction quality, we simulate a set of 5000 new series in the same way as described above, i.e. with $\widetilde{\Delta} = \frac{\Delta}{100}$, and investigate for each of the 500 prediction intervals over time, if they include the new series or not. All results are averaged and can be seen in Figure 9. The coverage rates are depicted in Figure 9(a), where we can see that the blue line, i.e. the posterior sampling, leads to the highest rates, even higher than the level 0.95 in the end. The proposed methods, Algorithm 1 in black and Algorithm 2 in red, perform better than the naive sampling, but do not hold the level. As explained before, we are interested in precise prediction, also taking into account the interval sizes. For that reason we have a look at the interval score (2.9). The intervals of the posterior sampling are so large that they lead to a higher score than the other methods even with a higher coverage rate.

We see in Figure 10 two examples of the simulated series with the corresponding prediction intervals for the four methods. In gray, the 5000 simulated new series are shown. All methods have the lower bound in common. But the upper interval bounds differ. The



(a) Coverage rates

(b) Average interval score over time

Figure 9: Example 1: Comparison of the prediction methods in Algorithms 1 and 2, the naive sampling, Algorithm 3, and the posterior sampling, Algorithm 4, each depicted for time points $t_1 = 0.05, ..., t_{20} = 1$.

posterior sampling, i.e. the blue line, leads to the largest intervals which is not surprising, since we simulate with all the posterior samples instead of integrating out. The difference between the red and the green lines, i.e. the Algorithm 2 and the naive sampling, is small, and the lines are almost parallel. The procedures are comparable, the only difference is, that Algorithm 2 includes the estimation uncertainty whereby the naive sampling only takes the point estimations into account and the only uncertainty stems from the model. The smallest intervals are given by the black line, i.e. Algorithm 1, which is also not surprising, because it calculates pointwise intervals instead of simultaneous bands.

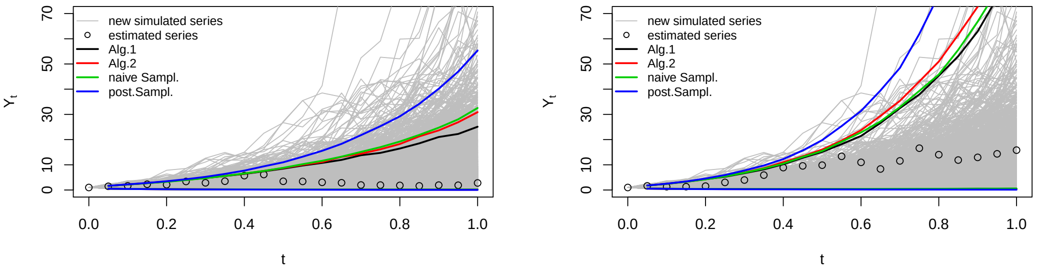


Figure 10: Prediction intervals with Algorithms 1 and 2, the naive sampling, Algorithm 3, and the posterior sampling, Algorithm 4, for two exemplary series, in gray: the 5000 simulated new series.

With the Euler approximation, we changed the variables for the inference. In the following, we compare what happens, if we simulate new series from the model assumed for the estimation without any simulation of time-continuity. That means, we simulate the series only in time points $t_0 = 0, t_1 = 0.05, ..., t_{20} = 1$. The resulting coverage rates are shown in Figure 11 in solid lines. The difference between the algorithms stays the same. But the coverage rate in the beginning is above the level and decreases afterwards. For comparison, the coverage rates from Figure 9(a) are displayed in dotted lines. We will investigate the influence of the Euler approximation in detail in the next section.

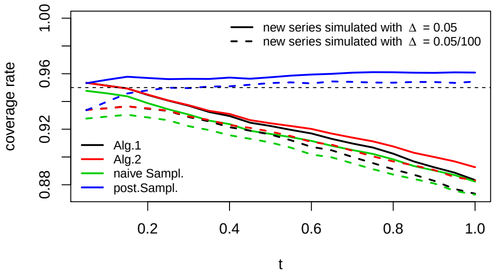


Figure 11: Comparison of coverage rates of Figure 9(a) (dotted lines) to the the coverage rates based on the Euler approximation variables in time points $t_0 = 0, t_1 = 0.05, ..., t_{20} = 1$ without simulation of time-continuity (solid lines).

Since the methods have to be applicable, runtimes are an interesting fact in the decision for one procedure. For Example 1, we test the runtimes with $n = 50$ and the proposed sampling algorithms. We take $K \in \{1000, 10\,000\}$ samples from the posterior and calculate, in each step, $K = L \in \{1000, 10\,000\}$ simulations from the predictive distribution to have a fair comparison. In addition, in each procedure, the candidate area to sample Y_{m+1}^* is chosen in the same way by

$$\Big[\min_{k=1}^{K} Y_m^{*(k)} + b(\widehat{\phi}, \tau_m, \widehat{Y}_m^*)\Delta^* - 5s(\sqrt{\widehat{\gamma}^2}, \tau_m, \widehat{Y}_m^*),$$
$$\max_{k=1}^{K} Y_m^{*(k)} + b(\widehat{\phi}, \tau_m, \widehat{Y}_m^*)\Delta^* + 5s(\sqrt{\widehat{\gamma}^2}, \tau_m, \widehat{Y}_m^*)\Big]$$

with $\widehat{\phi}$ and $\widehat{\gamma}^2$ the point estimates, i.e. the posterior mean, and $\widehat{Y}_m^* = \frac{1}{K}\sum_{k=1}^{K} Y_m^{*(k)}$.

As explained in Section 2.2.6, we compare the vectorial sampling to the separately drawing methods, each for the rejection sampling and the inversion method. As mentioned, the vectorial version is inefficient to draw only one sample, therefore, this is not used for Algorithm 2. In the case of vectorial sampling, we fix for comparison $C \in \{1000, 5000\}$ equidistant points $y_l = y_1 < y_2 < ... < y_C = y_u$ between the lower and upper bound of the interval. In the other case, we choose a fineness degree of $\frac{y_u - y_l}{C}$ to have a fair comparison. Again, the calculations are made with an Intel(R) Core(TM) i5-3470 CPU (3.20GHz), 8 GB RAM, Windows 7 64bit personal computer.

In Table 3, the resulting runtimes are displayed, averaged over 10 runs. We clearly see the advantage of Algorithm 1 that turns out in the vectorial sampling. In brackets, we can see the variances between the ten runs of the algorithms. As imaginable, the rejection sampling has higher variances, since the algorithm quality strongly depends on the candidate area, as many samples are rejected, if the candidate area is too large.

Table 3: Overview of the runtimes in seconds of the two presented algorithms for the rejection sampling (RS) and the inversion method (IM), averaged over 10 runs. In brackets: variances.

		Algorithm 1				Algorithm 2	
		vectorial					
		$C = 1000$	$C = 5000$	$C = 1000$	$C = 5000$	$C = 1000$	$C = 5000$
RS	$K = 10^3$	10.1 (0.04)	39.5 (0.03)	144.1 (42.7)	139.6 (6.4)	149.2 (22.3)	140.5 (15.8)
	$K = 10^4$	90.6 (1.4)	320 (0.7)	12 544 (35 493)	12 247 (141 312)	12 225 (50 399)	11 886 (49 696)
IM	$K = 10^3$	8.5 (0.01)	40.9 (0.4)	93.6 (3.4)	116.7 (0.8)	89.9 (0.8)	109.9 (0.02)
	$K = 10^4$	66.2 (0.1)	313 (0.02)	7668 (521.4)	9845 (23 383)	6979 (4452)	8853 (2880)

2.5. Influence of the Euler Approximation

In the following, we will consider the strengths and weaknesses of the Euler approximation. Originally, we assume a continuous-time process, which we approximate to a time series with possibly non-equidistant time points. Of course, both models behave differently. In the case of an explicit solution, this approximation should be avoided and inference should be made with the true transition density. However, in many cases, the model is defined by an SDE without an explicit solution or a solution exists, but is not easy to calculate. In these cases, the only possibility is an approximation. While more sophisticated schemes as, for example, the Milstein or the Runge-Kutta scheme, exist, see for example Iacus (2008); Fuchs (2013); Bruti-Liberati and Platen (2007), in this thesis, the Euler approximation is used. Of course, from the theoretical point of view, the Euler scheme can be seen as starting point and it can be future work to extend the theory to other schemes. However, for the considered data sets in this work, inference and prediction based on the Euler approximation works well.

At first, we will compare through the first example the explicit solution to the Euler approximated variables, simulated with different time distances. Secondly, we have a look at the results of the simulation study in the last section and consider the quality of the prediction intervals for the Euler variables with different time distances. Thirdly, we investigate what happens, if the time distances of the estimation variables and the prediction variables are different.

Simulating the Euler error

The process in Example 1 has the explicit solution

$$Y_t = y_0 \exp(\phi t - \frac{\gamma^2}{2} t + \gamma W_t), \tag{2.10}$$

also known as geometric Brownian motion, see, e.g., Øksendal (2003). Therefore, we can compare the process simulated with the explicit solution to the process simulated with the Euler approximation based on different time distances. We investigate the process in time points with $\Delta_i = 0.01$ and compare the Euler approximation variables in (2.2) based on time distances $\frac{\Delta_i}{10}, \frac{\Delta_i}{100}, \frac{\Delta_i}{1000}$ and $\frac{\Delta_i}{10\,000}$. We simulate 100 series for the Brownian motion $\{W_t, t \in [0,1]\}$ in the time points with $\frac{\Delta_i}{10\,000}$ to have the same realizations each.

In Figure 12, we see the average of squared differences between (2.10) and the Euler approximated process in time points $t_1 = 0.01, ..., t_{100} = 1$. The differences between the squared errors are large. However, in Figure 13, we can see one example of the 100 repetitions of Figure 12 and the lines are visually hardly to distinguish. This means, for a real data set in the case of no available explicit solution for the SDE, the Euler scheme is a reasonable tool to fit the model approximately to the data.

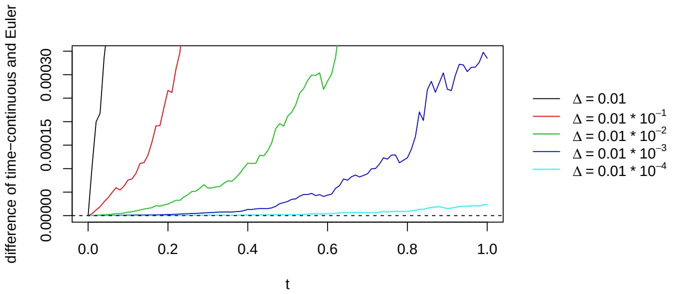


Figure 12: Squared differences between the continuous-time process (2.10) and the Euler approximated variables based on time distances $\Delta_i = 0.01$, $\frac{\Delta_i}{10} = 0.01 \cdot 10^{-1}$, $\frac{\Delta_i}{100} = 0.01 \cdot 10^{-2}$, $\frac{\Delta_i}{1000} = 0.01 \cdot 10^{-3}$ and $\frac{\Delta_i}{10\,000} = 0.01 \cdot 10^{-4}$, averaged over 100 repetitions, in time points $t_1 = 0.01, ..., t_{100} = 1$.

Quality of prediction with Euler approximated variables for continuous-time process

As we have seen in the simulation study of the last section, the Euler approximation has a strong influence on the prediction quality. Similar to Figure 11, we see in Figure 14 the coverage rates for the four prediction interval coverage rates, if we draw the new series with Δ_i, $\Delta_i/10$, $\Delta_i/100$ (as for the simulations for estimation and prediction) and $\Delta_i/1000$ from the process in Example 1. We can see that the coverage rates for the simulations drawn from the same Euler variables as used for estimation and prediction, are the highest, while we can see nearly no differences between the other three cases.

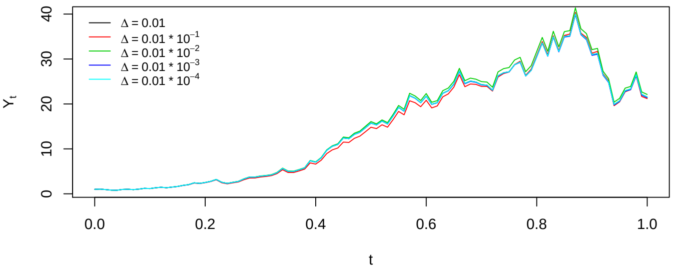
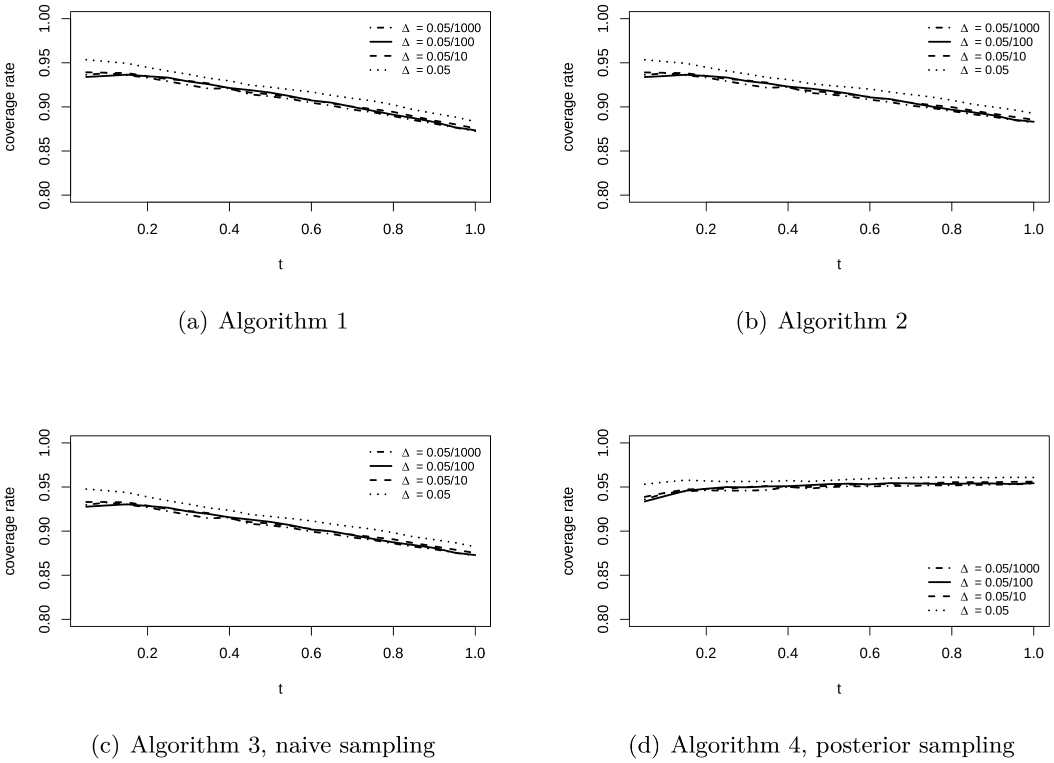


Figure 13: The Euler approximated variables, based on time distances $\Delta_i = 0.01$, $\frac{\Delta_i}{10} = 0.01 \cdot 10^{-1}$, $\frac{\Delta_i}{100} = 0.01 \cdot 10^{-2}$, $\frac{\Delta_i}{1000} = 0.01 \cdot 10^{-3}$ and $\frac{\Delta_i}{10\,000} = 0.01 \cdot 10^{-4}$, of one of the 100 repetitions of Figure 12 in time points $t_1 = 0.01, ..., t_{100} = 1$.



(a) Algorithm 1

(b) Algorithm 2

(c) Algorithm 3, naive sampling

(d) Algorithm 4, posterior sampling

Figure 14: Comparison of the prediction interval coverage rates for the new simulated series on different grids to simulate time-continuity. Each with Algorithms 1 and 2, the naive sampling, Algorithm 3, and the posterior sampling, Algorithm 4, depicted for time points $t_1 = 0.05, ..., t_{20} = 1$.

The same simulation was made for a similar process with the same drift function, but with constant variance function. This process results in a linear curve, where the Euler approximation works equally well for all time distances. Therefore, the coverage rates are equal, regardless, whether the new series are simulated with the time distances of the simulation or them for the estimation.

To investigate this effect closer, we try a non-linear drift function:

Example 2
We consider the process (2.1) *with* $h(\eta, t, y) = 0$, $s(\gamma, t, y) = \gamma$ *and* $b(\phi, t, y) = \phi_2 y(1 - \frac{1}{\phi_1} y)$, *which is known as the logistic curve in the literature, see, for example, Donnet et al. (2010). The process is approximated by*

$$Y_i = Y_{i-1} + \phi_2 Y_{i-1}(1 - \frac{1}{\phi_1} Y_{i-1})\, \Delta_i + \gamma \sqrt{\Delta_i}\, \zeta_i, i = 1, ..., n,$$

with $Y_0 = y_0$ *and* $\zeta_i \sim \mathcal{N}(0, 1)$. *We simulate random samples of the process with* $y_0 = 1, \phi = (10, 7)$, $\gamma = \frac{1}{2}$ *and* $t \in [0, 1]$.

We repeated the same simulation study as for Example 1, but with a 10 000 times finer grid for the time points to simulate time-continuity, see also Table 2 on page 21. In Figure 15, we can see clearly the difference between the coverage rates for the 5000 simulated data series, first in solid lines: with the time distances from the observation variables, second in dotted lines: with the time distances used for the simulation, that means 10 000 times finer than the observation variables to simulate the continuous-time process.

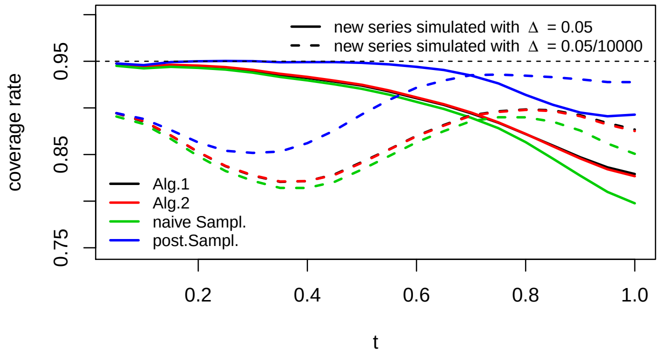


Figure 15: Comparison of the prediction interval coverage rates for the new simulated series on the grid used for simulation (dotted line) and on the grid of the estimation variables (solid line) in the non-linear Example 2. Each with Algorithms 1 and 2, the naive sampling, Algorithm 3, and the posterior sampling, Algorithm 4, depicted for time points $t_1 = 0.05, ..., t_{20} = 1$.

Of course, in real data examples, one can only assume a model, where data stem from. Originally, we started with a continuous-time stochastic process, which can only be simulated by the Euler approximation with very small time distances, as we have done for the simulations. But the process is only discretely observed. The described algorithms are developed to achieve a smaller grid for the prediction. The question is, if the prediction quality increases with smaller or larger time distances compared to these from the data used for estimation.

Different time distances in estimation and prediction

In the following, we want to investigate the consequence of different time distances in the estimation and the prediction. If the time distances are equal, the Euler approximation is just a model change and prediction works with the estimations on the same model. But in the case of non-equidistant time points or predictions made on other time differences Δ^* than these from the data, $\Delta_i, i = 1, ..., n$, this could be a problem.

We run a simulation study similar to the one before, but with $n = 50$. Here, we only consider Algorithm 1, since the Euler approximation is the same for all methods, Algorithm 1 to 4, and, therefore, the results would be virtually the same.

We will investigate the topic for two examples in the following. Firstly, the process seen in Example 1, which is a diffusion with linear drift and autoregressive variance, will be used. Secondly, Example 2 with non-linear drift function and constant variance will be compared.

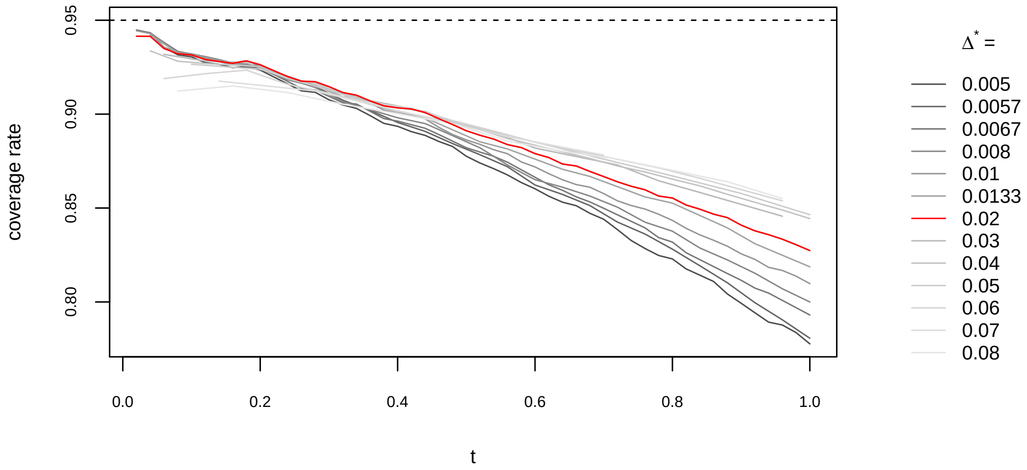
We here simulate 100 series with $n = 50$ and $t_0 = 0, ..., t_n = 1$, i.e. $\Delta_i = \Delta = 0.02$, but without a smaller grid for the simulation of time-continuity. Hence, we should only see the effect of the different time distances for estimation and prediction without any side effects. We compare the following time differences for prediction:

$$\Delta^* \in \left\{ \frac{\Delta}{4}, \frac{2\Delta}{7}, \frac{\Delta}{3}, \frac{2\Delta}{5}, \frac{\Delta}{2}, \frac{2\Delta}{3}, \Delta, 1.5\Delta, 2\Delta, 2.5\Delta, 3\Delta, 3.5\Delta, 4\Delta \right\}. \tag{2.11}$$

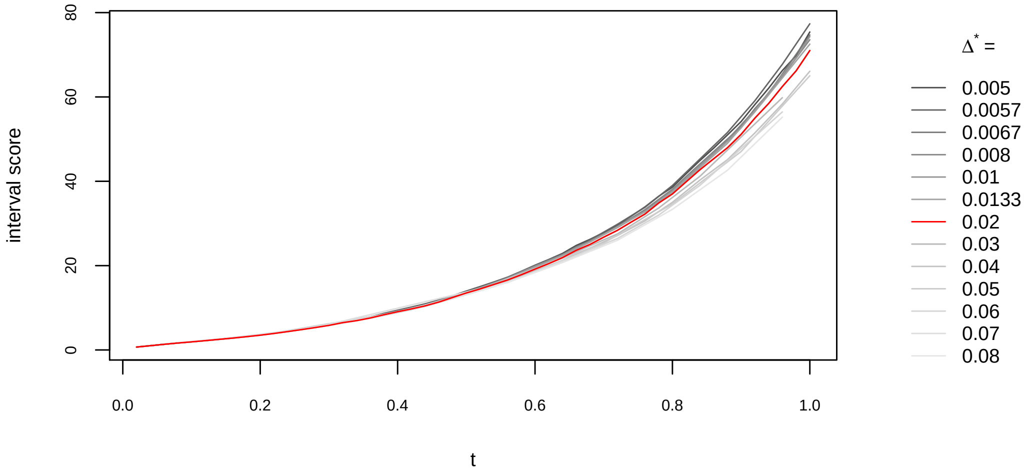
After calculating the the prediction intervals, we draw 5000 new series with time distances Δ of the estimation variables. In Figures 16 and 17, the average coverage rate over all 100 prediction intervals and all 5000 new series are depicted for the two examples.

In Figure 16, we see the resulting coverage rates and the average interval score for Example 1. In red, $\Delta^* = \Delta = 0.02$ is marked. The darker lines are for smaller time differences, the lighter lines for larger time differences. The prediction intervals based on the larger time distances, have higher coverage rates and smaller interval scores in the second half of the time range.

For Example 2, a totally different picture occurs in Figure 17. The non-linear drift function has a large effect. Large time distances lead to a small coverage rate which lies clearly below the level in the first half of the time range. In all cases, the red line lies in the middle and close to the level. Therefore, we recommend to take the time difference of the observation series. If they are not equidistant, a small time distance is suitable for prediction, since the dark lines do not lie far away from the red one.



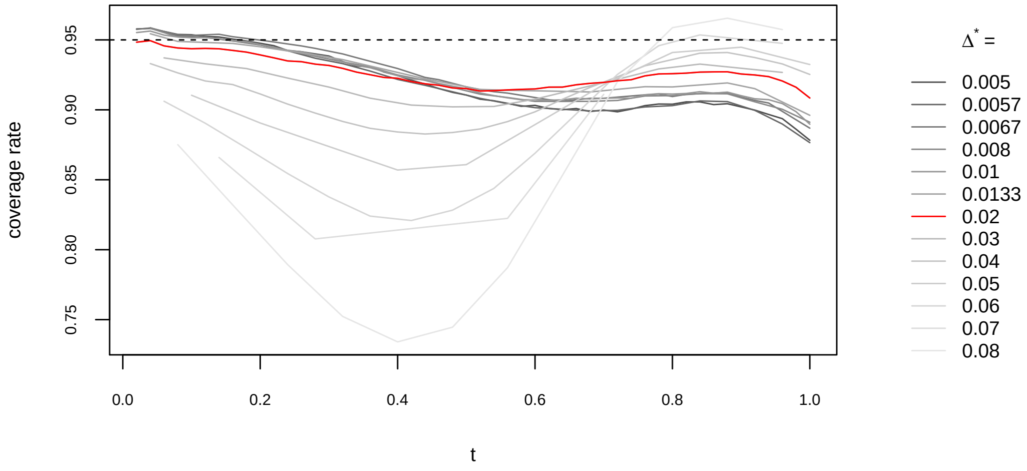
(a) Coverage rates



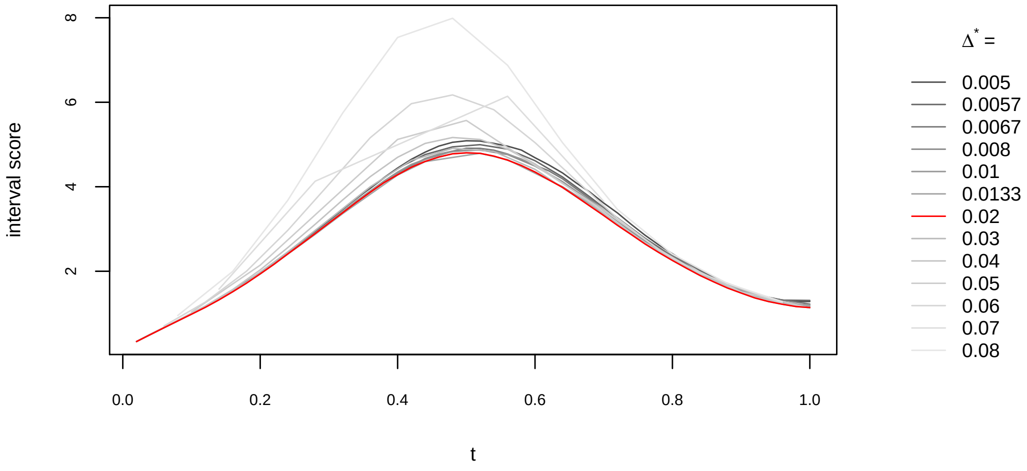
(b) Average interval scores

Figure 16: Investigation of different time distances for prediction with Example 1 (Algorithm 1); right: legend for Δ^*; red line: $\Delta^* = 0.02$, which is the time distance of the simulations used for estimation. In (a): amount of prediction intervals covering the 5000 new simulated series; in (b) the corresponding interval scores, both in time points $t_0 = 0, t_1 = 0.02, ..., t_{50} = 1$.

Until now, we only simulated the Euler approximation variables without simulation of time-continuity. It is interesting, if there is some change of the results of Figures 16 and 17, if we simulate the process with smaller time distances. In Figure 18, we see the results of the same simulation study as for Figure 17, but with simulation of the process on a 10 000 times finer grid, see also Table 2 on page 21. The coverage rates are all a bit smaller than in Figure 17, which fits into



(a) Coverage rates

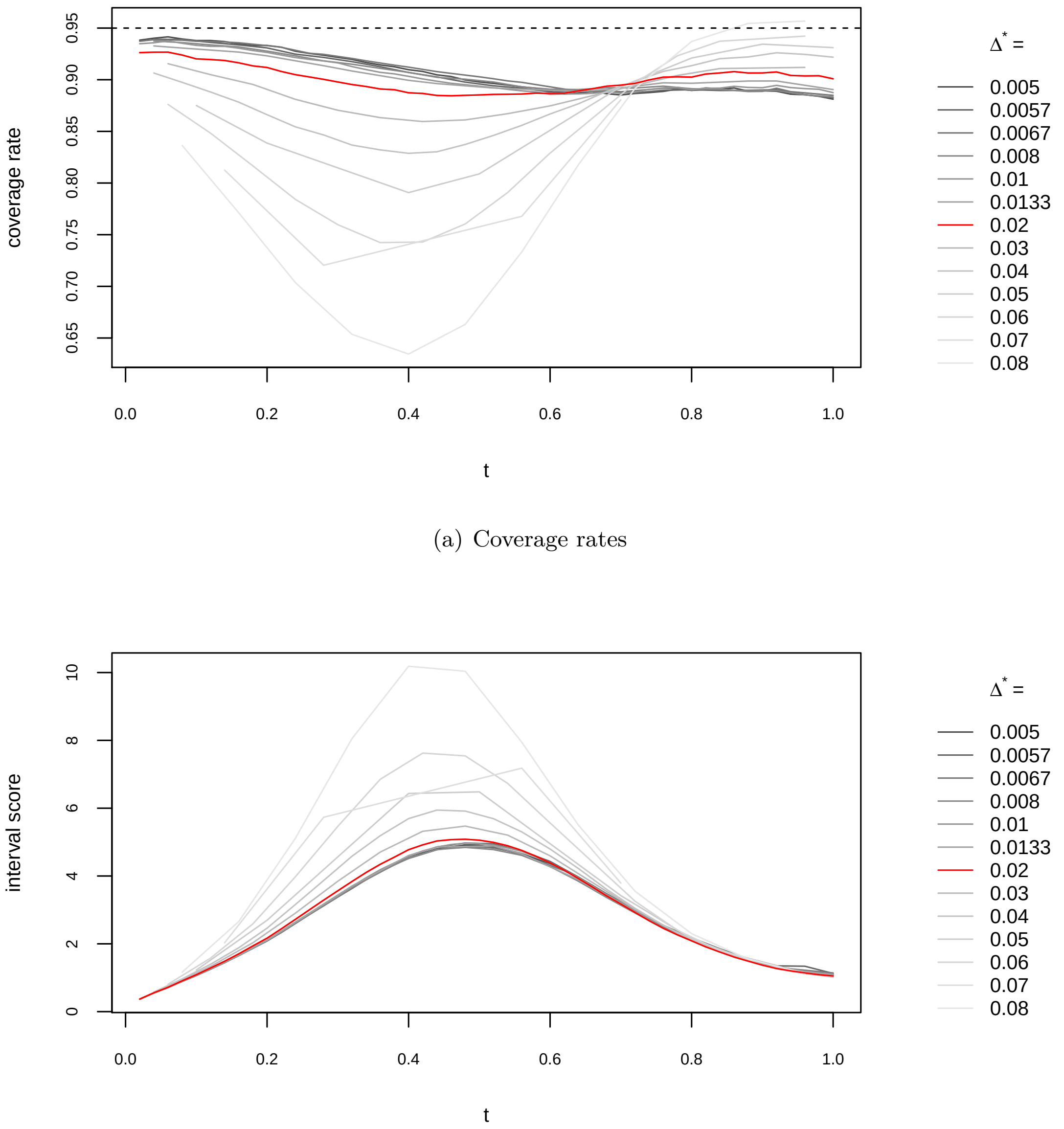


(b) Average interval scores

Figure 17: Investigation of different time distances for prediction with Example 2 (Algorithm 1); right: legend for Δ^*; red line: $\Delta^* = 0.02$, which is the time distance of the simulations used for estimation. In (a): amount of prediction intervals covering the 5000 new simulated series; in (b) the corresponding interval scores, both in time points $t_0 = 0, t_1 = 0.02, ..., t_{50} = 1$.

the results from Section 2.4. But the curves among themselves provide the same picture as Figure 17.

To conclude the results of this section, the prediction for the continuous-time process becomes better with small time distances, but on the other hand, the prediction quality is best, if



(a) Coverage rates

(b) Average interval scores

Figure 18: Repeated simulation study from Figure 17, but simulations made with a 10 000 times finer grid as the observation variables.

the time distances in the prediction are the same as in the simulated data used for estimation. In summary, it could be a good solution to use a data augmentation approach for the estimation to assume Euler variables based on smaller time distances, which are augmented inside a Gibbs sampler, since they are not completely observed. Afterwards the same Euler variables are used for the prediction procedure. A suitable data augmentation scheme is presented in Fuchs (2013). Implementing this to the procedures of this thesis will be future work.

In the remainder of the thesis, simulation studies will be made without simulating time-continuity. That means, simulated data will be generated with the same Euler variables as used in the estimation and prediction to avoid the need of distinguishing between the quality of the prediction procedure and the quality of the Euler approximation.

Part I. Bayesian Analysis and Prediction for Models Based on Continuous Processes

In many research areas of engineering, material fatigue plays an important role. Experiments often are very expensive because they take a long time and the constructions are costly. To extract as much information as possible from existing experiments and to predict fatigue, statistical models are a valuable tool. One of these existing experiments was conducted by Virkler et al. (1979). Sixty-eight replicate constant amplitude tests in aluminum alloy were carried out to investigate the fatigue crack propagation. In each of these tests, the number of cycles that leads to fixed crack lengths was observed. Against the natural assumption that a value is observed at fixed times, here the time is the dependent variable and the crack length is the independent variable. In Figure 19 we can see the number of cycle counts divided by 10 000 plotted against the crack lengths in *mm*. There are 68 observation series each with 164 observation points.

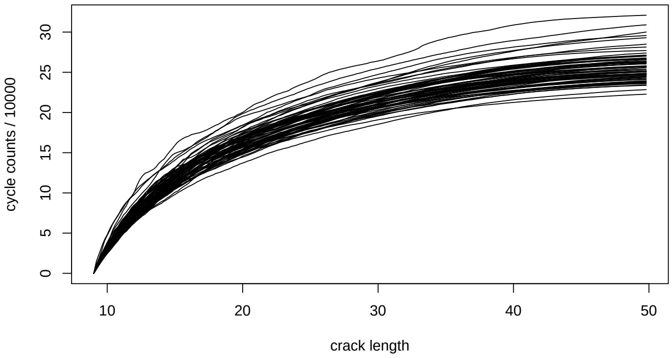


Figure 19: Observation series of the 68 experiments of Virkler et al. (1979).

In Hermann et al. (2016b), we compared several growth curves for these data and applied each in a regression and a diffusion model, both hierarchical to include knowledge from all the curves into estimation. Prediction for the further development of one series was made. It turned out, that the non-linear regression model can compete with the diffusion model for prediction in the far future. For predictions in the near future, the diffusion model is more suitable, because the Markov property leads to a small uncertainty close to the last observed value.

In the data series in Figure 19, no jumps are recognizable which leads to the assumption of a continuous stochastic process. Therefore, all over this chapter, we investigate the process defined by the SDE in (2.1) with the special case of $h(\eta, t, y) = 0$, at first for the diffusion process itself, and afterwards, in Section 4, for a noisy observed diffusion.

3. Diffusion Process

In the first section of Chapter I, we first present a single series model with the estimation and prediction procedure and afterwards extend the approach to the hierarchical model. Since prediction of the further development of one series in a hierarchical model is exactly the same as in the single series model, we restrict in the hierarchical model to the case of prediction of a new series.

3.1. Single Series Model

Estimation

In the following, an estimation procedure based on the Euler approximation of the process defined by the SDE (2.1) with $h(\eta, t, y) = 0$ is given. We here restrict to the simple case of estimation with the transition density of the Euler approximated variables in (2.2). Fuchs (2013) presents a data augmentation approach for the estimation based on the SDE in the case of large inter-observation times in the data.

The transition distribution is given in the Bayes model

$$\begin{aligned} Y_i|Y_{i-1}, \phi, \gamma^2 &\sim \mathcal{N}(Y_{i-1} + b(\phi, t_{i-1}, Y_{i-1})\,\Delta_i, s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i),\ i = 1, ..., n, \\ \phi|m_\phi, V_\phi &\sim \mathcal{N}(m_\phi, V_\phi), \quad V_\phi = \operatorname{diag}(v_1^2, ..., v_p^2), \\ \gamma^2|a_\gamma, b_\gamma &\sim \operatorname{IG}(a_\gamma, b_\gamma). \end{aligned} \tag{3.1}$$

Therefore, the likelihood of the observation variables is given by

$$\begin{aligned} p(Y_0, ..., Y_n|\phi, \gamma^2) &= \prod_{i=1}^n p(Y_i|Y_{i-1}, \phi, \gamma^2) \qquad (3.2) \\ =&\frac{1}{\sqrt{2\pi}^n} \prod_{i=1}^n \frac{1}{\sqrt{\Delta_i} s(\gamma, t_{i-1}, Y_{i-1})} \exp\left(-\frac{1}{2\Delta_i s^2(\gamma, t_{i-1}, Y_{i-1})} (Y_i - Y_{i-1} - b(\phi, t_{i-1}, Y_{i-1})\Delta_i)^2\right). \end{aligned}$$

In the case of a non-linear function b, a MH algorithm for ϕ is suitable, described in Section 2.2.1, with $\theta = \phi$. For generalization, only this case is considered. Parts of ϕ, which are linear in b, could be estimated with a conjugate prior, see, for example, Donnet et al. (2010).

In the case of a specific variance function $s(\gamma, t, y)$ in the form $s(\gamma, t, y) = \gamma \cdot \widetilde{s}(t, y)$, a full conditional posterior can be calculated. Examples are $\widetilde{s}(t, y) = 1$ with constant variance behavior, $\widetilde{s}(t, y) = t$ to have a variance structure dependent on the time or $\widetilde{s}(t, y) = y$ that leads to an autoregressive variance dependent on the current value of the process.

In this case, the full conditional posterior of γ^2 is given by

$$\gamma^2|Y_{(n)}, \phi \sim \operatorname{IG}\left(a_\gamma + \frac{n}{2}, b_\gamma + \frac{1}{2}\sum_{i=1}^n \frac{(Y_i - Y_{i-1} - b(\phi, t_{i-1}, Y_{i-1})\Delta_i)^2}{\widetilde{s}^2(t_{i-1}, Y_{i-1})\Delta_i}\right).$$

This can be seen by

$$\begin{aligned}
p(\gamma^2|Y_{(n)},\phi) &= p(\gamma^2)p(Y_{(n)}|\gamma^2,\phi)\\
\propto &(\gamma^2)^{-a_\gamma-1}\exp\left(-\frac{b_\gamma}{\gamma^2}\right)\cdot\frac{1}{(\gamma^2)^{\frac{n}{2}}}\exp\left(-\frac{1}{2\gamma^2}\sum_{i=1}^n\frac{(Y_i-Y_{i-1}-b(\phi,t_{i-1},Y_{i-1})\Delta_i)^2}{\widetilde{s}^2(t_{i-1},Y_{i-1})\Delta_i}\right)\\
= &(\gamma^2)^{-(a_\gamma+\frac{n}{2})-1}\exp\left(-\frac{1}{\gamma^2}\left(b_\gamma+\frac{1}{2}\sum_{i=1}^n\frac{(Y_i-Y_{i-1}-b(\phi,t_{i-1},Y_{i-1})\Delta_i)^2}{\widetilde{s}^2(t_{i-1},Y_{i-1})\Delta_i}\right)\right).
\end{aligned}$$

In the application later on as well as in the package BaPreStoPro, only $s(\gamma,t,y)=\gamma\cdot\widetilde{s}(t,y)$ is considered. In the case of a general variance function $s(\gamma,t,y)$, estimation of γ^2 can be conducted through a MH algorithm as well.

Sampling from the posterior of both parameters ϕ and γ^2 jointly can be performed with a Metropolis within Gibbs sampler, also described in Section 2.2.2, with $\theta=(\theta_1,\theta_2)=(\phi,\gamma^2)$.

Prediction

As explained in detail in Section 2.4, predictions can be made with one of the two presented Algorithms 1 and 2. The transition density $p(Y_m^*|Y_{m-1}^*,\theta)$ is the density of the normal distribution

$$\mathcal{N}\left(Y_{m-1}^*+b(\phi,\tau_{m-1},Y_{m-1}^*)\Delta^*, s^2(\gamma,\tau_{m-1},Y_{m-1}^*)\Delta^*\right) \tag{3.3}$$

with $\theta=(\phi,\gamma^2)$ and starting value $Y_0^{*(k)}=Y_n$ for within-sample prediction or $Y_0^{*(k)}=y_0$ for new-sample prediction, $k=1,...,K$.

3.2. Hierarchical Model

In many cases, not only one series is observed. For the example of crack growth, several test specimen are stressed over time. The fatigue behavior should be the same for all specimen but probably there are small differences between the individuals. This effect is taken into account by a hierarchical model. This means, the parameters, in our case only the location parameter ϕ, are assumed to be random with a mixture distribution. We here restrict to the case of the normal distribution, but other choices are possible as well with little effort.

We define the Euler approximated model

$$\begin{aligned}
Y_{ij}|Y_{i-1,j},\phi_j,\gamma^2 &\sim \mathcal{N}(Y_{i-1,j}+b(\phi_j,t_{i-1,j},Y_{i-1,j})\,\Delta_{ij}, s^2(\gamma,t_{i-1,j},Y_{i-1,j})\Delta_{ij}),\ i=1,...,n_j,\\
\phi_j|\mu,\Omega &\sim \mathcal{N}(\mu,\Omega)\text{ i.i.d., } j=1,...,J,\quad \Omega=\mathrm{diag}(\omega_1^2,...,\omega_p^2),\\
\mu|m_\mu,V_\mu &\sim \mathcal{N}(m_\mu,V_\mu),\quad V_\mu=\mathrm{diag}(v_1^2,...,v_p^2),\\
\omega_r^2|a_{\omega,r},b_{\omega,r} &\sim \mathrm{IG}(a_{\omega,r},b_{\omega,r}),\ r=1,...,p,\\
\gamma^2|a_\gamma,b_\gamma &\sim \mathrm{IG}(a_\gamma,b_\gamma)
\end{aligned} \tag{3.4}$$

with $\Delta_{ij}=t_{ij}-t_{i-1,j}$ and $Y_{0j}=y_0(\phi_j), j=1,...,J$. Denote with $Y_{(n_j,j)}$ the jth observation vector and $Y_{(n\cdot,J)}$ all the observations $\{Y_{ij}\}_{i=1,...,n_j,j=1,...,J}$.

Estimation

As mentioned, the estimation procedure can also be found in Hermann et al. (2016b) for fixed $y_0 = y_0(\phi)$.

Since Y_{0j} depends on the random effect, the likelihood is given by

$$p(Y_{(n\cdot,J)}|\phi_1, ..., \phi_J, \gamma^2) = \prod_{j=1}^{J} p(Y_{0j}|\phi_j) \prod_{i=1}^{n_j} p(Y_{ij}|Y_{i-1,j}, \phi_j, \gamma^2)$$

with $p(Y_{0j}|\phi_j) = \mathbb{1}_{\{y_0(\phi_j)\}}(Y_{0j})$.

We assume a conjugate normal prior distribution of μ with mean m_μ and diagonal variance matrix V_μ. In Hermann et al. (2016b), the matrix representation of the posterior distribution can be found. Since V_μ and Ω are assumed to be diagonal in our model, we can calculate the posterior distribution for each component of μ. It is

$$\begin{aligned}
&p(\mu_r|\phi_1, ..., \phi_J, \Omega) = p(\mu_r)p(\phi_1, ..., \phi_J|\mu, \Omega)\\
&\propto \exp\left(-\frac{1}{2}\frac{(\mu_r - m_{\mu,r})^2}{v_r^2}\right) \cdot \exp\left(-\frac{1}{2}\frac{1}{\omega_r^2}\sum_{j=1}^{J}((\phi_j)_r - \mu_r)^2\right)\\
&\propto \exp\left(-\frac{1}{2}\left\{\frac{\mu_r^2 - 2\mu_r m_{\mu,r}}{v_r^2} + \frac{1}{\omega_r^2}\sum_{j=1}^{J}\mu_r^2 - 2(\phi_j)_r\mu_r\right\}\right)\\
&= \exp\left(-\frac{1}{2}\left\{\mu_r^2\left(\frac{1}{v_r^2} + \frac{J}{\omega_r^2}\right) - 2\mu_r\left(\frac{m_{\mu,r}}{v_r^2} + \sum_{j=1}^{J}\frac{(\phi_j)_r}{\omega_r^2}\right)\right\}\right)\\
&\propto \exp\left(-\frac{1}{2}\left(\frac{1}{v_r^2} + \frac{J}{\omega_r^2}\right)\left\{\mu_r - \left(\frac{1}{v_r^2} + \frac{J}{\omega_r^2}\right)^{-1}\left(\frac{m_{\mu,r}}{v_r^2} + \sum_{j=1}^{J}\frac{(\phi_j)_r}{\omega_r^2}\right)\right\}^2\right),
\end{aligned}$$

which is proportional to the

$$\mathcal{N}\left(\left(\frac{1}{v_r^2} + \frac{J}{\omega_r^2}\right)^{-1}\left(\frac{m_{\mu,r}}{v_r^2} + \sum_{j=1}^{J}\frac{(\phi_j)_r}{\omega_r^2}\right), \left(\frac{1}{v_r^2} + \frac{J}{\omega_r^2}\right)^{-1}\right)$$

density, $r = 1, ..., p$.

In the case of correlations between the random effects, i.e. non-diagonal matrix Ω, one could take the Wishart prior distribution which is also conjugate to the normal likelihood, see, for example, Donnet et al. (2010). Choosing the inverse gamma distribution for the diagonal elements, we get the conditional posterior distribution

$$\omega_r^2|(\phi_1)_r, ..., (\phi_J)_r, \mu_r \sim \text{IG}\left(\alpha_{\omega,r} + \frac{J}{2}, \beta_{\omega,r} + \frac{1}{2}\sum_{j=1}^{J}((\phi_j)_r - \mu_r)^2\right),\ r = 1, ..., p.$$

The calculation is analogous to the posterior of γ^2 in the single series model.

The full conditional posterior of γ^2 is similar to the single series model given by

$$\gamma^2|Y_{(n\cdot,J)},\phi_1,...,\phi_J \sim \mathrm{IG}\left(a_\gamma+\sum_{j=1}^{J}\frac{n_j}{2}, b_\gamma+\frac{1}{2}\sum_{j=1}^{J}\sum_{i=1}^{n_j}\frac{(Y_{ij}-Y_{i-1,j}-b(\phi_j,t_{i-1,j},Y_{i-1,j})\Delta_{ij})^2}{\widetilde{s}^2(t_{i-1,j},Y_{i-1,j})\Delta_{ij}}\right).$$

After choosing starting values $\phi_{j0}^*,\ j=1,...,J$, for the MH step and $\mu_0^*,\ \Omega_0^*$ and γ_0^{2*} for the Gibbs sampler, we draw for $k=1,...,K$ from

$$\begin{aligned}\phi_{jk}^* &\sim p(\phi_j|Y_{(n_j,j)},\gamma_{k-1}^{2*},\mu_{k-1}^*,\Omega_{k-1}^*),\ j=1,...,J,\\ \mu_k^* &\sim p(\mu|\phi_{1k}^*,...,\phi_{Jk}^*,\Omega_{k-1}^*),\\ \Omega_k^* &\sim p(\Omega|\phi_{1k}^*,...,\phi_{Jk}^*,\mu_k^*),\\ \gamma_k^{2*} &\sim p(\gamma^2|Y_{(n\cdot,J)},\phi_{1k}^*,...,\phi_{Jk}^*).\end{aligned}$$

Prediction

As already mentioned, for within-sample prediction for one of the series, for example the last series J, the prediction for the hierarchical model is exactly the same as in the single series model. The transition density is the same as in (3.3) with $\theta=(\phi_J,\gamma^2)$ and the procedure starts with $Y_0^{*(k)}=Y_{n_J}, k=1,...,K$.

In the hierarchical model, prediction for a new series is different. We first have to calculate the predictive distribution of the unobserved random effect for the new series, say the $(J+1)$th series. It is given by

$$\begin{aligned}p(\phi_{J+1}|Y_{(n\cdot,J)}) &= \int p(\phi_{J+1}|\mu,\Omega)p(\mu,\Omega|Y_{(n\cdot,J)})\,d(\mu,\Omega) \qquad (3.5)\\ &\approx \frac{1}{K}\sum_{k=1}^{K}p(\phi_{J+1}|\mu_k^*,\Omega_k^*)\end{aligned}$$

with μ_k^* and Ω_k^* resulting from the Gibbs sampler, see, for example, Dion, Hermann and Samson (2016). Again, this is not a known distribution. Sampling from this density can be realized through one of the sampling methods in Sections 2.2.5 or 2.2.6, e.g. the inversion method. With the samples $\phi_{J+1,1}^*,...,\phi_{J+1,K}^*$ from the predictive distribution, we can compute a prediction for a future observation variable Y_{J+1}^* in t_{J+1}^* based on the Euler approximated variables with Algorithm 1 or 2 where $\theta_k^*=(\phi_{J+1,k}^*,\gamma_k^{2*})$ and the first step begins with $Y_0^{*(k)}=y_0(\phi_{J+1,k}^*)$ because we predict a new series.

3.3. Application

For the hierarchical model, a wide simulation study in the case of prediction for the further development of one of the series is presented in Hermann et al. (2016b) using Algorithm 1. In addition, six growth curves were applied to the data set of Virkler et al. (1979). The result was that the best fitting curves are one curve derived by the Paris-Erdogan equation (1.1) and the Weibull curve. We here restrict to the curve

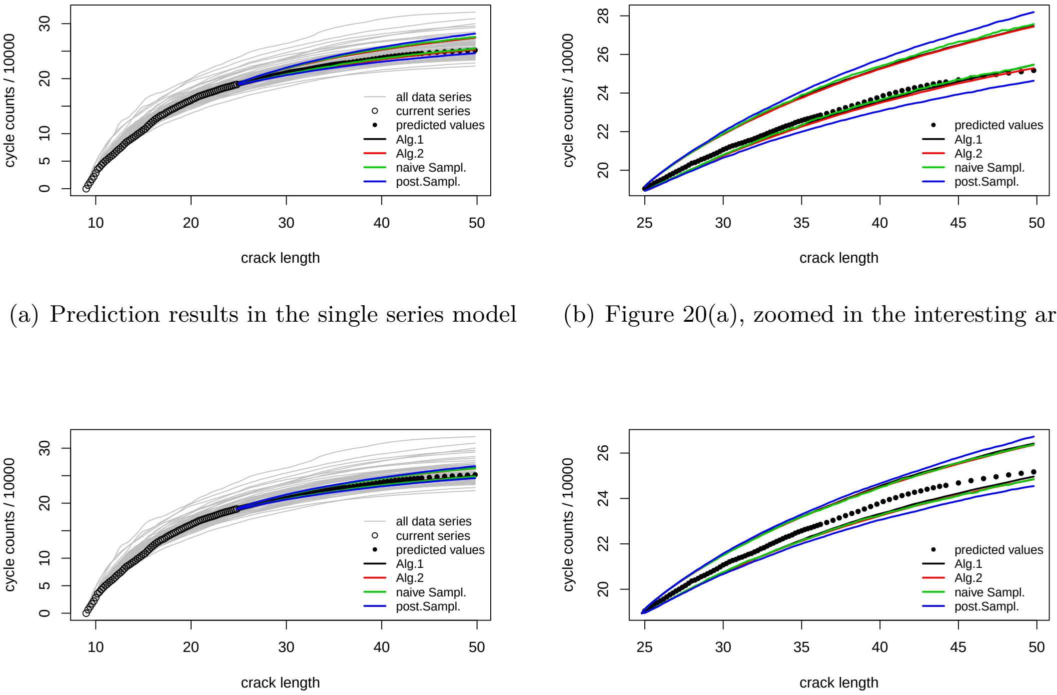
$$b(\phi, t, y) = \frac{\phi_2}{t}(\phi_1 - y)$$

with $\phi \in (0, \infty)^2$, derived from the Paris-Erdogan equation.

In the following, we will compare prediction results in the single series and the hierarchical model. For both, we assume a constant variance function $\widetilde{s}(t, y) = 1$. In Hermann et al. (2016b), the starting value for the location parameter ϕ is (25, 1.8). Here, we choose $\gamma_0^{2*} = 0.1$ and take the last of the series, i.e. the 68th, for a pre-estimation. The resulting posterior means are taken as starting points, for μ_0^* in the hierarchical and for ϕ_0^* in the single series model, whereby γ_0^* is the same for both models. In addition, for the hierarchical model, we set $\Omega_0^* = \frac{\mu_0^*}{100}$. These are starting parameters for the Gibbs sampler and also taken for the prior means. The prior parameters are chosen so that the standard deviation is equal to the mean. This choice is based on the results of the sensitivity analysis in Section 2.3, where estimation works well even with wrong prior mean. Therefore, we avoid a too informative prior information by setting the standard deviation equal to the mean.

To verify the prediction results, we truncate one of the series for estimation, here we arbitrarily choose the 10th, after the first 80 observations and predict the further development. This means, for the single series model, the posterior depends only on the first 80 observations of the 10th series. For the hierarchical model, beside the 68th series used for the pre-estimation, all remaining 66 series are used in addition. In Figure 20, we compare the different prediction methods for the two models. Firstly, for the single series model, Figure 20(a) shows the 67 series except the 10th in gray lines, the 10th series up to the 80th observation point is depicted by circles and the observations to be predicted in dots. Algorithms 1 to 3 lead to very similar prediction intervals. Algorithm 4, in blue lines, has larger intervals similar to the results in the previous section. In Figure 20(b), we see the interesting part without the gray lines in detail, where we can see that the curve is not well followed by the intervals. The aim of our modeling is first to find a model that describes the data well and second to calculate a precise prediction. Therefore, even if the blue lines include the observed values, this is not a satisfying result. In Figure 20(c), we see the results for the hierarchical model, again in gray lines the 67 series except the 10th, in circles the part of the 10th series used for estimation and in dots the part to be predicted. In detail, Figure 20(d) shows that the prediction intervals follow the curve much better than the single series model.

In Figure 21, we compare the two models, single series and hierarchical more closely for Algorithm 1. At first, the intervals are smaller in the hierarchical model than in the single series model. One reason can be the more informative prior distribution for the random effect through the mixture distribution than a fixed chosen prior distribution for a fixed effect in the single series model, which leads to a smaller variance estimation. Secondly, the prediction intervals follow the curve better. Here, again the reason might be, that the estimation of the random effect is more precise for the data series in the hierarchical than in the single series model, since information of the other series goes in. In summary, we can say that the hierarchical model performs much better, with the drawback of larger runtimes, since in every Gibbs



(a) Prediction results in the single series model

(b) Figure 20(a), zoomed in the interesting area

(c) Prediction results in the hierarchical model

(d) Figure 20(c), zoomed in the interesting area

Figure 20: 95% prediction intervals for the 10th series of the Virkler data with the four different prediction methods Algorithms 1 and 2, the naive sampling, Algorithm 3, and the posterior sampling, Algorithm 4; top: in the single series model, bottom: in the hierarchical model.

sampler iteration the likelihood in the MH step of the random effects for all series has to be calculated.

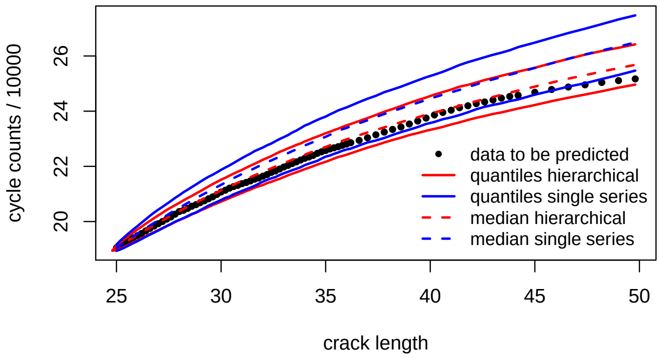


Figure 21: 95% prediction intervals for Algorithm 1 with the single series and the hierarchical model for the 10th series.

4. Hidden Diffusion Process

We here consider a stochastic model to describe a noisy observed diffusion. This can be suitable if two types of uncertainties are assumed. Firstly, the randomness of the crack growth as described by the diffusion process itself and secondly, a measuring error according to the experimental set up. In Gunkel et al. (2012), for example, crack growth is observed by taking photos of the test specimen at several time points during the experiment. Evaluating data from these images is a very sophisticated procedure and exhibits a high measuring variance, which is the same over time and, therefore, can be represented by an additive error term in a regression model, whereby the crack growth process itself follows a diffusion model.

In the case of a mixed model and in the case of an explicit solution of the diffusion process, an estimation procedure is described in Donnet et al. (2010). But diffusion processes often are defined by a SDE without a known explicit solution. In the following, we propose how to estimate the parameters of the hidden diffusion model based on the Euler approximation scheme.

In the described model, the diffusion process is latent and has to be estimated. Without an explicit solution that results in a known multivariate distribution, this step is not self-evident. Our model fits into the class of hidden Markov models and is closely related to the discrete-time state space models. The underlying problem of estimating a high-dimensional latent variable is well-known in the literature. Several filtering approaches are available, see, for example, Liu and Chen (1998); Doucet et al. (2000); Del Moral et al. (2006); Carvalho et al. (2010); Barber et al. (2011). In the following, we will shortly present the idea of Andrieu et al. (2010) and implement the particle Gibbs sampler introduced in their work. The following results of this section are ongoing work and are not published yet.

Analogously to the previous section, we will start with the single series model and extend it afterwards to a hierarchical model.

4.1. Single Series Model

Similar to the previous section, we start with a model for one data series

$$\begin{aligned} Z_i &= Y_{t_i} + \epsilon_i, \\ dY_t &= b(\phi, t, Y_t)\, dt + s(\gamma, t, Y_t)\, dW_t,\ Y_0 = y_0(\phi), \\ \epsilon_i &\sim \mathcal{N}(0, \sigma^2) \text{ i.i.d.},\ i = 0, ..., n. \end{aligned} \tag{4.1}$$

As mentioned, we will approximate the hidden diffusion process with the Euler approximation in

(2.2) with $h(\eta, t, y) = 0$. We assume the following Bayes model

$$\begin{aligned} Z_i|Y_i, \sigma^2 &\sim \mathcal{N}(Y_i, \sigma^2), \\ Y_i|Y_{i-1}, \phi, \gamma^2 &\sim \mathcal{N}(Y_{i-1} + b(\phi, t_{i-1}, Y_{i-1})\,\Delta_i, s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i),\ i = 1, ..., n, \\ \phi|m_\phi, V_\phi &\sim \mathcal{N}(m_\phi, V_\phi), \quad V_\phi = \mathrm{diag}(v_1^2, ..., v_p^2), \\ \gamma^2|a_\gamma, b_\gamma &\sim \mathrm{IG}(a_\gamma, b_\gamma), \\ \sigma^2|a_\sigma, b_\sigma &\sim \mathrm{IG}(a_\sigma, b_\sigma). \end{aligned} \tag{4.2}$$

We pool the model parameters with $\theta = (\phi, \gamma^2, \sigma^2)$ and for simplification introduce the additional short notation for the vector $Z_{(n)} = (Z_0, ..., Z_n)$.

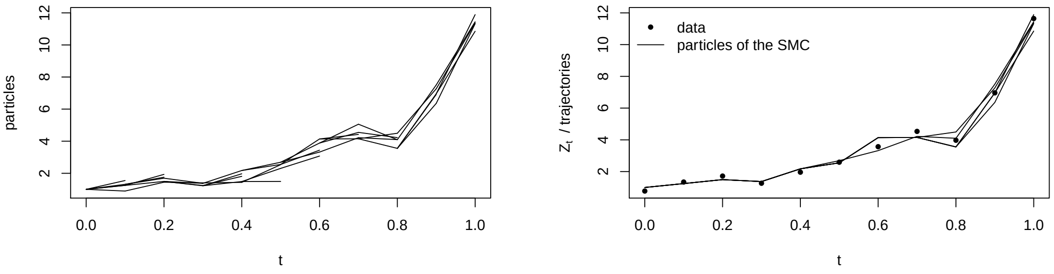
Estimation

In the paper of Donnet et al. (2010) the estimation procedure is described for the special case of the Gompertz process, i.e. $b(\phi, t, y) = BCe^{-Ct}y,\ \phi = (A, B, C)$ and $s(\gamma, t, y) = \gamma y$, because in this case there exists a solution of the process that leads to a multivariate normal likelihood for the vector of logarithmic observations. But the question arises if there is a possibility to develop a procedure based the SDE without explicit solution.

The discretized model (4.2) is also known as state-space model in the literature and several filtering approaches are available. We here present the idea of Andrieu et al. (2010), called particle Gibbs sampler, for the special case of our model. This procedure works equal to a usual Gibbs sampler explained in Section 2.2.2, where one step is sampling of the latent variable $Y_{(n)}$ to approximate its density $p(Y_{(n)}|Z_{(n)}, \theta)$. This will be done by sequentially approximating $p(Y_{(i)}|Z_{(i)}, \theta), i = 1, ..., n$, with L particles $Y_1^l, ..., Y_i^l, l = 1, ..., L$, which is called sequential Monte Carlo (SMC). The procedure is an extension of the sequential importance sampling, see Liu and Chen (1998); Doucet et al. (2000); Del Moral et al. (2006). Classically, Y_i^l is drawn iteratively from importance, or proposal, density $p(Y_i|Y_{i-1}, Z_i, \theta)$, which is the optimal one, see Liu and Chen (1998); Doucet et al. (2000), and importance weights $W_i^l, l = 1, ..., L, i = 1, ..., n$, are calculated. This can lead to a poor approximation of $p(Y_{(n)}|Z_{(n)}, \theta)$ if many of the weights are close to 0. In the sequential Monte Carlo algorithm, in each step, an index $A_i^l \sim p(\cdot) = \sum_{l=1}^L W_i^l \mathbb{1}_l(\cdot)$ is drawn and $Y_i^{A_i^l}$ is taken as the ancestor of Y_{i+1}^l. This means, the trajectories with higher probability are proceeded more often than less probable trajectories. An illustration is given in Figure 22 in a toy example with $L = 5$ particles. In Figure 22(b), the dots mark the example observations, while Figure 22(a) shows the particles, that are drawn during the algorithm. One can see, that some trajectories end, when they are less probable than others. In these points, two or more paths start at a more probable trajectory. In Figure 22(b), the resulting trajectories can be seen, which are estimations of the latent $Y_{(n)}$.

In detail, it is

$$p(Y_i|Y_{i-1}, Z_i, \theta) \propto p(Y_i|Y_{i-1}, \phi, \gamma^2) \cdot p(Z_i|Y_i, \sigma^2).$$



(a) Drawn particles

(b) Resulting trajectories

Figure 22: Example result of the sequential Monte Carlo algorithm, (a): the particles that are drawn during the algorithm, (b): the resulting trajectories.

Both are normal densities, which leads to

$$Y_i|Y_{i-1}, Z_i, \theta \sim$$

$$\mathcal{N}\left(\left(\frac{1}{s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i} + \frac{1}{\sigma^2}\right)^{-1}\left(\frac{Y_{i-1} + b(\phi, t_{i-1}, Y_{i-1})\,\Delta_i}{s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i} + \frac{Z_i}{\sigma^2}\right),\right.$$

$$\left.\left(\frac{1}{s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i} + \frac{1}{\sigma^2}\right)^{-1}\right).$$

This will be the proposal distribution for the increments.

For notation simplicity, we define $p_{1:L}(k|p_1, ..., p_L) = \sum_{l=1}^{L} p_l \mathbb{1}\{l = k\}$. The SMC algorithm for fixed $\theta = (\phi, \gamma^2, \sigma^2)$ is given as follows.

Algorithm 5 (sequential Monte Carlo (SMC))
To draw L particles, run the following steps:

```
$i = 0$ : Set $Y_0^l = y_0(\phi)$ and $W_0^l = \frac{1}{L}$, $W_0 = (W_0^1, ..., W_0^L)$.
$i = 1, ..., n$ :
    (i) Draw $A_{i-1}^l \sim p_{1:L}(\cdot|W_{i-1}),\ l = 1, ..., L,$
    (ii) set $Y_j^l = Y_j^{A_{i-1}^l},\ j = 1, ..., i-1, l = 1, ..., L,$
    (iii) draw $Y_i^l \sim p(Y_i|Y_{i-1}^l, Z_i, \theta),\ l = 1, ..., L,$
    (iv) set $w_i^l = \frac{p(Y_i^l|Y_{i-1}^l, \phi, \gamma^2) p(Z_i|Y_i^l, \sigma^2)}{p(Y_i^l|Y_{i-1}^l, Z_i, \theta)} = p(Z_i|Y_{i-1}^l, \theta),\ l = 1, ..., L,$
    (v) set $W_i^l = \frac{w_i^l}{\sum_{j=1}^L w_i^j},\ l = 1, ..., L,$ and $W_i = (W_i^1, ..., W_i^L)$.
```

For further details and theoretical background, see Andrieu et al. (2009) and Andrieu et al. (2010).

Point *(iv)* can be seen by the calculation for the denominator

$$\begin{aligned}p(Y_i|Y_{i-1},Z_i,\theta) =&\frac{p(Y_i|Y_{i-1},\theta)p(Z_i|Y_i,Y_{i-1},\theta)}{p(Z_i|Y_{i-1},\theta)}\\ =&\frac{p(Y_i|Y_{i-1},\phi,\gamma^2)p(Z_i|Y_i,\sigma^2)}{p(Z_i|Y_{i-1},\theta)}.\end{aligned}$$

To have one sample for the rest of the Gibbs sampler, we draw $j \sim p_{1:L}(\cdot|W_n)$ and set in the kth iteration $Y_{(n)}^{(k)} = (Y_0^j, ..., Y_n^j)$.

Algorithm 5 is extended by a conditional sequential Monte Carlo (CSMC). This means, in each of the iterations of the Gibbs sampler, the particle of the last iteration step $Y_{(n)}^{(k)}$ is fixed for the sequential Monte Carlo step and surely one of the particles. For reasons of clarity and comprehension, the extensions are explained in the Appendix B.

In the following, we present the full conditional posteriors of the parameters ϕ, γ^2 and σ^2 conditional on $Y_{(n)}$. The full conditional posterior distribution of γ^2 is exactly the same as in Section 3. For ϕ, a small difference in the likelihood appears because of the starting point $y_0(\phi)$. The posterior does not only depend on the likelihood of the $Y_{(n)}$, but also on the starting point Z_0 as can be seen by

$$\begin{aligned}p(\phi|Z_{(n)},Y_{(n)},\gamma^2,\sigma^2) &\propto p(\phi|Y_{(n)},\gamma^2,\sigma^2)\cdot p(Z_{(n)}|\phi,Y_{(n)},\gamma^2,\sigma^2)\\ &= p(\phi|Y_{(n)},\gamma^2)\cdot p(Z_0|y_0(\phi),\sigma^2)\cdot\prod_{i=1}^n p(Z_i|Y_i,\sigma^2)\\ &\propto p(\phi|Y_{(n)},\gamma^2)\cdot p(Z_0|\phi,\sigma^2),\end{aligned}$$

where $p(\phi|Y_{(n)},\gamma^2)$ is proportional to the likelihood of $Y_{(n)}$ in (3.2) multiplied to the normal prior density and $p(Z_0|\phi,\sigma^2)$ is the density of the $\mathcal{N}(y_0(\phi),\sigma^2)$ distribution.

Finally, with the prior $\sigma^2 \sim \mathrm{IG}(a_\sigma, b_\sigma)$ we get the posterior

$$\sigma^2|Z_{(n)},Y_{(n)} \sim \mathrm{IG}\left(a_\sigma+\frac{n+1}{2}, b_\sigma+\frac{1}{2}\sum_{i=0}^n(Z_i-Y_i)^2\right).$$

Choose starting parameters $\theta_0^* = (\phi_0^*, \gamma_0^{2*}, \sigma_0^{2*})$ and draw starting latent variable $Y_{(n)}^{(0)}$ with the ordinary SMC algorithm. Afterwards, conduct the particle Gibbs sampler for $k = 1, ..., K$ through

$$\begin{aligned}\phi_k^* &\sim p(\phi|Z_{(n)},Y_{(n)}^{(k-1)},\gamma_{k-1}^{2*},\sigma_{k-1}^{2*}),\\ Y_{(n)}^{(k)} &\sim p(Y_{(n)}|Z_{(n)},\phi_k^*,\gamma_{k-1}^{2*},\sigma_{k-1}^{2*}),\\ \gamma_k^{2*} &\sim p(\gamma^2|Y_{(n)}^{(k)},\phi_k^*),\\ \sigma_k^{2*} &\sim p(\sigma^2|Z_{(n)},Y_{(n)}^{(k)}),\end{aligned}$$

where $Y_{(n)}^{(k)}$ is drawn by the CSMC for $k = 1, ..., K$.

The resulting samples $\theta_k^* = (\phi_k^*, \gamma_k^{2*}, \sigma_k^{2*}), k = 1, ..., K$, can be seen as samples from the posterior $p(\theta|Z_{(n)})$.

Prediction

To calculate a predictive distribution for Z^* in t^*, we first simulate from the predictive distribution of Y^* in t^* with one of the algorithms in Sections 2.2.5 or 2.2.6. The transition density is the same as in Section 3, namely the density of the normal distribution in (3.3). Algorithm 1 or 2 is then conducted with $\theta = (\phi, \gamma^2)$ and starting values $Y_0^{*(k)} = Y_n^{(k)}, k = 1, ..., K$, stemming from the particle Gibbs sampler in the case of within-sample prediction, and $Y_0^{*(k)} = y_0(\phi_k^*), k = 1, ..., K$, in the case of new- sample prediction. This yields samples $Y_1^{*(k)}, ..., Y_M^{*(k)}$ in time points $\tau_1, ..., \tau_M = t^*, k = 1, ..., K$ for the latent diffusion variables.

For the prediction of Z^* we use the samples $Y_M^{*(k)} \sim p(Y^*|Z_{(n)}), k = 1, ..., K$, and the samples $\sigma_k^{2*}, k = 1, ..., K$, from the Gibbs sampler and calculate the distribution

$$\begin{aligned} p(Z^*|Z_{(n)}) &= \int p(Z^*|Y^*, \sigma^2) \cdot p(Y^*, \sigma^2|Z_{(n)})\, d(Y^*, \sigma^2) \\ &\approx \frac{1}{K} \sum_{k=1}^{K} p(Z^*|Y_M^{*(k)}, \sigma_k^{2*}). \end{aligned} \tag{4.3}$$

Since Y^* and σ^2 are independent, this approximation is straightforward. We will see a simulation study and the application to the data set of Virkler et al. (1979) in Section 4.3.

4.2. Hierarchical Model

We here present an extension of the hierarchical diffusion model presented in Section 3 and add an i.i.d. regression error. Of course, several extensions are possible. As mentioned above, other mixture distributions than the normal one could be of interest. Here, in addition other variance structures for the regression error can be suitable. For example, a time-dependent variance function for the regression error variable could be chosen. But this is future work and is beyond the scope of this thesis.

We define the Euler approximated Bayes model

$$\begin{aligned} Z_{ij} &= Y_{ij} + \epsilon_{ij}, \\ Y_{ij}|Y_{i-1,j}, \phi_j, \gamma^2 &\sim \mathcal{N}(Y_{i-1,j} + b(\phi_j, t_{i-1,j}, Y_{i-1,j})\, \Delta_{ij}, s^2(\gamma, t_{i-1,j}, Y_{i-1,j})\Delta_{ij}), \\ \phi_j|\mu, \Omega &\sim \mathcal{N}(\mu, \Omega), \quad \Omega = \mathrm{diag}(\omega_1^2, ..., \omega_p^2), \\ \epsilon_{ij} &\sim \mathcal{N}(0, \sigma^2), \ i = 1, ..., n_j, j = 1, ..., J, \\ \mu|m_\mu, V_\mu &\sim \mathcal{N}(m_\mu, V_\mu), \quad V_\mu = \mathrm{diag}(v_1^2, ..., v_p^2), \\ \omega_r^2|a_{\omega,r}, b_{\omega,r} &\sim \mathrm{IG}(a_{\omega,r}, b_{\omega,r}), \ r = 1, ..., p, \\ \gamma^2|a_\gamma, b_\gamma &\sim \mathrm{IG}(a_\gamma, b_\gamma), \\ \sigma^2|a_\sigma, b_\sigma &\sim \mathrm{IG}(a_\sigma, b_\sigma) \end{aligned} \tag{4.4}$$

with $Y_{0j} = y_0(\phi_j), j = 1, ..., J$. Denote with $Z_{(n_j,j)}$ the jth observation vector and $Z_{(n\cdot,J)}$ all the observations $\{Z_{ij}\}_{i=1,...,n_j,j=1,...,J}$.

Estimation

Analogously to the single series model, the estimation of the latent variables $Y_{(n_1,1)}, ..., Y_{(n_J,J)}$ is the crucial estimation step and will be conducted by the CSMC algorithm in each of the particle Gibbs sampler iterations.

Based on the estimation of these latent variables, estimation of $\phi_1, ..., \phi_J, \mu, \Omega$ and γ^2 are the same as for the hierarchical diffusion model in Section 3. The full conditional posterior of σ^2 is similar to the single series model given by

$$\sigma^2 | Z_{(n\cdot,J)}, Y_{(n\cdot,J)} \sim \text{IG}\left(a_\sigma + \sum_{j=1}^{J} \frac{n_j+1}{2}, b_\sigma + \frac{1}{2}\sum_{j=1}^{J}\sum_{i=0}^{n_j}(Z_{ij} - Y_{ij})^2\right).$$

Prediction

Analogously to Section 3, we here restrict to the case of prediction for a new series, since prediction for the further development of one observed series is exactly the same as for the single series model.

Prediction for a new series $J+1$ is similar to the hierarchical diffusion model in Section 3. Analogously to the predictive distribution of ϕ_{J+1} in (3.5), it is

$$\begin{aligned} p(\phi_{J+1}|Z_{(n\cdot,J)}) &= \int p(\phi_{J+1}|\mu, \Omega) p(\mu, \Omega | Z_{(n\cdot,J)})\, d(\mu, \Omega) \\ &\approx \frac{1}{K}\sum_{k=1}^{K} p(\phi_{J+1}|\mu_k^*, \Omega_k^*) \end{aligned}$$

with μ_k^* and Ω_k^* resulting from the particle Gibbs sampler.

With the samples $\phi_{J+1,1}^*, ..., \phi_{J+1,K}^*$ from the predictive distribution, we can, similar to Section 3, compute a prediction for a future process variable Y_{J+1}^* based on the Euler approximated variables with Algorithm 1 or 2, where $\theta_k^* = (\phi_{J+1,k}^*, \gamma_k^{2*})$ and $Y_0^{*(k)} = y_0(\phi_{J+1,k}^*)$, $k = 1, ..., K$, since we predict a new series.

For the prediction of Z_{J+1}^*, we use the predicted variables $Y_M^{*(k)}$, $k = 1, ..., K$, and the samples σ_k^{2*}, $k = 1, ..., K$, from the particle Gibbs sampler and calculate the pointwise distribution equal to (4.3).

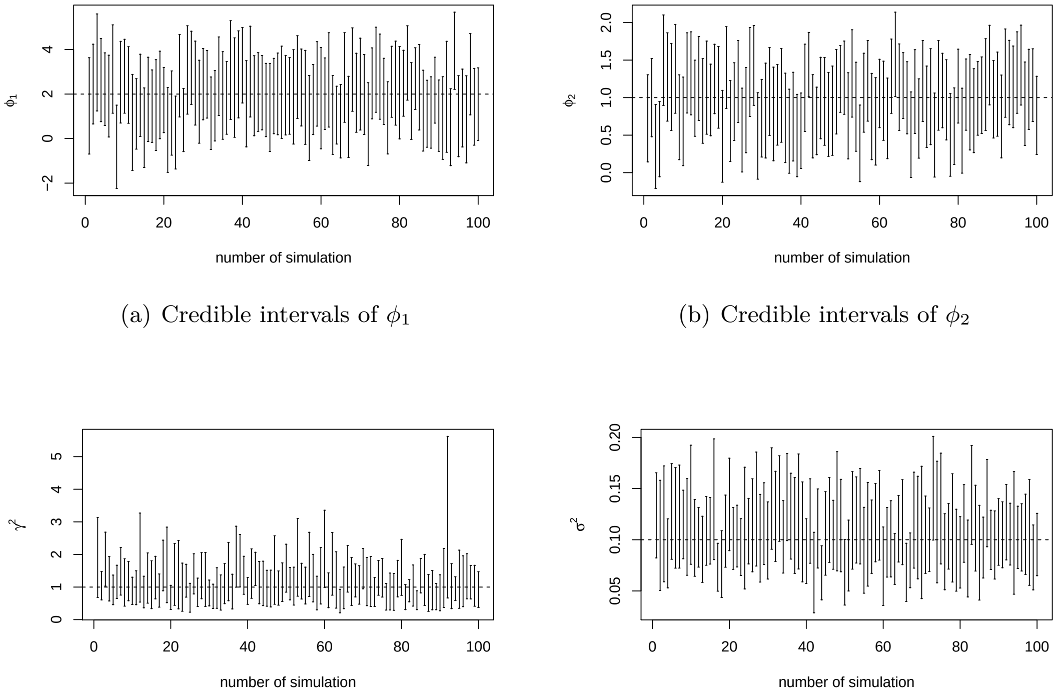
4.3. Simulation Study and Application

In this section, we will run a simulation study for the single series model in Section 4.1 and show results for one simulation example for the hierarchical model of the last section. Afterwards, both models are applied to the data set of Virkler et al. (1979) and, in addition, the hierarchical model will be applied to the wear data set of Hermann and Ruggeri (2016).

We first run a simulation study to investigate the quality of the particle Gibbs sampler. We again use Example 1. In addition, we choose an error variance $\sigma^2 = 0.1$ and starting point

$y_0(\phi) = \phi_2$ to be estimated. This leads to $\phi = (2, 1)$ for a comparable process. We simulate trajectories in $t_0 = 0, t_1 = 0.01, ..., t_{100} = 1$ for the estimation and will make predictions for the further development in $\tau_1 = 1.01, ..., \tau_M = 1.5, M = 50$. Since we have large data series with $n = 100$ for the estimation in comparison to the study in Section 2.4, we expect a small estimation uncertainty and that the prediction with the naive sampling can compete. We simulate 500 series for estimation and prediction. We here start the Gibbs sampler with the chosen values, which are also taken as prior mean. The prior parameters are chosen so that the standard deviation is equal to the mean. It will be future work to try several diffused starting and prior values. The number of particles for the SMC algorithm is here chosen to $L = 100$. It will also be future work to investigate, how many particles are at least necessary for a most effective algorithm. There is a trade-off between running time and quality of the algorithm. Many particles lead to a high quality of the algorithm but take a long time and vise versa.

We iterate the particle Gibbs sampler 101 000 times, and less a burn-in phase of 1000 and a thinning rate of 10 we get $K = 10\,000$ samples from the posterior. In Figure 23, we see the 95% credible intervals of the parameters for the first 100 series. The coverage rates of all 500 credible intervals can be seen in Table 4, which leads to the assumption that the algorithm works well.



(a) Credible intervals of ϕ_1

(b) Credible intervals of ϕ_2

(c) Credible intervals of γ^2

(d) Credible intervals of σ^2

Figure 23: Estimation results in the simulation study for the single series model based on the particle Gibbs sampler: 95% credible intervals of the parameters for the first 100 series, horizontal line: chosen value.

Table 4: Coverage rates of the 95% credible intervals of the parameters in the hidden diffusion model.

ϕ_1	ϕ_2	γ^2	σ^2
0.982	0.942	0.97	0.96

In Figure 24, we see for one example of the 500 series the result of the CSMC algorithm inside the particle Gibbs sampler. The green lines mark the estimated latent diffusion processes, where the dots mark the simulated data points used for estimation. For prediction of the further development, the last trajectory points are used as starting point for the prediction algorithm.

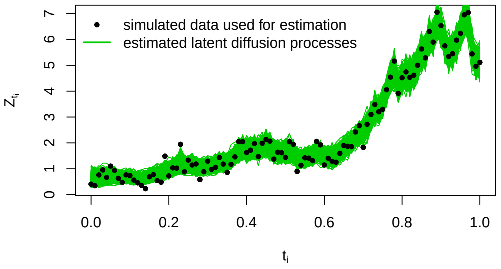
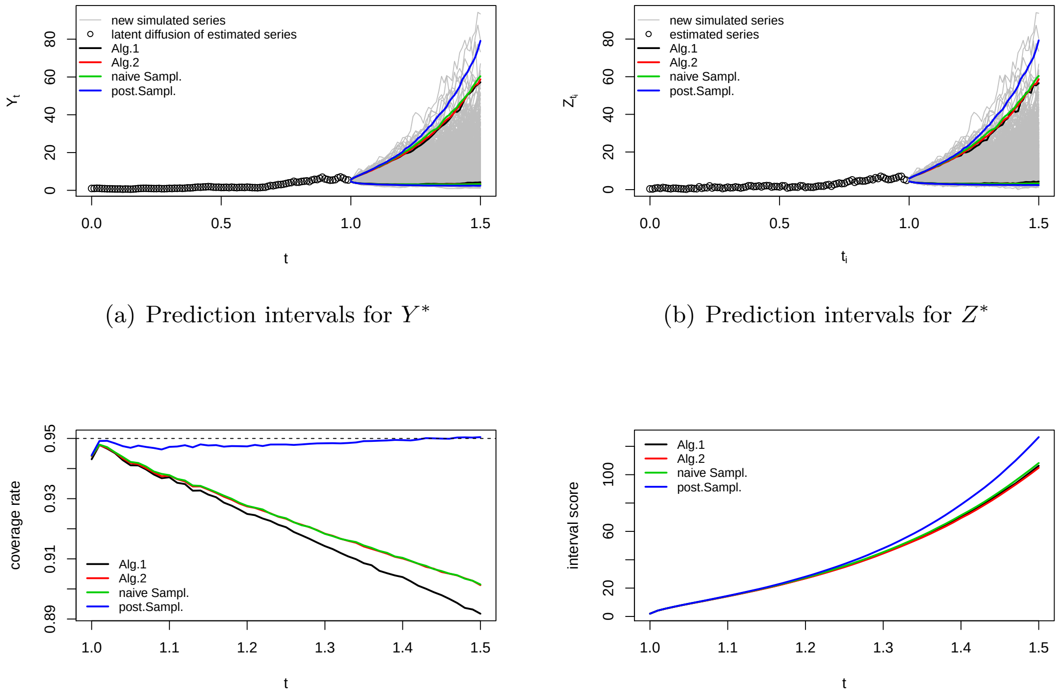


Figure 24: Result of the CSMC algorithm for one example series of the simulation study for the single series model; in green: the estimated latent variables $Y_{(n)}^{(k)}, k = 1, ..., K =$ 10 000 from the particle Gibbs sampler.

In the following we have a look at the prediction results. For each of the 500 simulated series and corresponding prediction intervals, we simulate 1000 new series starting each with the last simulated point for Y_{100}.

One example can be seen in Figure 25(a) for the latent diffusion process and in Figure 25(b) for the observation variables. Again, the posterior sampling leads to the largest intervals, whereas the difference between Algorithms 1 to 3 is small. In addition, because of the small size of σ^2, the difference between the two figures for the latent and the observed variables is small. Therefore, we concentrate on the observation variables.

For the 500 simulated series, we have 500 corresponding prediction intervals leading to a coverage rate and averaged interval score for the 1000 simulated new series each. Both can be seen in Figures 25(c) and 25(d). At first, we see that the posterior sampling has, with large intervals, a high coverage rate close to the level, but nevertheless the highest interval scores. At second, the results of the naive sampling and the trajectory-wise Algorithm 2 are close to each other with a slightly higher interval score for the naive sampling. In comparison to the simulation studies in Section 2.4, with $n = 100$ we have a large amount of observations, which leads to a small estimation uncertainty. Therefore, both algorithms work similar. At third, the coverage rates of Algorithm 1 are the smallest ones with 0.89 at the end, but the interval scores lie between the lines of Algorithm 2 and the naive sampling, because of the small, i.e. precise, intervals.



(a) Prediction intervals for Y^*

(b) Prediction intervals for Z^*

(c) Coverage rates of the prediction intervals for observation variable Z^*

(d) Average interval scores of the prediction intervals for observation variable Z^*

Figure 25: Prediction results for one series simulated from Example 1: Comparison of the prediction methods in Algorithms 1 and 2, the naive sampling, Algorithm 3, and the posterior sampling, Algorithm 4. (a): prediction intervals for the latent diffusion variable Y^*, (b): Prediction intervals for the observation variable Z^*, (c): amount of prediction intervals covering the observation variables of the 1000 new simulated series, (d) the corresponding interval scores, both in time points $\tau_1 = 1.01, ..., \tau_{50} = t^* = 1.5$.

As already mentioned, the algorithm is very slow. Therefore, for the hierarchical hidden diffusion model, we only run one simulation to show some results. It will be future work to implement the algorithm in another language like C to make it faster. The parameters are similar to above except for $\mu = (2, 1)$ and $\Omega = (1, 0.04)$. We simulate $J = 50$ samples for the random effect and, therefore, J series each with one random effect in $t_0 = 0, t_1 = 0.01, ..., t_{100} = 1$. For the estimation, we truncate the Jth series after the first half and predict the second half afterwards.

The starting values and prior means are chosen equal to the values for the simulation and the prior parameters are set so that standard deviation is equal to the mean. In Figure 26, we see the resulting chains with 11 000 iterations. The horizontal red lines mark the chosen values. All chains look stationary with a good mixing and the posterior is close to the chosen value, which is marked in red.

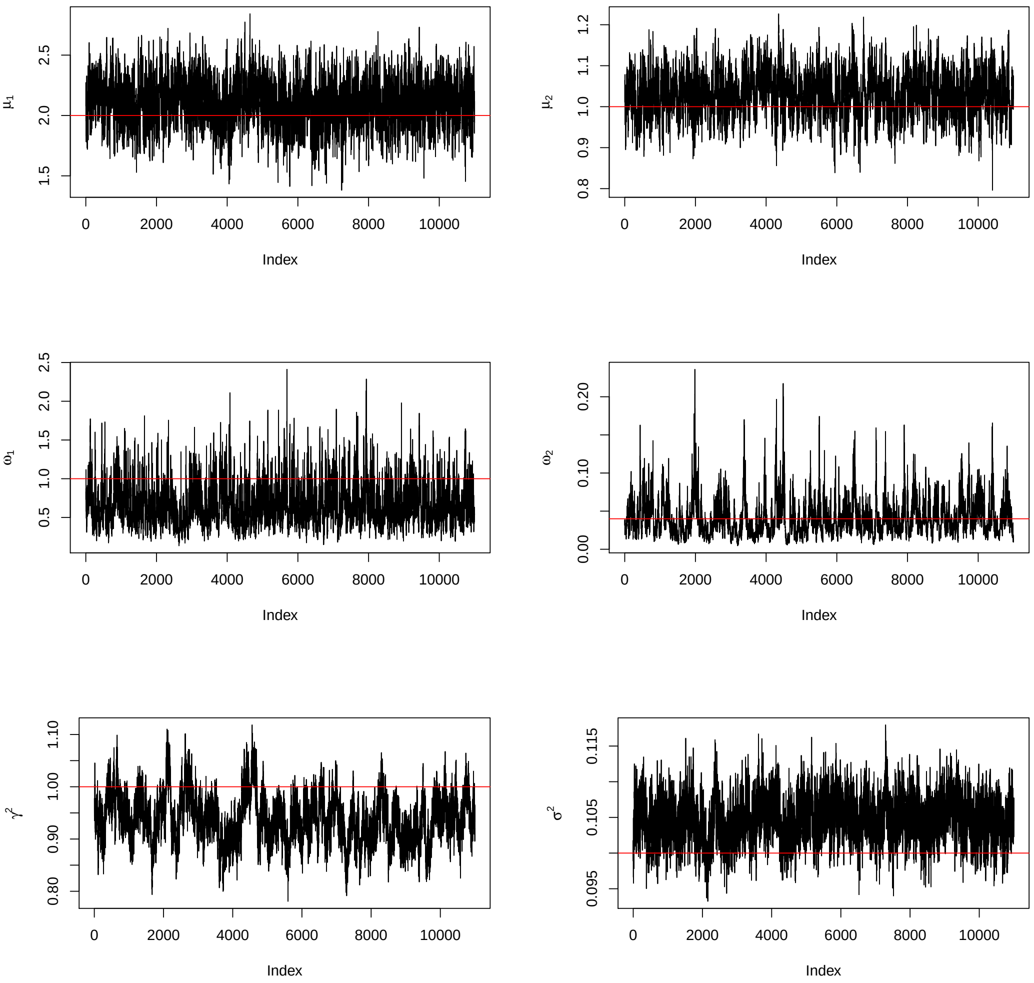


Figure 26: Markov chains of the parameters μ, Ω, γ^2 and σ^2 from the particle Gibbs sampler in the hierarchical hidden diffusion model for one simulated data set. In red: the chosen values for simulation.

We take a burn-in phase of 1000 and a thinning rate of 10 and run with the remaining samples the prediction algorithm. We here restrict to Algorithm 1, since the differences between the algorithms were investigated in the study before. The prediction results with Algorithm 1 for the Jth series truncated for estimation can be seen in Figure 27. On the left side the underlying latent diffusion variables are displayed and on the right side the observation variables.

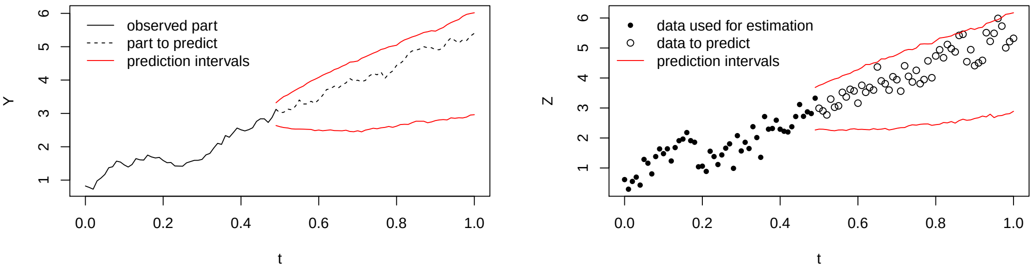


Figure 27: Prediction results for the Jth series in the hierarchical simulated data set: prediction for the latent variables (left) and the future observations (right).

Application to the Virkler data

We have already seen the data set of Virkler et al. (1979) in the model motivation and application of the diffusion model in Section 3.3. We here fit the hidden diffusion model to the data. For good starting parameters, we use the values from Section 3.3 of the single series model and $\sigma_0^{2*} = 0.1$. We first estimate the single series hidden diffusion model for the last, i.e. the 68th, series of the Virkler data. The resulting point estimates are taken for the starting values for the further inference, for the single series and the hierarchical model. In addition, we set $\Omega_0^* = \frac{\mu_0^*}{100}$.

Since the estimation algorithm is computationally expensive, we here take only the first 10 of the series for the hierarchical model into account. For both models, the tenth series is truncated after the 80th observation to evaluate the prediction afterwards. We compare the predictions with the single series and the hierarchical model for this tenth series. We iterate for both models the particle Gibbs sampler 100 000 times, and take a burn-in phase of 25 000 and a thinning rate of 10. The resulting samples in the hierarchical model for γ^2 and σ^2 can be seen in Figure 28. Both chains converge and the locations are very small.

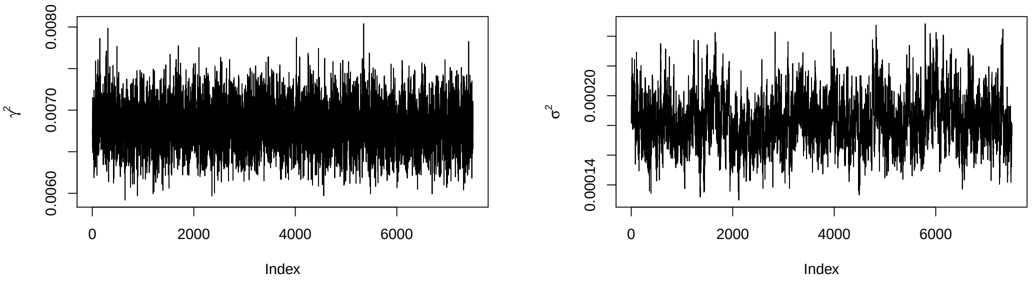


Figure 28: Results for the Virkler data in the hierarchical model (4.4): Markov chains of γ^2 (left) and σ^2 (right), reduced by the burn-in phase and the thinning rate.

In Figure 29, we see in green the estimated latent variables. The variance γ^2 is that small that the latent diffusion looks like a solid line. In addition, the difference between the observed points and the estimated diffusions is very small, therefore, the small location of the posterior of σ^2 is not surprising.

In Figure 30, we see the prediction results for the tenth Virkler series with the hierarchical and the single series model, both with Algorithm 1. The point predictions, i.e. the median of the predictive distribution, are marked with the dotted lines. Similar to the results in Section 3.3, the intervals with the hierarchical model are smaller and follow the curve better.

In addition, it is interesting to compare the results with the ones from the previous section. In Figure 31, we compare each the hidden diffusion model with the observed diffusion model from Section 3, in Figure 31(a) for the single series model and in Figure 31(b) for the hierarchical model. In the single series model, there is no recognizable difference, whereas for the hierarchical model, the hidden diffusion model seems a bit more appropriate, even if there are informations of only 9 additional series used instead of 66 in the diffusion model.

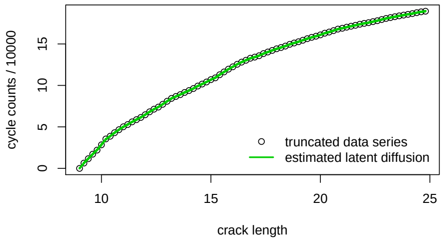


Figure 29: Estimation of the latent diffusion variables of the 10th truncated Virkler series in the hierarchical model (4.4).

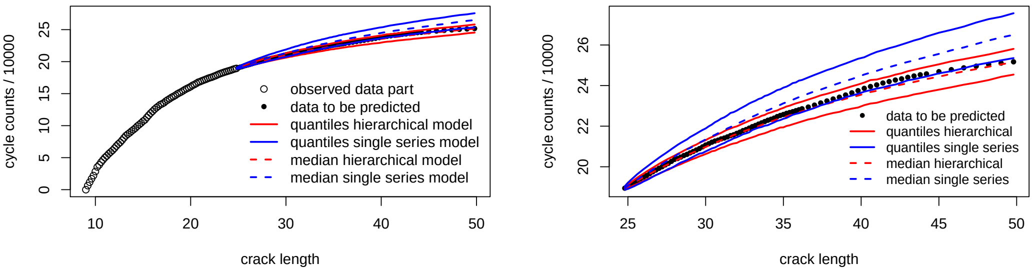
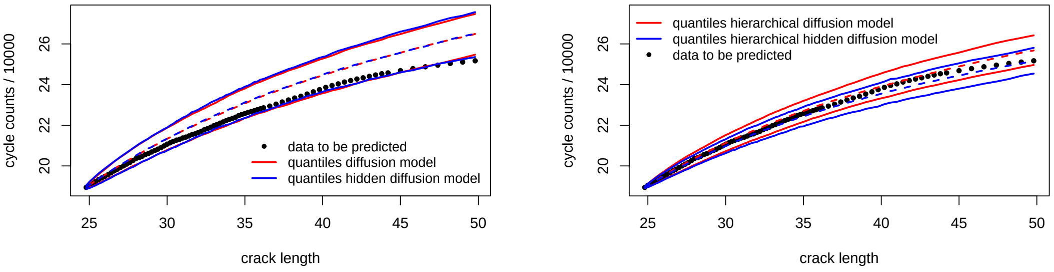


Figure 30: 95% prediction intervals and the pointwise median for the 10th series of the Virkler data set based on Algorithm 1, left: whole series, right: zoomed in the interesting area, in red: the hierarchical model, in blue: the single series model.



(a) Single series model

(b) Hierarchical model

Figure 31: Comparison of the hidden and the observed diffusion models. 95% prediction intervals and the pointwise median for the 10th series of the Virkler data set.

Application to wear degradation in cylinder liners

In the following, we will apply the model to the wear of cylinder liners in marine diesel engines. A detailed explanation of the data set can be found in Hermann and Ruggeri (2016). The data concern the wear process of liners protecting cylinders in marine diesel engines. The thickness of the liner walls are measured over time in 10^4 hours, starting at 100 millimeters. The whole data set can be seen in Figure 34 in black points, where we take the wear increment as observed variables, which is the same as 100 minus the thickness decrement. We have observations from 30 cylinder liners over time with non-equidistant time points and every series has one to four observations, beside the starting point, which is a sparse data set. Therefore, we restrict to a one-parameter drift function based on the Paris-Erdogan equation (1.1) with $m = 0$, namely $b(\phi, t, y) = \phi$ and use a constant variance function $\widetilde{s}(t, y) = 1$. All series start in $y_0(\phi) = 0$ since in $t_0 = 0$, no wear exists. We start the algorithm with $\mu_0^* = 0.8, \Omega_0^* = 0.01, \gamma^2 = 0.001$ and $\sigma^2 = 0.01$, which are taken as prior means as well. Again, the prior parameters are chosen so that the standard deviation is equal to the mean. The starting values are chosen manually by simulating the process in the time points of the data set and comparing the data to the simulated set.

In Figure 32, we see the 95% credible intervals of the random effects for the 30 series. They differ only slightly because of the sparse data series and the mixture distribution has much influence.

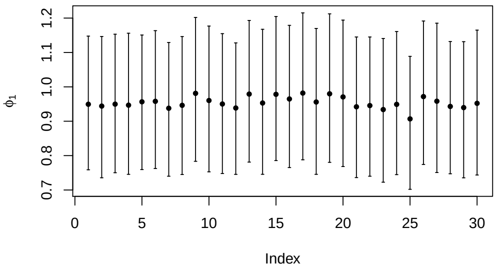


Figure 32: 95% credible intervals for the random effects for the wear data set.

In Figure 33, the chains of the parameters can be seen. We ran the Gibbs sampler with 101 000 iterations and took a burn-in phase of 11 000 samples and a thinning rate of 10. The four chains converge and have a good mixing.

In Figure 34, the prediction result for a new series with Algorithm 1 is displayed. We see in solid red lines the 95% prediction intervals and in dotted lines the pointwise median of the predictive distribution. One could imagine a saturation curve in the data, which the assumed model does not take into account. Nevertheless, the prediction intervals enclose the data except three points in the beginning. The same data set has been modeled in Hermann and Ruggeri (2016) with a jump diffusion model as will be introduced as Merton model in Section 6. We see in Figure 34(b) the comparison of the results to Figure 34(a). The lines are very similar, but the here presented model leads to slightly smaller intervals by including the same amount of data points.

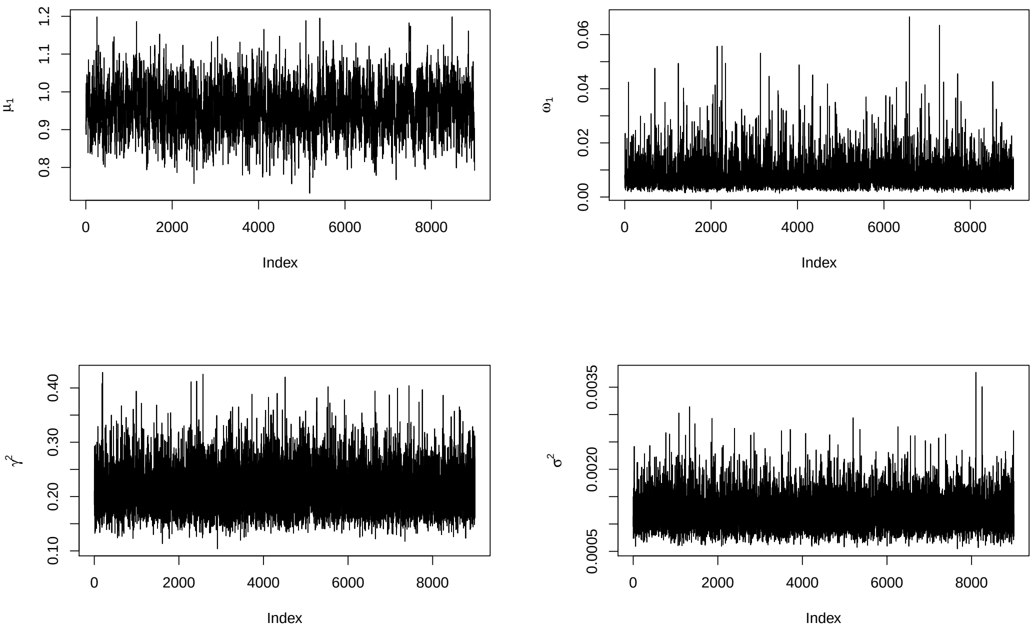
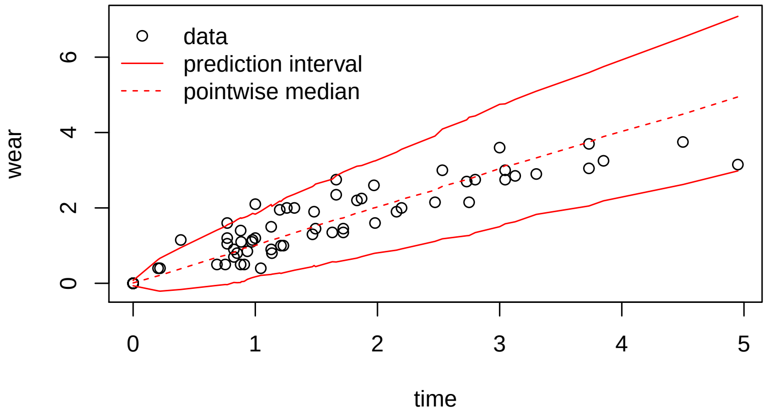
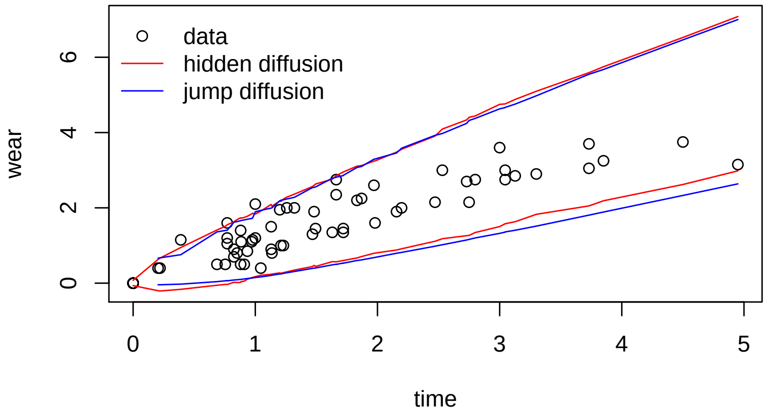


Figure 33: Markov chains of μ, Ω, γ^2 and σ^2, reduced by the burn-in phase and the thinning rate in the hierarchical hidden diffusion model (4.4) for the wear data set.

As described in detail in the work of Hermann and Ruggeri (2016), the jump diffusion model is nicely motivated by the intuition of tiny and large soot particles in the cylinder liners. The large soot particles are responsible for the most relevant wear increments, represented by the Poisson process in the jump diffusion. Instead of this heuristically motivated, sophisticated model, the hidden diffusion model presented here, is a bit more simple but lead to similar, maybe slightly better, results. From the computational point of view, both models are very complex with filtering of latent variables, the Poisson process variables in the jump diffusion model and the diffusion process in the hidden diffusion model.



(a) Prediction result with the hierarchical hidden diffusion model, in dotted lines: the pointwise median



(b) Comparison with results in Hermann and Ruggeri (2016) with a jump diffusion model

Figure 34: 95% prediction intervals for a new series of wear in cylinder liners with the hierarchical hidden diffusion model (4.4) (in red) and the jump diffusion process model (6.2) introduced in Section 6 (in blue).

Part II.
Bayesian Analysis and Prediction for Non-continuous Processes

In constructional engineering, material fatigue is a relevant topic of research because it is important to predict the lifetime of, for example, bridges. Experiments in this field, especially under low loadings, are very rare because they are very expensive and take several months. Maurer and Heeke (2010) carried out five experiments where prestressed concrete beams with initial cracks have been put under cyclic load. Recently, five new experiments were conducted, see Szugat et al. (2016). Each prestressed concrete beam contains 35 tension wires which break at random time points. Therefore, the resulting crack widths, which can be seen in Figure 35 for two of the experiments, exhibit jumps with increasing frequency that influence the crack growth process substantially. Structure-borne noise measurements during the experiments provide information concerning the break times of the tension wires which match the observed jumps in the crack width data.

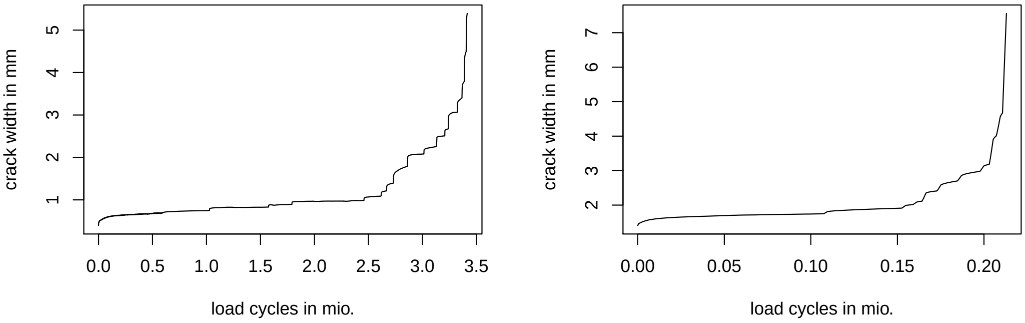


Figure 35: Crack width data resulting from the first two experiments of Maurer and Heeke (2010), left: 200 MPa, right: 455 MPa.

In the following, the wire breaking process will be modeled by a non-homogeneous Poisson process (NHPP). NHPPes are well-known in the literature. Meeker and Escobar (1998), for example, give a historical overview of the literature employing NHPPes in reliabilty theory, see pp. 413–420, and present maximum likelihood estimators for several intensity functions. Sobczyk and Spencer (1992) introduce cumulative jump models based on homogeneous and non-homogeneous Poisson processes to describe crack growth as a discontinuous random process.

We will see two approaches including the NHPP into a model for the crack width process. Firstly, in Section 6, the whole jump diffusion model in (2.1) will be presented. Secondly, in Section 7, a non-linear regression model based on a physically motivated function including the NHPP will be proposed.

5. Non-homogeneous Poisson Process

In this section, we consider the counting process representing the wire breaking process. A Bayesian approach for NHPPes can be found in Kuo and Yang (1996) with application in software reliability, in Yuan et al. (2009) for modeling pitting corrosion in steam generator tubes and in Pievatolo and Ruggeri (2004) for reliability of complex repairable systems. A good overview of the Bayesian analysis of NHPPes can be found in Ríos Insua et al. (2012).

5.1. Model Description, Inference and Prediction

The Poisson process is nested in (2.1) by setting $b(\phi, t, y) = 0$, $s(\gamma, t, y) = 0$, $h(\eta, t, y) = 1$ and $y_0 = 0$, whereby the differences $N_{t_i} - N_{t_{i-1}}, i = 1, ..., n$, are independent and Poisson distributed with parameter $\Lambda_\xi(t_i) - \Lambda_\xi(t_{i-1})$.

In Hermann et al. (2016a), a comparison of two different types of $\Lambda_\xi(t)$ can be found for the data set, one for polynomial and one for exponential growth:

$$\Lambda_\xi(t) = \left(\frac{t}{\beta}\right)^\alpha, \qquad \xi = (\alpha, \beta) \in [1, \infty) \times (0, \infty), \tag{5.1}$$

$$\Lambda_\xi(t) = \exp(at - b) - \exp(-b), \qquad \xi = (a, b) \in (0, \infty) \times \mathbb{R}, \tag{5.2}$$

where the first function (5.1) has the nice property that the derivative of $\Lambda_\xi(t)$ is the hazard rate of the Weibull distribution and the homogeneous case of the process is nested by setting $\alpha = 1$. This specification is also known as power law in the literature, see for example Yu et al. (2007). The second function (5.2) represents an exponential intensity rate, that takes up the idea of the Paris-Erdogan equation in (1.1) with the special case $m = 2$. This equation would lead to a function $f(t) = \widetilde{b} \exp(at) = \exp(at - b)$ with $\widetilde{b} = \exp(-b)$. This function has to be pulled down to zero in $t = 0$ which leads to the cumulative intensity $\exp(at - b) - \exp(-b)$. Of course, one could try an approach based on the general Paris-Erdogan equation with a third parameter m. Since we have four or five observed event times it is difficult to estimate more than two parameters. Therefore, we restrict to the two-parameter approach with $m = 2$.

Estimation

Note that the counting process $\{N_t, t \in [0, T]\}$ is stochastically completely defined by its occurrence times $0 < T_1 < ... < T_I \leq T$. In the following, we assume that the process is observed up to time T, i.e. $T = t_n$.

The first occurrence time has probability

$$P(T_1 > t|\xi) = P(N_t = 0|\xi) = e^{-\Lambda_\xi(t)},$$

which leads to the distribution function $F_{T_1}(t) = 1 - e^{-\Lambda_\xi(t)}$ with density $p(T_1|\xi) = \lambda_\xi(T_1) \cdot e^{-\Lambda_\xi(T_1)}$. For the second occurrence time, it is

$$P(T_2 > t|T_1, \xi) = P(N_t - N_{T_1} = 0|T_1, \xi) = e^{-(\Lambda_\xi(t) - \Lambda_\xi(T_1))}$$

and the resulting density $p(T_2|T_1,\xi) = \lambda_\xi(T_2) \cdot e^{-(\Lambda_\xi(T_2)-\Lambda_\xi(T_1))}$. For the general case, it is

$$p(T_i|T_{i-1},\xi) = \lambda_\xi(T_i) \cdot e^{-(\Lambda_\xi(T_i)-\Lambda_\xi(T_{i-1}))}, i = 1, ..., I, \tag{5.3}$$

with $T_0 = 0$ and $\Lambda_\xi(0) = 0$. Since we investigate the counting process on a compact interval $[0,T]$ we also have the information that T_I is the last occurrence up to time T, i.e. $P(N_T - N_{T_I} = 0|T_I,\xi) = e^{-(\Lambda_\xi(T)-\Lambda_\xi(T_I))}$. Altogether, we get the likelihood

$$\begin{aligned} p(N_{(n)}|\xi) &= p(T_1|\xi) \cdot \prod_{i=2}^{I} p(T_i|T_{i-1},\xi) \cdot P(N_T - N_{T_I} = 0|T_I,\xi) \qquad (5.4)\\ &= e^{-\Lambda_\xi(T_1)} \cdot \prod_{i=1}^{I} \lambda_\xi(T_i) \cdot e^{\sum_{i=2}^{I} -\Lambda_\xi(T_i)+\Lambda_\xi(T_{i-1})} \cdot e^{-\Lambda_\xi(T)+\Lambda_\xi(T_I)} \\ &= e^{-\Lambda_\xi(T)} \cdot \prod_{i=1}^{I} \lambda_\xi(T_i), \end{aligned}$$

see also Ríos Insua et al. (2012).

Based on this likelihood, there is no obvious conjugate prior distribution for the whole vector ξ so we will use the MH algorithm with $\theta = \xi$, see Section 2.2.1. In the case of the two intensity functions in (5.1) and (5.2), a proposal density with positive support is suitable. In the BaPreStoPro package, the lognormal density is implemented for that case. For other intensity functions with possibly negative parameters, the normal proposal density is implemented as well.

In Hermann et al. (2016a), a non-informative approach without any prior specification is used. In the data set of Maurer and Heeke (2010), we have a maximum of 15 observed jumps. Therefore, prior knowledge has to be justified since it will have large influence with few observations and a non-informative approach is very suitable.

Prediction

Based on the MH resulting samples $\xi_1^*, ..., \xi_K^* \sim p(\xi|N_{(n)})$, a predictive distribution can be approximated.

There are two possibilities. Firstly, the occurrence times and their predictive distribution can be of interest. Secondly, we are interested in the Poisson process variables.

If we want to predict the occurrence times, both Algorithms 1 and 2 are suitable, because the process $\{T_i, i = 1, 2, ...\}$ is a Markov process itself with transition density (5.3). For within-sample prediction, set $Y_0^{*(k)} = T, k = 1, ..., K$, and otherwise $Y_0^{*(k)} = 0, k = 1, ..., K$. With the notations of the algorithm, it is $\theta = \xi$.

If we want to predict the Poisson process variables, in the case of Algorithm 1 we start with $Y_0^{*(k)} = 0, k = 1, ..., K$, in the case of new-sample prediction and with $Y_0^{*(k)} = N_T, k = 1, ..., K$, otherwise. The transition density is given by

$$\begin{aligned}P(N_{t_i}=l|N_{t_{i-1}}) =&P(N_{t_i}-N_{t_{i-1}}=l-N_{t_{i-1}}|N_{t_{i-1}})\\ =&\frac{1}{l-N_{t_{i-1}}}(\Lambda_\xi(t_i)-\Lambda_\xi(t_{i-1}))^{l-N_{t_{i-1}}}\exp\left(\Lambda_\xi(t_i)-\Lambda_\xi(t_{i-1})\right)\mathbb{1}_{\mathbb{N}}(l-N_{t_{i-1}}).\end{aligned}$$

If we want to predict a whole trajectory of the process, it can be more efficient to sample the event times, since the calculation of the predictive density is computationally costly and the process will stay constant over periods. In the notation of Algorithm 2, it is $\theta=\xi$ and starting values $Y_0^{*(k)}=0, k=1,...,K$, in the case of prediction for a new series and $Y_0^{*(k)}=T, k=1,...,K$, in the case of prediction for the further development, for example the next jump. Here, the time points are $\tau_1=1,...,\tau_M=M$. In Algorithm 2, step (iii), we mentioned the case that also a critical value y_c can be defined, up to which the samples should run. This can be very suitable for the series of event times of the Poisson process. If prediction of the Poisson process up to a time point t^* is the value of interest, this critical value would be $y_c=t^*$. The prediction procedure leads to trajectory samples $T_1^{*(l)}<T_2^{*(l)}<...<T_M^{*(l)}$ or, in the case of stopping at a critical value $T_1^{*(l)}<T_2^{*(l)}<...<T_{M_l}^{*(l)}, l=1,...,L$. The trajectory samples of the Poisson process can be calculated as follows

$$N_t^{*(l)}=\{j:T_j^{*(l)}\leq t<T_{j+1}^{*(l)}\},\ l=1,...,L. \tag{5.5}$$

5.2. Application

In Hermann et al. (2016a), a simulation study each with 1000 series for both presented intensity functions (5.1) and (5.2) was made to validate the presented estimation and prediction procedure for the Poisson process as well as for the jump diffusion process in Section 6. We here restrict to the application for the presented data set.

We will concentrate on the first data series and compare the results for the two intensity rates in (5.1) and (5.2). For the Weibull function (5.1), we start the MH algorithm in $\xi_0^*=(5,0.5)$ and for the exponential function (5.2) in $\xi_0^*=(4,1)$, both are chosen manually with a look on the wire breaking process in comparison to the cumulative intensity function. As mentioned, we here use a non-informative specification, i.e. $p(\xi)=1$, because we do not have expert knowledge. We have only few data points and any prior specification would have a large effect and would have to be justified.

For the estimation, we take into account the first 10 wire breaking times and calculate a prediction for the next five wire breaks. We draw 11 000 samples for the Weibull function and take a burn-in phase of 1000 and a thinning rate of 10. For the exponential function, the chains need longer to converge and do not have a good mixing. We draw 110 000 samples and with a burn-in of 10 000 and a thinning rate of 100, the posterior is approximated well.

For the prediction, we use Algorithm 2 for the event times and calculate for the wire breaking process the variables in (5.5). We see on the left side of Figure 36 the resulting prediction intervals for the eleventh up to the fifteenth event times based on the estimation with the first ten event times. The corresponding prediction intervals for the NHPP, i.e. the wire breaking process, can be seen on the right side of the figure.

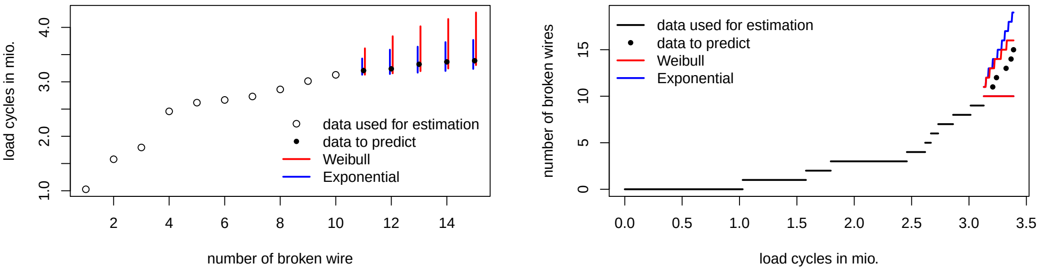


Figure 36: 95% prediction intervals for the first observed wire breaking series of the data set. Estimations are done with the first 10 observed event times, predictions calculated for the next 5 wire breaking times. Left: the predictions for the event times, right: prediction for the NHPP, both based on the exponential (5.2) (blue) and the Weibull (5.1) (red) intensity rates.

Equal to the findings in Hermann et al. (2016a), the exponential intensity rate yields a better result than the Weibull function. Nevertheless, the prediction intervals cover the true values for both curves.

Until now, the intensity rate is a deterministic function. One suitable extension is to include the heuristic fact that the same load is distributed on less wires after wire breaks. The stress range of the experiment would be considered as a respective physical parameter.

Outlook: self-exciting process

Searching for a possibility to include the stress range into the intensity rate, we start with heuristics. In Figure 37, we see the load of the I tension wires over the time. In the beginning of the experiment, the load distributes on 35 wires. When one wire breaks, the load distributes on only 34 and so on. Therefore, the idea is to use this fact in the intensity rate. But in the framework of NHPPes, the intensity rate has to be deterministic and may not depend on the past of the process. To allow this, we turn to the class of self-exciting processes, see, for example, Ríos Insua et al. (2012).

We assume the wire breaking process to have distribution

$$N_t|\{N_r, r < t\} \sim \text{Pois}(\Lambda_\xi(t))$$

in time t, whereby $\Lambda_\xi(t)$ is a random process based on $\{N_r, r < t\}$.

To continue the idea from Figure 37, we assume for the cumulative intensity function

$$\mathbb{E}[N_t|N_r, r < t] = \Lambda_\xi(t) = \sum_{i=1}^{N_{t-}} f_\xi\left(\tfrac{s}{35-(i-1)}\right)(T_i - T_{i-1}) + f_\xi\left(\tfrac{s}{35-N_{t-}}\right)(t - T_{N_{t-}}) \tag{5.6}$$

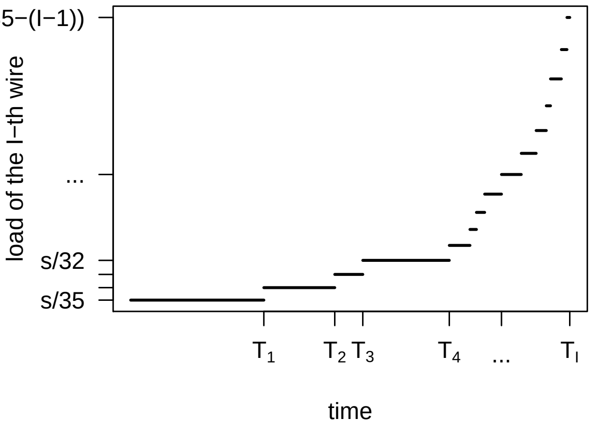


Figure 37: Load over time for the Ith wire dependent on stress range s.

with $T_0 = 0$ and $N_{t-} = \lim_{s \nearrow t} N_s$. Derivation yields the intensity rate $\lambda_\xi(t) = f_\xi\left(\frac{s}{35-N_{t-}}\right)$. This class of load sharing models is well-known in the reliability literature, see, for example, Kvam and Lu (2007); Nakagawa (2007); Crowder et al. (1994).

In Szugat et al. (2016), a simulation free prediction method has been developed and applied to the data set of ten wire breaking series. It turns out, that for the function $f_\xi = \exp(-\xi_1 + \xi_2 \log(x))$, $\xi \in \mathbb{R}_+^2$, the model can predict the data very well. In Müller et al. (2016), two simulation free approximated prediction methods have been presented, one based on the delta method and one based on data depth, and compared for three different functions of f_ξ. High coverage rates of the prediction intervals lead to the assumption that the model is a good choice for the data set. At the moment, a master student, Laura Marquis, is working on the Bayesian analysis for the model.

6. Jump Diffusion Process

We here consider the full model in (2.1). The idea of the jump process for the data is formulated in Hermann et al. (2016a), where a two-stage modeling procedure for a SDE with explicit solution is presented. Firstly, estimation and prediction is presented for the NHPP. Secondly, the posterior distribution of the parameters in the jump diffusion process, modeling the crack width, is approximated by samples drawn by a Gibbs sampler. These parameter samples and the counting process samples, drawn from the predictive distribution from the first modeling step, are employed to approximate the predictive distribution for the jump diffusion process.

In Hermann and Ruggeri (2016), the same process is considered based on an unobserved homogeneous Poisson process (HPP) and applied on wear degradation in cylinder liners. The procedure is extended here for the non-homogeneous case.

6.1. Model Description, Inference and Prediction

In Hermann et al. (2016a), the SDE (2.1) with the special cases

$$b(\phi, t, y) = \phi y, \quad s(\gamma, t, y) = \gamma y, \quad h(\eta, t, y) = \eta y$$

is presented, which is given by

$$dY_t = \phi Y_t \, dt + \gamma Y_t \, dW_t + \eta Y_t \, dN_t, Y_0 = y_0. \tag{6.1}$$

This is a stochastic extension of the simplified version of the Paris-Erdogan equation $dY_t = \phi Y_t \, dt$ as mentioned above in (1.1). In addition, under a homogeneous Poisson process, (6.1) is known as Merton model, introduced in Merton (1976) for modeling stock returns. The SDE has a unique strong solution, which is given by

$$Y_t = y_0 \cdot \exp\left(\phi t - \frac{\gamma^2}{2} t + \gamma W_t + \log(1 + \eta) N_t\right) \tag{6.2}$$

for $\eta > -1$. This can be seen by direct calculation using Itô's formula for noncontinuous semimartingales, see Øksendal and Sulem (2005), p. 7 in example 1.15. As mentioned, inference and prediction for this process is done in Hermann et al. (2016a) for observed NHPP and in Hermann and Ruggeri (2016) for unobserved HPP.

We here restrict to the case of a SDE without explicit solution and consider the Euler approximation in (2.2) for the general SDE (2.1). In addition, we restrict to the case of an unobserved Poisson process and mention the simplification of the case where it is observed.

Based on the approximation variables in (2.2), the parameter vector $\theta = (\phi, \gamma^2, \eta, \xi)$ and definition $\Delta N_i = N_{t_i} - N_{t_{i-1}}$, we assume the Bayesian model

$$\begin{aligned} Y_i | Y_{i-1}, \Delta N_i, \theta &\sim \mathcal{N}(Y_{i-1} + b(\phi, t_{i-1}, Y_{i-1})\Delta_i + h(\eta, t_{i-1}, Y_{i-1})\Delta N_i, s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i), \\ \Delta N_i &\sim \text{Pois}(\Lambda_\xi(t_i) - \Lambda_\xi(t_{i-1})),\ i = 1, ..., n, \\ \phi &\sim \mathcal{N}(m_\phi, v_\phi), \\ \gamma^2 &\sim \text{IG}(a_\gamma, b_\gamma), \\ \eta &\sim \mathcal{N}(m_\eta, v_\eta), \\ \xi &\sim p(\xi) = 1. \end{aligned} \tag{6.3}$$

Estimation

Define $\Delta\Lambda_{\xi,i} := \Lambda_\xi(t_i) - \Lambda_\xi(t_{i-1})$ and $\Delta Y_i = Y_{t_i} - Y_{t_{i-1}}$. This leads to the likelihood conditional on the Poisson process variables

$$\begin{aligned} &p(Y_{(n)} | \theta, N_{(n)}) \\ =& \prod_{i=1}^n \frac{1}{\sqrt{2\pi\Delta_i} s(\gamma, t_{i-1}, Y_{i-1})} \exp\left(-\frac{(\Delta Y_i - b(\phi, t_{i-1}, Y_{i-1})\Delta_i - h(\eta, t_{i-1}, Y_{i-1})\Delta N_i)^2}{2s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i} \right). \end{aligned}$$

This likelihood and the prior distributions from the model definition suffice for the MH steps inside the Metropolis within Gibbs sampler of the parameters ϕ, γ^2, η. The likelihood of ξ is the same as in Section 5 given in (5.4). Estimation of all the parameters ϕ, γ^2, η and ξ will be implemented with a Metropolis within Gibbs sampler in the case of observed Poisson process.

The observability of the Poisson process is a special case of the data set of Heeke, Hermann et al. (2015). In the following, we consider a filtering procedure for the process given in (2.1) for the Euler approximated variables, respectively the Bayes model (6.3), in the case of an unobserved Poisson process. One step is included in the Gibbs sampler for the estimation of $N_{(n)}$.

Since multivariate sampling always is a challenge, we will reduce the problem to the posterior of the independent differences $\Delta N_i \sim \text{Pois}(\Delta\Lambda_{\xi,i})$. It is

$$\Delta Y_i | \Delta N_i, Y_{i-1}, \phi, \gamma^2, \eta \sim \mathcal{N}\left(b(\phi, t_{i-1}, Y_{i-1})\Delta_i + h(\eta, t_{i-1}, Y_{i-1})\Delta N_i, s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i \right).$$

Therefore, the posterior density for ΔN_i is given by

$$\begin{aligned} &p(\Delta N_i | \Delta Y_i, Y_{i-1}, \phi, \gamma^2, \eta, \xi) \\ \propto& \frac{\exp(-\Delta\Lambda_{\xi,i})}{(\Delta N_i)!} (\Delta\Lambda_{\xi,i})^{\Delta N_i} \frac{1}{\sqrt{2\pi\Delta_i} s(\gamma, t_{i-1}, Y_{i-1})} \cdot \\ &\exp\left(-\frac{(\Delta Y_i - b(\phi, t_{i-1}, Y_{i-1})\Delta_i - h(\eta, t_{i-1}, Y_{i-1})\Delta N_i)^2}{2s^2(\gamma, t_{i-1}, Y_{i-1})\Delta_i} \right), \end{aligned}$$

which is not a density of a known distribution family. We propose the following sampling procedure, which is simply the inversion method, see Section 2.2.6, for the kth Gibbs sampler iteration.

Algorithm 6 (Filtering of the Poisson process variables)
For all $i = 1, ..., n-1$

(i) choose $R \in \mathbb{N}$ (large enough) for the candidate set $\{0, ..., R\}$,

(ii) calculate posterior probabilities $p_0, p_1, ..., p_R$ with

$$p_j = p(j|\Delta Y_i, Y_{i-1}, \phi^*_{k-1}, \gamma^{2*}_{k-1}, \eta^*_{k-1}, \xi^*_{k-1}),\ j = 0, 1, ..., R,$$

(iii) draw $u \sim \mathcal{U}(0, \sum_{j=0}^R p_j = 1)$ and calculate $m = \min\{r : \sum_{j=0}^r p_j \geq u\}$, and

(iv) set $\Delta N_i^k = m$.

Additionally, set $N_{(n)}^k = (0, \Delta N_1^k, ..., \sum_{i=1}^n \Delta N_i^k)$.

Of course, the proposed filtering procedure is computationally costly. One alternative would be a simple MH step for all $\Delta N_i, i = 1, ..., n-1$. But the whole Gibbs sampler gets inefficient with a MH step, since all other parameters are updated for each k, except γ^2 but the acceptance rate is very high due to the calculated proposal density. Another alternative is a particle filtering procedure, see Johannes et al. (2009), but this gets computationally more expensive than our proposed method, since a number of particles has to be drawn. In our model, a posterior of ΔN_i can be calculated, therefore, the proposed method is suitable.

After choosing starting values $\phi_0^*, \eta_0^*, \gamma_0^{2*}, \xi_0^*$, the Metropolis within Gibbs sampler explained in Section 2.2.3 unites all estimation steps for $\theta = (N_{(n)}, \xi, \phi, \gamma^2, \eta)$ as follows

$$\begin{aligned}
N_{(n)}^k &\sim p(N_{(n)}|Y_{(n)}, \phi^*_{k-1}, \eta^*_{k-1}, \gamma^{2*}_{k-1}, \xi^*_{k-1}),\\
\xi_k^* &\sim p(\xi|N_{(n)}^k),\\
\phi_k^* &\sim p(\phi|Y_{(n)}, N_{(n)}^k, \eta^*_{k-1}, \gamma^{2*}_{k-1}),\\
\eta_k^* &\sim p(\eta|Y_{(n)}, N_{(n)}^k, \phi_k^*, \gamma^{2*}_{k-1}),\\
\gamma_k^{2*} &\sim p(\gamma^2|Y_{(n)}, N_{(n)}^k, \phi_k^*, \eta_k^*),\ k = 1, ..., K.
\end{aligned}$$

Of course, for special cases of functions b, s and h, full conditional posteriors could be analytically available, possibly with conjugate priors. But, for generalization, this will not be done here.

In the case of observed NHPP variables, the first step of the Gibbs sampler can be ignored, the remaining steps are equal.

Prediction

As already seen in the estimation procedure, $N_{(n)}$ is latent variable and, therefore, treated like an additional parameter. Similar to prediction of a random effect in Sections 3 and 4, we first need a prediction of $N_{(n)}$, which is already described in Section 5.

In the case of Algorithm 2, the predictive samples $N_{\tau_m}^{*(k)}, k = 1, ..., K, m = 1, ..., M$, from the NHPP derived from (5.5) are used for the prediction of Y^* in t^*. With $\theta = (N_{(n)}, \xi, \phi, \gamma^2, \eta)$, the transition density in Algorithms 1 and 2 is the density of the distribution given by

$$\begin{aligned}&Y_m^*|Y_{m-1}^*, N_{\tau_m}, N_{\tau_{m-1}}, \theta \sim \\ &\qquad \mathcal{N}(Y_{m-1}^* + b(\phi, \tau_{m-1}^*, Y_{m-1}^*)\Delta^* + h(\eta, \tau_{m-1}^*, Y_{m-1}^*)(N_{\tau_m} - N_{\tau_{m-1}}), s^2(\gamma, \tau_{m-1}^*, Y_{m-1}^*)\Delta^*).\end{aligned}$$

Starting values are $Y_0^{*(l)} := Y_n$ for the case of prediction of the further development of the current series and $Y_0^{*(l)} := y_0$ in the case of prediction for a new series.

In the case of Algorithm 1, the starting values and the transition density are the same. There are two possibilities for the latent Poisson process variables. Firstly, the samples $N_{\tau_m}^{*(k)}, k = 1, ..., K, m = 1, ..., M$, derived from (5.5), drawn with Algorithm 2 can be taken. Secondly, we can sample predicted samples of the differences ΔN_i, since we do not need trajectories for the latent Poisson process variables to include in the stepwise Algorithm 1. In that case, we sample for $m = 1, ..., M$:

$$\Delta N_m^{*(k)} \sim p(\Delta N_m|N_{(n)}) \approx \frac{1}{K}\sum_{k=1}^{K} \frac{\exp\left(-(\Lambda_{\xi_k^*}(\tau_m) - \Lambda_{\xi_k^*}(\tau_{m-1}))\right)}{(\Delta N_m)!}(\Lambda_{\xi_k^*}(\tau_m) - \Lambda_{\xi_k^*}(\tau_{m-1}))^{\Delta N_m}.$$

Comparison of explictit and Euler approximated process

In the following, we will investigate the difference between the explicit solution and the Euler approximated process by an example. Model (6.2) will be compared to the Euler approximated SDE in (6.1). We arbitrarily set parameters $\phi = 0.5, \eta = 0.1, \gamma^2 = 0.01$ and $\xi = (2, 0.2)$ with the power law intensity rate function in (5.1). We start the process in $y_0 = 0.5$ and simulate in $t_0 = 0, ..., t_{100} = 1$. Both processes can be seen in Figure 38 on the right with the corresponding Poisson process on the left. The difference between the curves is small but recognizable in the last quarter of the curve.

Now, we will treat the Euler simulated curve with the estimation procedure from the explicit solution and the other way round for the simulated curve from (6.2) that will be estimated with posteriors based on the Euler approximation variables. In both cases, we keep the Poisson process variable as observed. In Figure 39, the resulting posterior densities for the parameters ϕ, γ^2 and η can be seen. Both solid lines are posterior densities dependent on the Euler simulated variables, and the dotted lines dependent on the variables in (6.2). At first, we can summarize, that both procedures applied to the corresponding data set, lead to similar posterior densities (red lines). The effect of the estimation on the wrong data set is small for ϕ. The effect for γ^2 can be ascribed to the biased estimation of η. The jump high parameter η seems to be sensitive to the choice of the model, Euler or explicit.

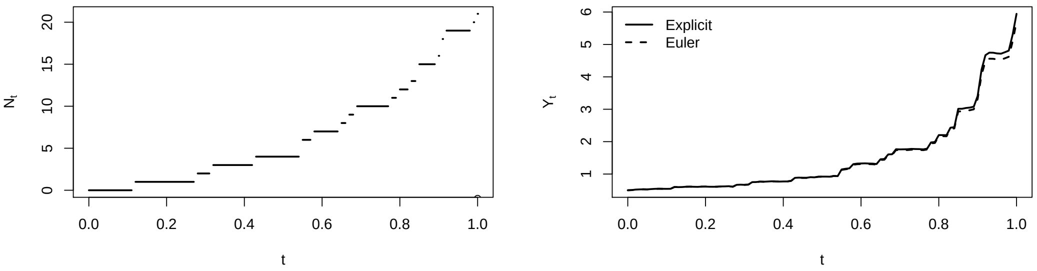


Figure 38: Left: simulated Poisson process, right: simulated process with formula (6.2) (solid) and with the Euler approximation (dotted).

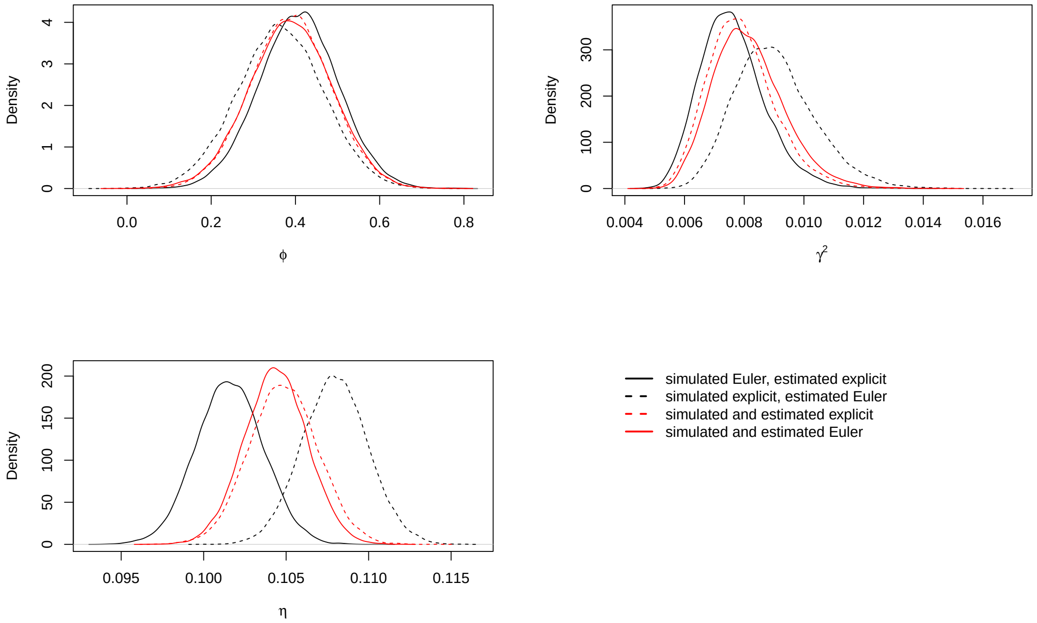


Figure 39: Comparison of posterior densities for ϕ (top left), γ^2 (top right) and η (bottom left). Solid lines: posterior based on simulated series with Euler, dotted lines: based on simulated series with explicit solution (6.2), black lines: estimation each with the other procedure, red lines: estimation each with the corresponding procedure.

With a look on Figure 40, it becomes clear that the biased estimation of η does not effect the prediction. The red lines mark the 95% prediction intervals. The predictive distribution was approximated with Algorithm 2 based on $K = 10^4$ samples from the posterior, in each step $L = 10^4$ samples are drawn. The solid red lines are the corresponding prediction intervals for the solid curve, based on the estimations with the explicit solution, whereas the dotted red lines mark the prediction intervals for the dotted black curve, estimation and predict with the explicit solution. The difference between the red curves, solid and dotted, is negligible.

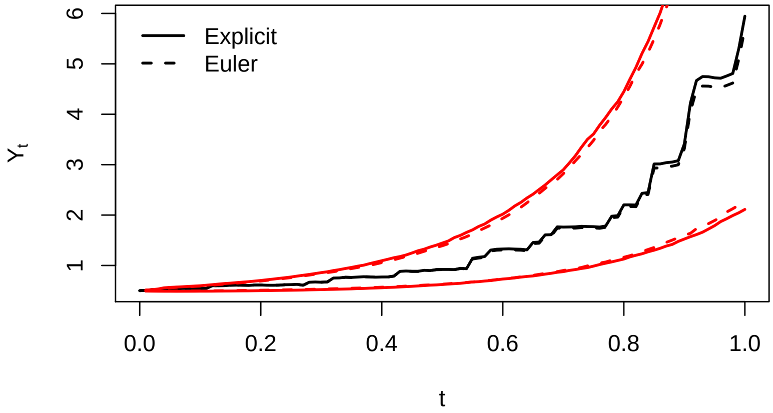


Figure 40: Prediction results; in black: simulated process with formula (6.2) (solid) and with the Euler approximation (dotted), in red: 95% prediction intervals, solid: estimation and prediction with the Euler approximation for the solid black curve simulated with explicit solution, dotted: estimation and prediction with (6.2) for the dotted black curve simulated with Euler.

6.2. Application

Again, we will apply the presented methods to the first series of the data set presented in Heeke, Hermann et al. (2015). Due to the results from Section 5, we restrict here to the exponential intensity rate (5.2). Similar to the work in Hermann et al. (2016a), we consider the model in Section 6.1, but based on the Euler approximation instead of the explicit solution (6.2). In the preparation of this thesis, both models, the Euler approximated and the explicit process, were compared and the results were equal. We choose the functions

$$b(\phi,t,y)=\phi y,\quad s(\gamma,t,y)=\gamma y,\quad h(\eta,t,y)=\eta y \text{ and } \Lambda_\xi(t)=\exp(\xi_1 t-\xi_2)-\exp(-\xi_2)$$

and the following starting values

$$\eta=0.1,\quad \phi=0.1,\quad \gamma=0.1,\quad \xi=(1,0.5),$$

again, manually by simulating the model and comparing the data series. In contrast to the calculations in Hermann et al. (2016a), which were based on the conjugate priors for the normal likelihood of the logarithmic variables, we here can also use a non-informative approach, as done in the following. As mentioned, we also did the calculations for the explicit model and used the starting values for the prior means and the standard deviations and the results were equal. This is not surprising because of the large amount of data points.

At first, we consider the new filtering algorithm for the whole data series. For the whole Gibbs sampler, we draw 101 000 chain iterations, take a burn-in of 1000 samples and a thinning rate of 10. In Figure 41, the estimated Poisson process variables are displayed in green lines, in comparison to the black line that is assumed to be the observed one. Because they are only observed through sound measurements, it is impossible to say if one or more wires are broken at the same time. This leads to a new measuring uncertainty, that is not taken into account yet. The question arises, if it would be useful to filter the counting process instead of taking it as observed. We can see, that up to the fifth broken wire, the lines are equal. Afterwards, the filtering algorithm assumes two broken wires at some times instead of only one for all event times. In addition, we can say that there is only few variation between the green lines, which means that the algorithm draws very often the same trajectory for the NHPP variables. This could be a hint that the algorithm is sure about the estimation of the latent process, since they are drawn independent from each other. Of course, these findings are fully dependent on the model which is just an assumption for the underlying data series. In addition, this algorithm is dependent on the starting values, as discussed in Hermann and Ruggeri (2016), because the likelihood has local maxima, in which the algorithm can end up dependent on the starting values.

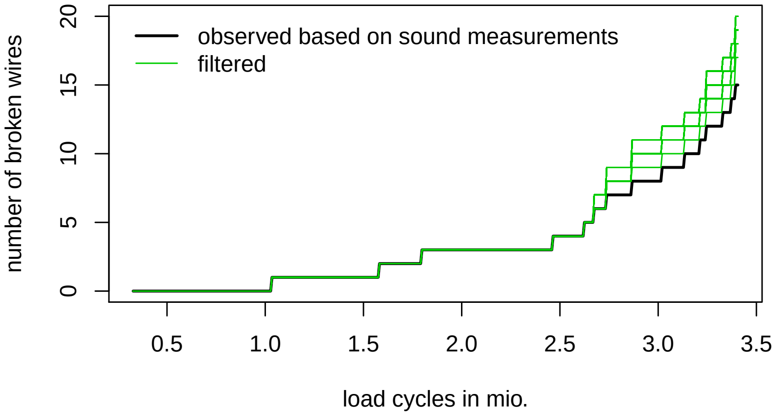
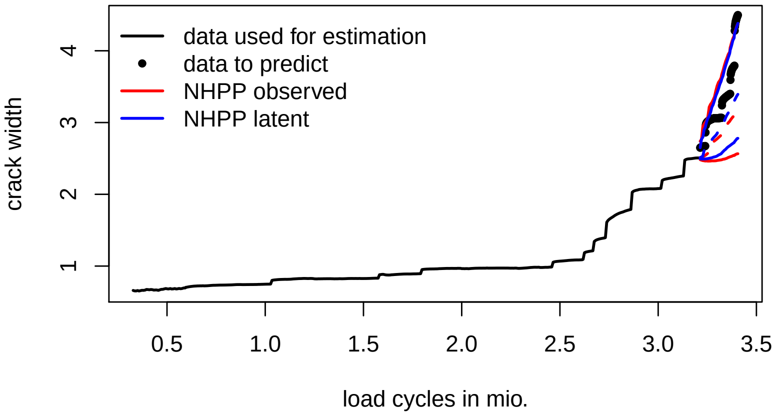


Figure 41: Filtered counting process variables from the Gibbs sampler (green), in black: the trajectory observed based on sound measurements.

In the following, we want to investigate the prediction results. Again, we truncate the series after the first 10 observed jumps for estimation and predict the further development. All algorithm specifications are the same as before. For the prediction, we use Algorithm 2. In Figure 42, we compare the prediction results for the investigated model based on the estimations with the observed Poisson process variables and the filtered one. Top, the whole series are shown. In the bottom figure, we only see the points to be predicted. One can see, that the intervals based on the samples from the Gibbs sampler with the filtering step for the counting process, in blue, are smaller and the median of the predictive distribution, i.e. the dotted blue line, gets closer to the observed data than the red dotted line, which is the median of predictive distribution based on the observed Poisson process.



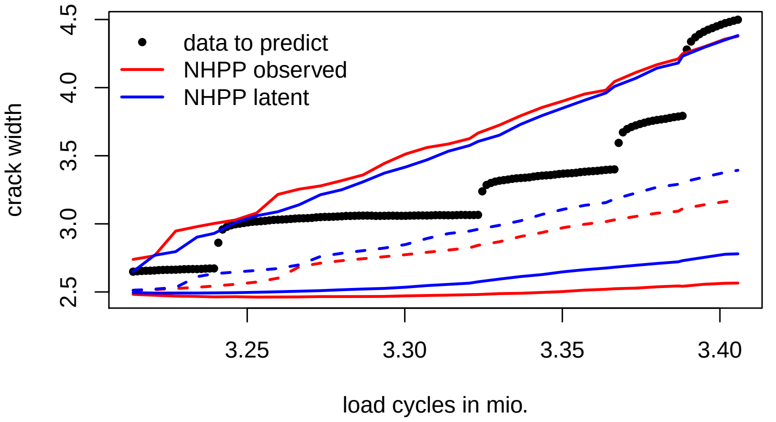


Figure 42: 95 % prediction intervals with Algorithm 2 for the first series with the jump diffusion model. In blue, prediction results based on the Gibbs sampler with filtering Algorithm 6 and in red, based on the Gibbs sampler with fixed, i.e. observed, Poisson process variables. Top: whole series, bottom: zoomed in the interesting area of predicted values.

7. Jump Regression Model

During the interdisciplinary work in the project B5 "Statistical methods for damage processes under cyclic load" of the collaborative research center 823, a physically motivated function has been developed to describe the crack width based on the observed jump process. This idea is formulated in the PhD thesis of Guido Heeke, see Heeke (2016), p. 129, Section 6.1.3, or the joint work Heeke, Hermann et al. (2015).

Based on the observed jump process $\{N_t, t \in [0,T]\}$, the concrete crack growth process can be recalculated with the time-dependent crack width function

$$w(t) = \frac{(1-k(t)) \cdot (\Delta\sigma(t))^2 \cdot A_P(t)}{0.72 \cdot \pi \cdot f_{ctm} \cdot E_P \cdot \sqrt{A_P(t)}}$$

with

$$k(t) = k_l + (k_s - k_l) \cdot e^{t \cdot c}, \qquad A_P(t) = A_P \cdot \left(1 - \frac{N_t}{35}\right),$$
$$\Delta\sigma(t) = \sigma_2 \cdot \frac{A_P}{A_P(t)} - \sigma_1 = \sigma_2 \cdot \frac{1}{1 - \frac{N_t}{35}} - \sigma_1,$$

$k_l, k_s, c, A_P, \sigma_1, \sigma_2, f_{ctm}, E_P$ constants, which are approximately known, but not exactly.

The first step is to transform the function to a more flexible parametric version with estimable parameters, because some of the constants in that function are approximate values. We obtain

$$\widetilde{w}(t, N_t, \phi) = (\phi_1 - \phi_2 \cdot \exp(-t \cdot \phi_3)) \cdot \tfrac{1}{\sqrt{h(N_t)}} \left(h(N_t) - \phi_4\right)^2,$$

where $\phi_1 = \frac{(1-k_l)\cdot\sqrt{A_p}\cdot\sigma_2^2}{0{,}72\cdot\pi\cdot f_{ctm}\cdot E_p}$, $\phi_2 = \frac{(k_s-k_l)\cdot\sqrt{A_p}\cdot\sigma_2^2}{0{,}72\cdot\pi\cdot f_{ctm}\cdot E_p}$, $\phi_3 = -c$, $\phi_4 = \frac{\sigma_1}{\sigma_2}$ and $h(x) = \frac{1}{1-\frac{x}{35}}$. The function $h(x) = \frac{1}{1-\frac{x}{35}} = \frac{35}{35-x}$ is motivated as follows. As seen in Section 5, in the beginning of the experiment, the load is distributed on 35 wires. With every broken wire, function $h(x)$ makes a jump and stays constant in-between. The corresponding calculation can be found in the Appendix B. Of course, other parameterizations are possible. However, if more than four parameters are chosen, they will not be identifiable. We have to mention, that this function only describes the increments of the crack growth, not the position itself. In practice, the starting value has to be subtracted from the data and $\widetilde{w}(t, N_t, \phi) - \widetilde{w}(t_0, N_{t_0}, \phi)$ has to be used. For notation simplicity, we skip this fact here.

7.1. Model Description, Inference and Prediction

The simplest possibility to include this function into a stochastic model is a non-linear regression model dependent on the Poisson process variables. The corresponding Bayes model is given by

$$\begin{aligned} Y_i &= \widetilde{w}(t_i, N_{t_i}, \phi) + \epsilon_i, \\ \epsilon_i &\sim \mathcal{N}(0, \sigma^2) \text{ i.i.d.}, \\ N_{t_i} &\sim \text{Pois}(\Lambda_\xi(t_i)), \; i = 0, ..., n, \\ \phi | m_\phi, V_\phi &\sim \mathcal{N}(m_\phi, V_\phi), \\ \sigma^2 | a_\sigma, b_\sigma &\sim \text{IG}(a_\sigma, b_\sigma), \\ \xi &\sim p(\xi) = 1. \end{aligned}$$

Here, ϵ_i and N_{t_i} have to be stochastically independent. Conditional on the counting process variables $N_{t_0}, N_{t_1}, ..., N_{t_n}$, we have a regression model whose parameters can be estimated based on the likelihood. Bayesian inference for non-linear regression models can be found in Carlin and Louis (2009). Following from

$$Y_i | N_{t_i}, \phi, \sigma^2 \sim \mathcal{N}\left(\widetilde{w}(t_i, N_{t_i}, \phi), \sigma^2\right), \; i = 0, ..., n,$$

the conditional likelihood is a product of normal distribution densities. Because of the non-linear function we have no closed form of the posterior of ϕ. Therefore, a Metropolis within Gibbs sampler will be used with $\theta = (\phi, \sigma^2, \xi)$, whereby

$$\sigma^2 | Y_{(n)}, N_{(n)}, \phi \sim \text{IG}\left(a_\sigma + \frac{n+1}{2}, b_\sigma + \frac{1}{2}\sum_{i=0}^{n}(Y_i - \widetilde{w}(t_i, N_{t_i}, \phi))^2\right).$$

The Gibbs sampler unites the three estimation steps as follows

$$\begin{aligned} \phi_k^* &\sim p(\phi | Y_{(n)}, N_{(n)}, \sigma_{k-1}^{2*}), \\ \sigma_k^{2*} &\sim p(\sigma^2 | Y_{(n)}, N_{(n)}, \phi_k^*), \\ \xi_k^* &\sim p(\xi | N_{(n)}), \; k = 1, ..., K. \end{aligned}$$

Here, we have expert knowledge for the parameter ϕ, which can be used for the prior parameters. If we denote the vector resulting from the physical constants given by the expert with ϕ^0, we assume $m_\phi = \sqrt{\text{diag}(V_\phi)} = \phi^0$.

With the samples $\phi_1^*, \sigma_1^{*2}, ..., \phi_K^*, \sigma_K^{*2}$ from the posterior distribution we can approximate the predictive distribution

$$\begin{aligned} & p(Y_i^* | Y_{(n)}, N_{(n)}) \\ =& \int p(Y_i^* | \phi, \sigma^2, N_{t_i}^*) \cdot p(\phi, \sigma^2 | Y_{(n)}, N_{(n)}) \cdot p(N_{t_i}^* | N_{(n)}) \, d(\phi, \sigma^2, N_{t_i}^*) \\ \approx& \frac{1}{K}\sum_{k=1}^{K} p(Y_i^* | \phi_k^*, \sigma_k^{*2}, N_{t_i}^{*(k)}), i = 0, ..., n, \end{aligned} \tag{7.1}$$

with $N_{t_i}^{*(k)}, \; k = 1, ..., K$, being the samples from the predictive distribution of the Poisson process defined in (5.5). The distribution in (7.1) can be approximated with rejection sampling or inversion method, see Section 2.

In Heeke, Hermann et al. (2015), we found out, that the first part of the regression function $\phi_1 - \phi_2 \cdot \exp(-t \cdot \phi_3)$ only describes the very beginning of the series and stays constant for the rest. However, we are mainly interested in the prediction for the last part of the experiment and we lose efficiency by estimating needless parameters. Therefore, we simplify the function $\widetilde{w}$ to

$$\widetilde{w}_s(N_t, \phi) = \phi_1 \cdot \frac{1}{\sqrt{h(N_t)}} \left(h(N_t) - \phi_2\right)^2$$

and estimate the parameters similar to the regression model defined above. For the estimation procedure, only the likelihood has to be changed in the MH step of the posterior sampling for ϕ. For the predictive distribution (7.1), only the new likelihood has to be used.

Outlook: filtering of the unobserved Poisson process

For this approach, again the observed Poisson process is very special for the data set of Heeke, Hermann et al. (2015). Filtering the unobserved Poisson process is not as self-evident as in the jump diffusion model, since the likelihood does not only depend on the independent differences of N_t. One possibility would be the particle Gibbs sampler with the CSMC algorithm step for the latent $N_{(n)}$ described in Section 4 or the particle filtering, developed for jump diffusions in Johannes et al. (2009), might be extendable to the current model, but this will be future work.

7.2. Application

Analogous to the last two sections, we here consider the first series of the crack width data, seen in Figure 35. In contrast to the evaluations in Heeke, Hermann et al. (2015), we here use the exponential intensity rate (5.2) due to the results in Section 5 and in Hermann et al. (2016a). For comparison with the jump diffusion model, we here as well truncate the series after the 10th jump and use the first part for estimation and predict the further development. Since we have seen two functions $\widetilde{w}$ and $\widetilde{w}_s$, we will compare the prediction results with both curves. For the first function, as mentioned, we have expert knowledge that leads to the starting values $\phi_0^* = (4.36, 2.03, 20, 0.64)$. For the simplified curve, we choose $(\phi_{1,0}^* - \phi_{2,0}^* \cdot \exp(\phi_{3,0}^*), \phi_{4,0}^*)$ as starting parameter. For the intensity rate parameter, we again start with $\xi_0^* = (1, 0.5)$ and for the variance, we choose $\gamma^2 = 1$. Except for ξ, all prior parameter are chosen, so that prior mean and standard deviation is equal to the starting values.

For the prediction, we first sample event times with Algorithm 2 and calculate the predictive samples for the wire breaking process in (5.5). Afterwards, we sample from the predictive distribution in (7.1). The pointwise 95% prediction intervals can be seen in Figure 43, the whole series on the left and only the points to be predicted on the right side. In addition, the prediction intervals with the jump diffusion model from Section 6 based on the estimations with observed Poisson process are displayed for comparison. At first, we can say that the jump diffusion model leads to smaller intervals and up to the fourth jump the prediction intervals cover the true values. The jump regression model leads to larger intervals that cover all observed data points. The blue line shows the results for the simplified curve which are smoother than these from the original curve. The reason is a larger estimated variance. In the model based on function $\widetilde{w}$, we have a posterior mean of 0.004 and for $\widetilde{w}_s$ it is 0.046, which is much larger and the intervals look smoother. But, even with a higher estimated variance, the intervals are smaller and cover the true values.

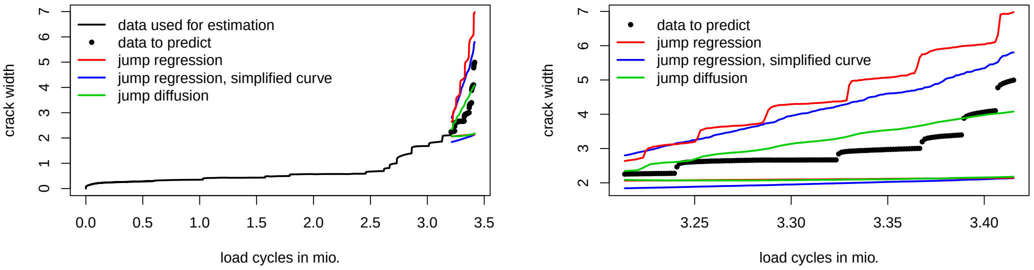


Figure 43: Prediction for the first observed series of the crack width data set. Estimations are done with the series up to 10 observed event times, predictions are made for further development. Left: whole series, right: zoomed in the interesting area, both based on the exponential (5.2) intensity rate. In addition: comparison with the jump diffusion model (in green).

8. Description of the R Package BaPreStoPro

In this thesis, we have seen lots of models with different estimation and prediction procedures. Beside the theoretical properties, the practical realization is important as well. Implementation can be time-consuming and, for example for the particle Gibbs sampler, very complicated and error-prone. An R package is a convenient tool for users with real data. The package presented in the following, provides solutions for the inference described in this thesis. In detail, for each presented model, an S4 class is constructed, wherein all the model details are stored. For each model class, simulation, estimation and prediction methods are available. In addition, for the estimation output, plot functions are implemented for a convenient visualization of the results.

In addition to the models described before, a non-linear regression model with the corresponding hierarchical version is also implemented for an easy comparison. These models are given by

$$\begin{aligned} Y_i|\phi,\gamma^2 &\sim \mathcal{N}(f(\phi,t_i),\gamma^2 s^2(t_i)),\ i=0,...,n,\\ \phi|m_\phi,V_\phi &\sim \mathcal{N}(m_\phi,V_\phi),\quad V_\phi=\mathrm{diag}(v_1^2,...,v_p^2),\\ \gamma^2|a_\gamma,b_\gamma &\sim \mathrm{IG}(a_\gamma,b_\gamma)\end{aligned}$$

for the single series model and by

$$\begin{aligned} Y_{ij}|\phi_j,\gamma^2 &\sim \mathcal{N}(f(\phi_j,t_{i,j}),\gamma^2 s^2(t_{i,j})),\ i=1,...,n_j,\\ \phi_j|\mu,\Omega &\sim \mathcal{N}(\mu,\Omega)\text{ i.i.d.},\ j=1,...,J,\quad \Omega=\mathrm{diag}(\omega_1^2,...,\omega_p^2),\\ \mu|m_\mu,V_\mu &\sim \mathcal{N}(m_\mu,V_\mu),\quad V_\mu=\mathrm{diag}(v_1^2,...,v_p^2),\\ \omega_r^2|a_{\omega,r},b_{\omega,r} &\sim \mathrm{IG}(a_{\omega,r},b_{\omega,r}),\ r=1,...,p,\\ \gamma^2|a_\gamma,b_\gamma &\sim \mathrm{IG}(a_\gamma,b_\gamma)\end{aligned}$$

for the hierarchical model.

The package BaPreStoPro is structured as follows. For each of the models, there is one S4 model class as can be seen in the second column of Table 5.

They all are constructed by the function `set.to.class` with the main input parameter `class.name`, denoting the name of the model class, and the input parameters `parameter`, `prior`, `start`, `b.fun`, `s.fun`, `h.fun`, `sT.fun`, `y0.fun`, `fun`, `Lambda` and `priorDensity`. An overview is given in Table 6. `prior` is optional, if missing, it is calculated from the list entries of `parameter` so that mean and standard deviation of the prior distribution are equal to the values of `parameter`. Analogously, `start`, if missing, is set equal to `parameter`. `b.fun` defines the function $b(\phi,t,y)$ in all (jump) diffusion models, `s.fun` defines the function $s(\gamma^2,t,y)$, and `h.fun` the function $h(\theta,t,y)$ in the jump diffusion model. `sT.fun` stands for the variance function $\widetilde{s}(t,y)$ in the (mixed) diffusion models as well as for $s^2(t)$ in the (mixed) regression models. `y0.fun` denotes the starting value function $y_0(\phi)$ for the hidden and mixed diffusion models. `fun` is the regression function $\widetilde{w}(t,N_t,\phi)$ in the jump regression and $f(\phi,t)$ in the (mixed) regression model. `Lambda` is for all models, containing the NHPP, the cumulative intensity function $\Lambda_\xi(t)$. `priorDensity` is for the jump diffusion model a list of prior densities,

Model	Class object	
	`simulate & estimate`	`plot & predict`
diffusion	`Diffusion`	`est.Diffusion`
hierarchical diffusion	`mixedDiffusion`	`est.mixedDiffusion`
hidden diffusion	`hiddenDiffusion`	`est.hiddenDiffusion`
hierarchical hidden diffusion	`hiddenmixedDiffusion`	`est.hiddenmixedDiffusion`
NHPP	`NHPP`	`est.NHPP`
jump diffusion	`jumpDiffusion`	`est.jumpDiffusion`
jump diffusion (6.2)	`Merton`	`est.Merton`
jump regression	`jumpRegression`	`est.jumpRegression`
regression	`Regression`	`est.Regression`
hierarchical regression	`mixedRegression`	`est.mixedRegression`

Table 5: Overview of the class names for the corresponding methods in the different models. For each model in the left column, a model class is available for which a simulation and an estimation method is implemented. For the resulting estimation class objects in the right column, a plot and a prediction method is realized.

input parameter	meaning
`parameter`	values for the starting and the prior values, if they are not specified
`prior`	list of prior values, if not specified, chosen so that mean and standard deviation is equal to the values in `parameter`
`start`	list of starting values, if not specified, equal to the values in `parameter`
`b.fun`	function $b(\phi, t, y)$ in all (jump) diffusion models
`s.fun`	function $s(\gamma^2, t, y)$ in the jump diffusion model
`h.fun`	function $h(\eta, t, y)$ in the jump diffusion model
`sT.fun`	variance function $\widetilde{s}(t, y)$ in the (mixed) diffusion models as well as for $s^2(t)$ in the (mixed) regression models
`y0.fun`	starting value function $y_0(\phi)$ for the hidden and mixed diffusion models
`fun`	regression function $\widetilde{w}(t, N_t, \phi)$ in the jump regression and $f(\phi, t)$ in the (mixed) regression model
`Lambda`	cumulative intensity function $\Lambda_\xi(t)$
`priorDensity`	list of prior densities in the jump diffusion models

Table 6: Overview of the input parameters of the function `set.to.class`.

for the Merton model and the NHPP a prior density for ξ, defaults are non-informative approaches.

For each of the model classes, a simulation method `simulate` is available. With the respective data set, method `estimate` can be applied to the model class with the additional input parameters `t`, i.e. the time vector $(\tau_0, \tau_1, ..., \tau_M)$, `data`, denoting the respective data set, which can be a vector, a matrix or a list in the respective model, and the number of Markov chain iterations, `nMCMC`. In addition, there is the possibility to choose the proposal standard deviation for the MH algorithms, `propSd`. If this is not specified, it is chosen proportional to the starting values, dependent on the model. If `adapt = TRUE`, this proposal standard deviation is just the starting one and is adapted after every 50 iterations, if the acceptance rate is smaller than 30% or larger than 60%, see for adaptive MCMC Rosenthal (2011). The proposal density itself can also be chosen between the normal and the log-normal distribution, `proposal = "normal"` or `"lognormal"`. Output of the `estimate` method is a new S4 class composed of `"est."` and the model class name, see Table 5 in the third column.

This estimation class object is an input parameter in the method `predict`. One of the main input parameters is `pred.alg` and denotes the prediction algorithm. Algorithm 1 (default) is named `"Distribution"`. Algorithm 2 is implemented with the `pred.alg = "Trajectory"`. As mentioned, two additional sampling prediction algorithms are implemented. With `pred.alg = "simpleTrajectory"`, the presented Algorithm 3 is implemented and with `pred.alg = "simpleBayesTrajectory"`, Algorithm 4 is conducted. The second main input parameter is `which.series`, where one can decide between `"new"`, i.e. $\tau_0 = t_0$, and `"current"`, i.e. $\tau_0 = t_n$. An overview can be seen in Table 7.

input parameter	choices	description
`pred.alg`	`"Distribution"`	Algorithm 1
	`"Trajectory"`	Algorithm 2
	`"simpleTrajectory"`	Algorithm 3
	`"simpleBayesTrajectory"`	Algorithm 4
`which.series`	`"current"`	$\tau_0 = t_n$
	`"new"`	$\tau_0 = t_0$

Table 7: Overview of the main input parameters of prediction method `predict`.

In the following, we go into detail through the examples given during the thesis.

Diffusion

The process given in Example 1 is defined by the functions $s(\gamma, t, y) = \gamma\widetilde{s}(y) = \gamma y$ and $b(\phi, t, y) = \phi y$, and the parameters $y_0 = 1, \phi = 2$ and $\gamma = 1$. We translate that to R-code:

```
b.fun <- function(phi, t, y) phi * y
sT.fun <- function(t, y) y
model <- set.to.class("Diffusion", parameter = list(phi = 2, gamma2 = 1),
      b.fun = b.fun, sT.fun = sT.fun)
t <- seq(0, 1, by = 0.01)
Y <- simulate(model, seed = 123, t = t, y0 = 1)
```

The simulated trajectory can be seen in Figure 44. There is an optional input parameter `mw`, default is 1, whose inverse serves as mesh width for simulating time-continuity of the process. For example, if `mw = 10`, nine points are equidistantly added between each two points of `t` and the process is simulated on the finer time grid. Afterwards, every tenth simulated point is given out as simulated series. We here restrict to `mw = 1` to avoid biased estimations.

Estimation is done by the method `estimate` as follows:

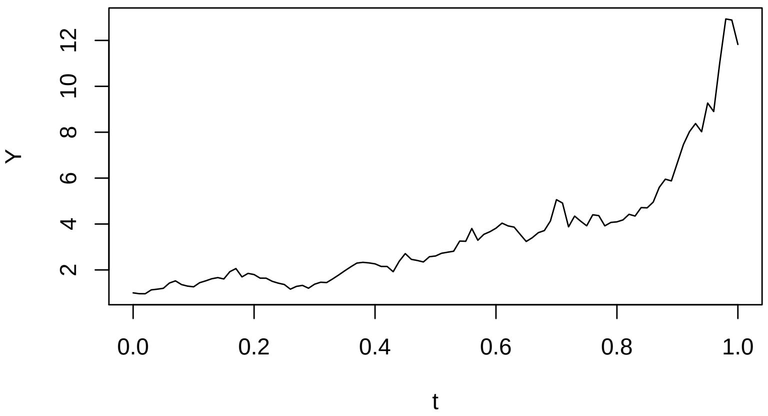


Figure 44: Trajectory created with method `simulate` for the diffusion model.

```
est <- estimate(model, t, Y, 21000)
plot(est)
```

Output of the method `estimate` for class object `Diffusion` is a new class object `est.Diffusion`, which contains all information of the model as well as the data and the estimation results. In addition it contains a proposal for the thinning rate and the burn-in phase. Both are input parameters in the following methods, but if they are missing, the proposed values are taken.

The proposed `burnIn` is calculated by dividing the Markov chains into 10 blocks and calculate the 95% credibility intervals and the respective mean. Starting in the first one, the block is taken as burn-in as long as the mean of the current block is not in the credibility interval of the following block or vice versa. The thinning rate is proposed by the smallest lag, which leads to a chain autocorrelation of less than 80%. It is not easy to automate these choices, so it is highly recommended by the author to verify the chains manually, for example with the R package `coda`, see Plummer et al. (2006).

The Markov chains are visualized with method `plot`, which has several options for input parameter `style`, namely `chains, acf` and `density`. The first one can be seen in Figure 45, the second one shows the autocorrelation functions for the chains and the third one the resulting posterior densities. Another important input parameter is `reduced`. If it is set to `TRUE`, the chains are reduced by the burn-in phase and the thinning rate, both optional input parameters but also proposed by the estimation method.

Class object `est.Diffusion` is used for method `predict`. Thinning rate and burn-in phase are also input parameters in the method `predict`.

```
burnIn <- 1000
thinning <- 2  # possibly small, only for example
```

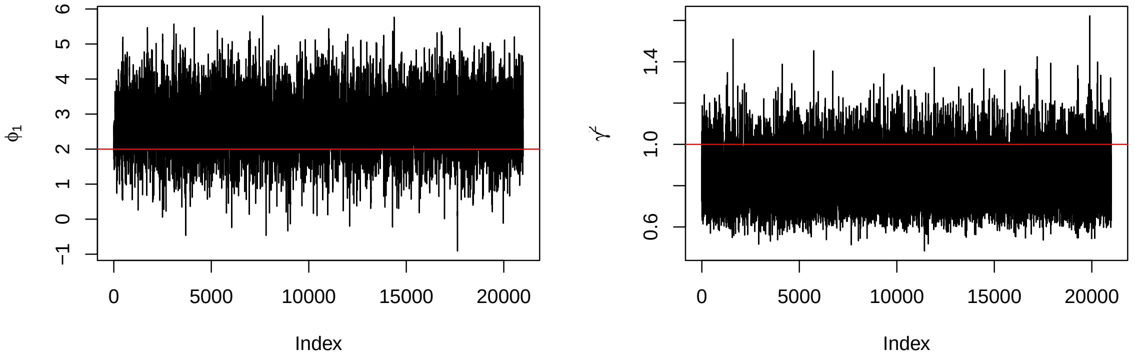
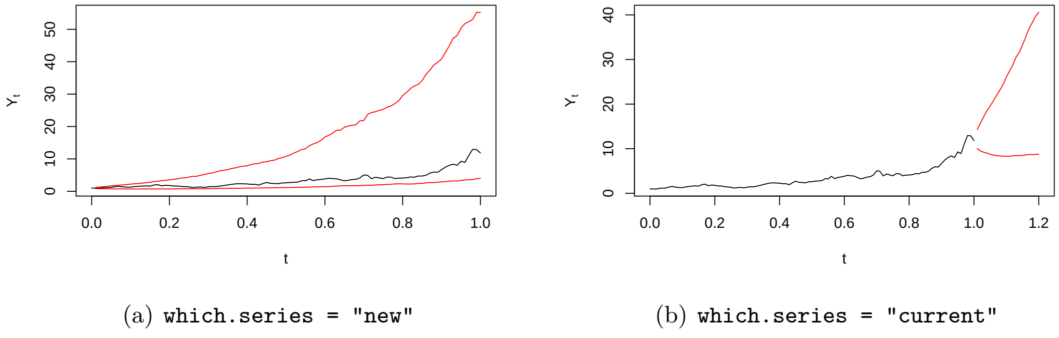


Figure 45: Markov chains produced by method `estimate` and visualized with method `plot` for the diffusion process in Figure 44.

```
pred <- predict(est, burnIn = burnIn, thinning = thinning,
   b.fun.mat = function(phi, t, y) phi[,1]*y)
```

Since the function `b.fun` is stored in the `est.Diffusion` object, the input parameter `b.fun.mat` is not necessary but the algorithm gets much faster with the matrix-wise function definition. If `plot.prediction = TRUE`, the 1-`level` prediction intervals are plotted with the observation series used for estimation. This leads to Figure 46(a). Default value for the input parameter `pred.alg`, i.e. the prediction algorithm, is `"Distribution"`, which denotes the pointwise prediction Algorithm 1.



(a) `which.series = "new"`

(b) `which.series = "current"`

Figure 46: 95% prediction intervals of method `predict`, visualized, if input parameter `plot.prediction` is `TRUE` (default).

One important input parameter of method `predict` is `which.series`. Default is `"new"`, which leads to prediction of a new series, starting in the first point of the observation series. The second option is `"current"`, which yields a prediction of the further development of the observed series. If no vector of time points is specified (also an input parameter, t), the input variable M2pred can be used (default: 10), which is the number of points that will be predicted with the time distances of the observation time vector. This can be seen as follows:

```
pred.current <- predict(est, b.fun.mat = function(phi, t, y) phi[,1]*y,
  which.series = "current", M2pred = 20)
```

The result can be seen in Figure 46(b). Default values for the starting points of the prediction is for `which.series = "new"` the starting point of the observed series, and for `which.series = "current"` the last observation point. If desired, this can be changed with the input parameter `y.start`, which denotes the point $Y_0^{*(k)}, k = 1, ..., K$.

Algorithm 2 is implemented with the input parameter `pred.alg = "Trajectory"`. Here, it could also be desired to choose $L \neq K$, implemented with `sample.length`. Default is $L = K$. In Hermann (2016b), the two presented prediction methods are compared to two commonly used sampling procedures. For the first one, `pred.alg = "simpleTrajectory"`, the point estimates, i.e. the posterior mean, are plugged into the model definition and `sample.length`, default is the number of posterior samples K, trajectories are drawn. For the second one, `pred.alg = "simpleBayesTrajectory"`, each of the K posterior samples is plugged into the model definition and one trajectory is drawn. If `sample.length` is specified and smaller than K, the first `sample.length` posterior samples are taken.

There is one additional feature of method `predict`. For comparison, a one step Euler prediction is implemented, which would be similar to Algorithm 1 with $M = 1$. In that case, sampling is not necessary and quantiles can be calculated directly. With `Euler.interval = TRUE`, for each of the time points, `level`/2 and 1-`level`/2 quantiles of the predictive distribution are calculated.

Mixed diffusion

We simulate a model similar to Example 1, but in a mixture model. Here, we can also assume the starting point to be random, i.e. $y_0(\phi) = \phi_2$ and $b(\phi, t, y) = \phi_1 y$. Now, for similar model as before, we choose $\mu = (2, 1)$ and $\Omega = \text{diag}(1, 0.04)$. We sample $J = 50$ series as follows.

```
J <- 50
mu <- c(2, 1)
Omega <- c(1, 0.04)
phi <- cbind(rnorm(J, mu[1], sqrt(Omega[1])), rnorm(J, mu[2], sqrt(Omega[2])))
gamma2 <- 1
y0.fun <- function(phi, t) phi[2]
b.fun <- function(phi, t, y) phi[1] * y
sT.fun <- function(t, y) y

model <- set.to.class("mixedDiffusion", y0.fun = y0.fun,
  b.fun = b.fun, sT.fun = sT.fun,
  parameter = list(phi = phi, mu = mu, Omega = Omega, gamma2 = gamma2))

t <- seq(0, 1, by = 0.01)
series <- simulate(model, seed = 123, t = t, plot.series = FALSE)
```

The variable `series` is a $J \times 101$ matrix. To estimate with the whole data set, this would also be input parameter for the method `estimate`. In this case, all series would be equally long, i.e. $n_1 = ... = n_J$. Otherwise, it is possible to include a list.

```
t.list <- lapply(1:(J-1), function(i) t)
t.list[[J]] <- t[1:50]
data.list <- lapply(1:(J-1), function(i) series[i,])
data.list[[J]] <- series[J, 1:50]
est <- estimate(model, t.list, data.list, 11000)
```

For the hierarchical models, there is an additional `style` option, called `"int.phi"` meaning the credible intervals for the random effects. The point in the middle of the intervals marks the posterior median. In addition, input parameter `par2plot`, which is a logical vector, contains `TRUE` for every parameter to be plotted and `FALSE` otherwise. For comparison with starting or prior values or, in a simulation study with the chosen values, `phi` itself can be included.

```
plot(est, style = "int.phi", phi = phi, par2plot = c(T, F))
legend("bottomleft", c("true value", "posterior median",
                       "95% credible interval"),
col = c(2, 1, 1), lty = c(-1, -1, 1), pch = c(20, 20, -1), cex = 0.7)
```

The result can be seen in Figure 47. Except one, all credible intervals include the simulated values.

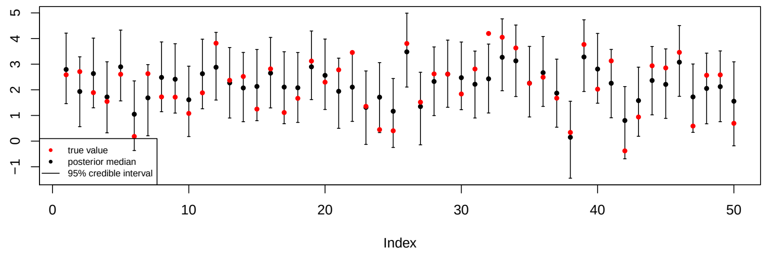


Figure 47: 95% credible intervals for the first component of the random effect in the mixed model, created with `plot(..., style = "int.phi")`.

Now, we want to make a prediction for the further development of the last, i.e. J, series.

```
pred <- predict(est, t = t[50:101], which.series = "current", ind.pred = J,
  b.fun.mat = function(phi, t, y) phi[,1]*y,
  burnIn = 1000, thinning = 2)
lines(t[50:101], series[J, 50:101], lty = 2)
```

We can see the result in Figure 48. The solid black line marks the part of the Jth series that is used for estimation. The dotted line is the simulation part that is skipped in the estimation and prediction is made for. The red lines are the 95% prediction intervals. Output of the method `predict` is a matrix $5\,000 \times 52$ matrix containing $5\,000$ samples of the predictive distribution in each of the 52 time points.

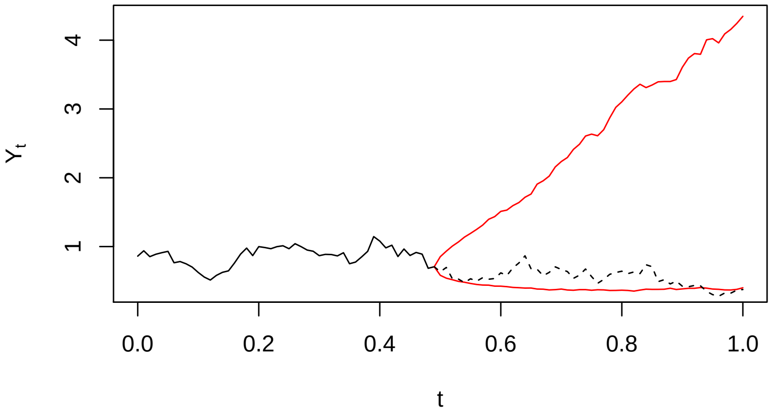


Figure 48: 95% prediction intervals for the last series of the mixed diffusion model.

Jump diffusion

We consider the following process.

Example 3
We will give an example by the process (2.1) *with* $b(\phi,t,y)=\phi$, $s(\gamma,t,y)=\gamma$, $h(\eta,t,y)=\eta y$ *and* $\Lambda_\xi(t)=\left(\frac{t}{\xi_2}\right)^{\xi_1}$. *It is approximated by*

$$Y_i = Y_{i-1} + \phi\,\Delta_i + \gamma\sqrt{\Delta_i}\,\zeta_i + \eta Y_{i-1}\Delta N_i, i = 1, ..., n,$$

with $Y_0 = y_0$, $\zeta_i \sim \mathcal{N}(0,1)$ *and* $\Delta N_i \sim \mathit{Pois}(\Lambda_\xi(t_i) - \Lambda_\xi(t_{i-1}))$. *We fix* $y_0 = 0.5, \phi = 0.2, \gamma = 0.5, \eta = 0.1, \xi = (2, 0.2)$ *and simulate random trajectories in* $t \in [0,1]$.

We translate the example to the language of the package. In the package, parameter η is called `theta`, all others are the same as in the text. This leads to

```
b.fun <- function(phi, t, y) phi
s.fun <- function(gamma2, t, y) sqrt(gamma2)
h.fun <- function(theta, t, y) theta * y
Lambda <- function(t, xi) (t/xi[2])^xi[1]
model <- set.to.class("jumpDiffusion",
   parameter = list(phi = 0.2, gamma2 = 0.25, theta = 0.1, xi = c(2, 0.2)),
   b.fun = b.fun, s.fun = s.fun, h.fun = h.fun, Lambda = Lambda)
t <- seq(0, 1, by = 0.01)
series <- simulate(model, seed = 123, t = t, y0 = 0.5, plot.series = FALSE)
```

In the jump diffusion model, there are two possibilities. In the first case, the Poisson process is not observed. This would be implemented by

```
est.hidden <- estimate(model, t, series$Y, 11000)
plot(est.hidden, par2plot = c(rep(F, 5), T), reduced = T)
lines(t, series$N, lwd = 2, col = 2)
legend("topleft", c("filtered", "simulated"), col = 1:2, lwd = 1:2)
```

The method `plot` has input paramter `par2plot`, which is a logical vector, which contains `TRUE` for every parameter to be plotted and `FALSE` otherwise. The order is (`phi`, `theta`, `gamma2`, `xi`, `N`). This leads to Figure 49, which compares the filtered trajectories of the Poisson process with the simulated series in red.

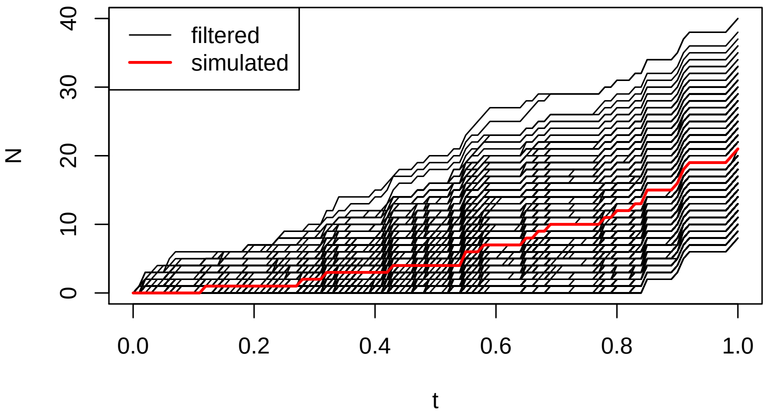


Figure 49: Example 3: Estimation of the latent variable $N_{(n)}$, i.e. the Poisson process vector, in red: the simulated series.

In the case of observed Poisson process variables, both observation $Y_{(n)}$ and $N_{(n)}$ have to be joint in a list with entries `Y` and `N`.

For the prediction of the jump diffusion with the presented methods in Section 2, it can be decided for each of the processes, the latent Poisson process as well as for the jump diffusion, if Algorithm 1 or 2 is desired. We have already seen the input parameter `pred.alg`, which is here used for the prediction of the jump diffusion. The additional input parameter `pred.alg.N` denotes the prediction algorithm for the Poisson process and can be chosen between `"Distribution"` and `"Trajectory"`. In the first case, pointwise sampling from the predictive distribution of the independent differences is conducted. In the second case, Algorithm 2 is run for the event times of the process. Default is `"Trajectory"`, since the combination `pred.alg = "Trajectory"` and `pred.alg.N = "Distribution"` does not make sense and is avoided to be accidentally chosen. The parameter `Lambda.mat` again is, similar to `b.fun.mat` in the diffusion model, to fasten the sampling algorithm.

9. Discussion and Outlook

In this thesis, a new Bayesian prediction approach for stochastic processes, general diffusions as well as jump diffusions, based on stochastic differential equations, has been developed. An overview of the considered models is given in Figure 50.

In the case of data following a continuous model, the diffusion, see Section 3, or the hidden diffusion model, see Section 4, are suitable. The hidden diffusion model has the advantage of different uncertainty structures, one of the diffusion model itself, where the variance grows over time, and one of the additional error, whose variance stays constant over time. This can be suitable in applications, where the crack growth process itself cannot be directly observed and a clear distinction between measuring error and random growth process is desired. On the other side, estimation for this model is time-consuming. A particle filtering algorithm has been applied for the latent diffusion process and joint with the estimations of the parameter in a particle Gibbs sampler. Here, in a first step, predictions are made for the latent diffusion process and, in a second step, these predictive variables are included in the predictive distribution of the observation variables. Both, estimation and prediction, have longer runtimes for the hidden diffusion model than for the observed diffusion. Hence, if a fast evaluation is necessary, the observed diffusion model is more appropriate with the disadvantage of a more rough description of the uncertainty structure.

In addition, we have seen both models in their hierarchical version. If several series are available, stemming from experiments on different individuals, the hierarchical is preferable to the single series model. But, if time plays a role in the evaluation, estimation is much faster for only one series instead of the whole hierarchical model for many series. Hence, if only a first, and fast, look on a model is desired, the single series is a good choice.

In the case of data containing jumps, one of the non-continuous models is suitable. If the counting process itself is of interest, as it is, for example, for the number of broken tension wires over time, the Poisson process presented in Section 5 is a good choice. In some cases, a deterministic intensity function does not describe the whole uncertainty structure, where a load sharing model can be more appropriate.

For modeling data following a piecewise continuous process containing jumps, like, for example, the concrete crack width process, a jump diffusion or a jump regression model can be suitable dependent on the underlying uncertainty structure. In the jump diffusion model, the variance of the Brownian motion part can grow over time or proportional to the current process or both, which makes the model very flexible. In the jump regression model, the curve is assumed to follow a deterministic curve, dependent on the NHPP, and the additive error has constant variance over time, whereby the extension with a time-dependent variance function is possible and already implemented in the package BaPreStoPro.

In both jump models, the Poisson process is latent and in most cases, this is not observed. For the jump diffusion, a filtering algorithm is already available and implemented in the package. For the jump regression model, this is future work.

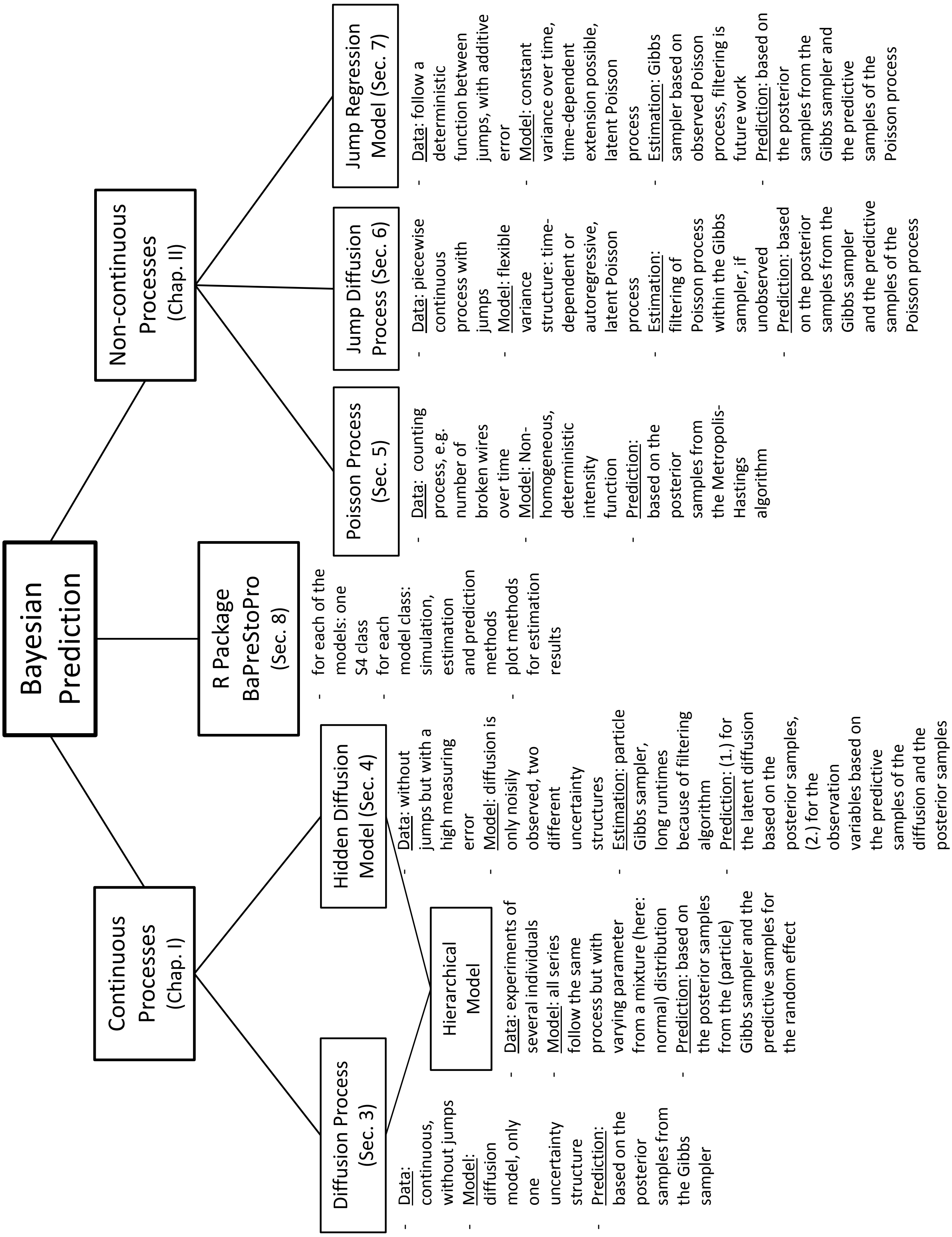

Bayesian Prediction
Continuous Processes (Chap. I)
R Package BaPreStoPro (Sec. 8)
Non-continuous Processes (Chap. II)
Diffusion Process (Sec. 3)
- Data: continuous, without jumps
- Model: diffusion model, only one uncertainty structure
- Prediction: based on the posterior samples from the Gibbs sampler
Hierarchical Model
- Data: experiments of several individuals
- Model: all series follow the same process but with varying parameter from a mixture (here: normal) distribution
- Prediction: based on the posterior samples from the (particle) Gibbs sampler and the predictive samples for the random effect
Hidden Diffusion Model (Sec. 4)
- Data: without jumps but with a high measuring error
- Model: diffusion is only noisily observed, two different uncertainty structures
- Estimation: particle Gibbs sampler, long runtimes because of filtering algorithm
- Prediction: (1.) for the latent diffusion based on the posterior samples, (2.) for the observation variables based on the predictive samples of the diffusion and the posterior samples
- for each of the models: one S4 class
- for each model class: simulation, estimation and prediction methods
- plot methods for estimation results
Poisson Process (Sec. 5)
- Data: counting process, e.g. number of broken wires over time
- Model: Non-homogeneous, deterministic intensity function
- Prediction: based on the posterior samples from the Metropolis-Hastings algorithm
Jump Diffusion Process (Sec. 6)
- Data: piecewise continuous process with jumps
- Model: flexible variance structure: time-dependent or autoregressive, latent Poisson process
- Estimation: filtering of Poisson process within the Gibbs sampler, if unobserved
- Prediction: based on the posterior samples from the Gibbs sampler and the predictive samples of the Poisson process
Jump Regression Model (Sec. 7)
- Data: follow a deterministic function between jumps, with additive error
- Model: constant variance over time, time-dependent extension possible, latent Poisson process
- Estimation: Gibbs sampler based on observed Poisson process, filtering is future work
- Prediction: based on the posterior samples from the Gibbs sampler and the predictive samples of the Poisson process

Figure 50: Overview of considered models in this thesis.

For prediction, the Markov property is a big advantage of the jump diffusion model. Starting at the last observed value, the prediction uncertainty is small in the beginning and grows over time. This effect is expected heuristically. But, of course, this advantage depends on the application. In experiments with a high measuring error, the jump regression model can be appropriate, because a high prediction uncertainty is justified even close to the last observed value.

Outlook

The presented prediction methods are based on the widely used Euler approximation for the SDE, which leads to a universally applicable approach. Of course, other, more sophisticated, approximation schemes might have a smaller error with respect to the true process, where the explicit solution might be unknown. It will be future work to investigate other approximation methods for a generalization of the presented methods. The Euler approximation is thought of a starting point, which is applicable to all of the considered models in this thesis.

In Section 6, we have seen the jump diffusion process based on the non-homogeneous Poisson process. For the presented crack width data, we assume to observe the underlying counting process based on sound measurements during the experiment. In this thesis, we presented a filtering algorithm for the latent Poisson process. The evaluation of one of the data set series raises the question, if on some event times, more than one wire breaks at the same time, which makes a large difference for estimation and even for prediction. A closer look at the problem will be future work. For the crack width data, there are currently 10 experiments made, most of them under different conditions. Here, a hierarchical jump diffusion model could be of interest, where covariates like the stress range will be considered.

In both jump models, the jump diffusion as well as the jump regression model, it could be of interest to exchange the latent Poisson process by the already mentioned self-exciting process. Here, a filtering algorithm for both models would have to be considered.

An R package has been developed, where all of the presented models are implemented as S4 class and simulation, estimation and prediction methods are available. By now, all codes are written in R. It will be future work to implement parts of it in C to make the functions, especially the filtering algorithms, much faster. Even from the methodical side, there are extensions thinkable. For example, by now, estimation is only based on the likelihood of the Euler approximated variables. Fuchs (2013), for example, presents a data augmentation approach, which makes it possible to get a smaller Euler approximation error. This would make the estimation part more sophisticated. For comparison, it would be also desirable to give the user the opportunity to include an own transition density, that might stem from an explicit solution of the process. By now, implementation is based on the drift, variance and height function given in (2.1). In addition, a filtering algorithm for the jump regression model would be a nice extension.

Acknowledgement

This work has been supported by the Collaborative Research Center “Statistical modeling of non-linear dynamic processes” (SFB 823) of the German Research Foundation (DFG) in project B5 “Statistical methods for damage processes under cyclic load”. We want to thank Professor Reinhard Maurer, Guido Heeke and Jens Heinrich for their experiments and providing us the data. In addition, we want to thank Professor B. M. Hillberry for his experiments and Eric J. Tuegel for providing us the data.

In addition, I want to thank Adeline Samson for her ideas and helpful discussions about the particle Gibbs sampler and the productive work on the package for mixed diffusion models. In particular, I want to thank Fabrizio Ruggeri for the opportunity to work with him on the filtering of the latent Poisson process in the jump diffusion model with application to wear in cylinder liners.

Persönliche Danksagung

Allem voran möchte ich mich hier bei Christoph Falkenau bedanken, ohne den ich diese Arbeit nie geschrieben hätte. In den fünf Jahren Promotionszeit gab es Höhen und vor allem Tiefen, in denen es immer Christoph war, der ein offenes Ohr und ein paar hilfreiche Worte für mich hatte.

Ich möchte hier auch meinen beiden Betreuerinnen Christine Müller und Katja Ickstadt danken für die Freiheit, die ich von Anfang an hatte. Auch wenn ursprünglich thematisch etwas anderes geplant war, konnte ich eigenverantwortlich meine eigenen Ideen verfolgen und umsetzen. Auch die beiden Forschungsaufenthalte wurden sofort bereitwillig von euch unterstützt, die mich jeweils sehr in meiner Arbeit weiter gebracht haben.

Ganz besonders möchte ich meinem Mann Eric danken für die Unterstützung und den Rückhalt gerade in der Endphase der Promotion. Dank dir gibt es inzwischen in meinem Leben so viel mehr als nur die Wissenschaft.

References

Andrieu, C., A. Doucet, and R. Holenstein (2009). “Particle Markov Chain Monte Carlo for Efficient Numerical Simulation”. *Monte Carlo and quasi-Monte Carlo methods 2008*. Berlin Heidelberg: Springer, pp. 45–60.

Andrieu, C., A. Doucet, and R. Holenstein (2010). “Particle Markov Chain Monte Carlo Methods”. *Journal of the Royal Statistical Society B* 72, pp. 269–342.

Barber, D., A. T. Cemgil, and S. Chiappa (2011). *Bayesian Time Series Models*. New York: Cambridge University Press.

Beskos, A., O. Papaspiliopoulos, G. O. Roberts, and P. Fearnhead (2006). “Exact and Computationally Efficient Likelihood-Based Estimation for Discretely Observed Diffusion Processes”. *Journal of the Royal Statistical Society B* 68, pp. 333–382.

Blanke, D. and D. Bosq (2015). “Bayesian Prediction for Stochastic Processes: Theory and Applications”. *Sankhyā: The Indian Journal of Statistics* 77, pp. 79–105.

Bruti-Liberati, N. and E. Platen (2007). “Strong Approximations of Stochastic Differential Equations with Jumps”. *Journal of Computational and Applied Mathematics* 205, pp. 982–1001.

Carlin, B. P. and T. A. Louis (2009). *Bayesian Methods for Data Analysis*. Boca Raton: Chapman & Hall.

Carvalho, C. M., M. S. Johannes, H. F. Lopes, and N. G. Polson (2010). “Particle Learning and Smoothing”. *Statistical Science* 25, pp. 88–106.

Chiquet, J., N. Limnios, and M. Eid (2009). “Piecewise Deterministic Markov Processes Applied to Fatigue Crack Growth”. *Journal of Statistical Planning and Inference* 139, pp. 1657–1667.

Cont, R. and P. Tankov (2004). *Financial modelling with jump processes*. London: CRC Press.

Crowder, M. J., A. C. Kimber, R. L. Smith, and T. J. Sweeting (1994). *Statistical Analysis of Reliability Data*. London: Chapman & Hall.

Del Moral, P., A. Doucet, and A. Jasra (2006). “Sequential Monte Carlo Samplers”. *Journal of the Royal Statistical Society B* 68, pp. 411–436.

Devroye, L. (1986). *Non-Uniform Random Variate Generation*. New York: Springer.

Dion, C., S. Hermann, and A. Samson (2016a). “Mixedsde: an R Package to Fit Mixed Stochastic Differential Equations”. hal-01305574.

Dion, C., A. Samson, and S. Hermann (2016b). *mixedsde: Estimation Methods for Stochastic Differential Mixed Effects Models*. R package version 1.0. URL: `http://CRAN.R-project.org/package=mixedsde`.

Donnet, S., J.-L. Foulley, and A. Samson (2010). “Bayesian Analysis of Growth Curves Using Mixture Models Defined by Stochastic Differential Equations”. *Biometrics* 66, pp. 733–741.

Doucet, A., S. Godsill, and C. Andrieu (2000). “On Sequential Monte Carlo Sampling Methods for Bayesian Filtering”. *Statistics and Computing* 10, pp. 197–208.

Fuchs, C. (2013). *Inference for Diffusion Processes.* Berlin Heidelberg: Springer.

Gneiting, T. and A. E. Raftery (2007). “Strictly Proper Scoring Rules, Prediction and Estimation”. *Journal of the American Statistical Association* 102, pp. 359–378.

Gunkel, C., A. Stepper, A. C. Müller, and Ch. H. Müller (2012). “Micro Crack Detection with Dijkstra’s Shortest Path Algorithm”. *Machine Vision and Applications* 23, pp. 589–601.

Heeke, G. (2016). “Untersuchungen zur Ermüdungsfestigkeit von Betonstahl und Spannstahl im Zeit-und Dauerfestigkeitsbereich mit sehr hohen Lastwechselzahlen”. PhD thesis. TU Dortmund, Fachbereich Architektur und Bauingenieurwesen.

Heeke, G., S. Hermann, R. Maurer, K. Ickstadt, and Ch. H. Müller (2015). “Stochastic Modeling and Statistical Analysis of Fatigue Tests on Prestressed Concrete Beams under Cyclic Loadings”. *SFB 823 Discussion Paper 25/2015.*

Hermann, S. (2016a). *BaPreStoPro: Bayesian Prediction of Stochastic Processes.* R package version 0.1. URL: `http://CRAN.R-project.org/package=BaPreStoPro`.

Hermann, S. (2016b). “Bayesian Prediction for Stochastic Processes based on the Euler Approximation Scheme”. *SFB 823 Discussion Paper 27/2016.*

Hermann, S. and F. Ruggeri (2016). “Modelling Wear in Cylinder Liners”. *Quality and Reliability Engineering International.* DOI: `10.1002/qre.2061`.

Hermann, S., K. Ickstadt, and Ch. H. Müller (2016a). “Bayesian Prediction for a Jump Diffusion Process with Application to Crack Growth in Fatigue Experiments”. *Reliability Engineering & System Safety.* DOI: `10.1016/j.ress.2016.08.012`.

Hermann, S., K. Ickstadt, and Ch. H. Müller (2016b). “Bayesian Prediction of Crack Growth Based on a Hierarchical Diffusion Model”. *Applied Stochastic Models in Business and Industry* 32, pp. 494–510.

Iacus, S. M. (2008). *Simulation and Inference for Stochastic Differential Equations.* New York: Springer.

Ikeda, N. and S. Watanabe (1989). *Stochastic Differential Equations and Diffusion Processes.* 2nd ed. Amsterdam, Oxford, New York: North-Holland.

Johannes, M. S., N. G. Polson, and J. R. Stroud (2009). “Optimal Filtering of Jump Diffusions: Extracting Latent States from Asset Prices”. *The Review of Financial Studies* 22, pp. 2759–2799.

Jokiel-Rokita, A., D. Lazar, and R. Magiera (2014). “Bayesian Prediction in Doubly Stochastic Poisson Process”. *Metrika* 77, pp. 1023–1039.

Klebaner, F. C. (2005). *Stochastic Calculus with Applications.* London: Imperial College Press.

Kluge, W. (2005). “Time-inhomogeneous Lévy processes in interest rate and credit risk models.” PhD thesis. University of Freiburg, 4. Jg., Nr. 5, S. 6.

Kuo, L. and T. Y. Yang (1996). “Bayesian Computation for Nonhomogeneous Poisson Processes in Software Reliability”. *Journal of the American Statistical Association* 91, pp. 763–773.

Kvam, P. H and J.-C. Lu (2007). “Load-Sharing Models”. *Encyclopedia of Statistics in Quality and Reliability.*

Liu, J. S. and R. Chen (1998). “Sequential Monte Carlo Methods for Dynamic Systems”. *American Statistical Association* 93, pp. 1032–1044.

Maurer, R. and G. Heeke (2010). *Ermüdungsfestigkeit von Spannstählen aus einer Älteren Spannbetonbrücke.* Tech. rep. 76. Dortmund: Landesbetrieb Straßenbau NRW, TU Dortmund.

Meeker, W. Q. and L. A. Escobar (1998). *Statistical Methods for Reliability Data.* New York: John Wiley & Sons.

Merton, R. C. (1976). “Option Pricing When Underlying Stock Returns are Discontinuous”. *Journal of Financial Economics* 3, pp. 125–144.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). “Equation of State Calculations by Fast Computing Machines”. *The Journal of Chemical Physics* 21, pp. 1087 –1092.

Müller, C. H., S. Szugat, and R. Maurer (2016). “Simulation Free Prediction Intervals for a State Dependent Failure Process Using Accelerated Lifetime Experiments”. *SFB 823 Discussion Paper 45/2016.*

Nakagawa, T. (2007). *Shock and Damage Models in Reliability Theory.* London: Springer.

Øksendal, B. (2003). *Stochastic Differential Equations: An Introduction with Applications.* Heidelberg New York: Springer.

Øksendal, B. and A. Sulem (2005). *Applied Stochastic Control of Jump Diffusions.* Berlin Heidelberg: Springer.

Phillips, P. C. B. (1973). “The Problem of Identification in Finite Parameter Continuous Time Models”. *Journal of Econometrics* 1, pp. 351–362.

Pievatolo, A. and F. Ruggeri (2004). “Bayesian Reliability Analysis of Complex Repairable Systems”. *Applied Stochastic Models in Business and Industry* 20, pp. 253–264.

Platen, E. and N. Bruti-Liberati (2010). *Numerical Solution of Stochastic Differential Equations with Jumps in Finance.* First. Berlin Heidelberg: Springer.

Plummer, M., N. Best, K. Cowles, and K. Vines (2006). “CODA: Convergence Diagnosis and Output Analysis for MCMC”. *R News* 6, pp. 7–11. URL: `http://CRAN.R-project.org/doc/Rnews/`.

Protter, P. E. (2005). *Stochastic Integration and Differential Equations.* Berlin Heidelberg: Springer.

R Core Team (2015). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing. Vienna, Austria. URL: `https://www.R-project.org/`.

Ríos Insua, D., F. Ruggeri, and M. P. Wiper (2012). *Bayesian Analysis of Stochastic Process Models.* United Kingdom: John Wiley & Sons, Ltd.

Robert, C. P. and G. Casella (2004). *Monte Carlo Statistical Methods.* New York: Springer.

Robinson, M. E. and M. J. Crowder (2000). “Bayesian Methods for a Growth-Curve Degradation Model with Repeated Measures”. *Lifetime Data Analysis* 6, pp. 357–374.

Rosenthal, J. S. (2011). “Optimal Proposal Distributions and Adaptive MCMC”. *Handbook of Markov Chain Monte Carlo*, pp. 93–112.

Shimizu, Y. and N. Yoshida (2006). “Estimation of Parameters for Diffusion Processes with Jumps from Discrete Observations”. *Statistical Inference for Stochastic Processes* 9, pp. 227–277.

Sobczyk, K. and B.F. Spencer (1992). *Random Fatigue: From Data to Theory.* London: Academic Press Limited.

Sørensen, H. (2004). “Parametric Inference for Diffusion Processes Observed at Discrete Points in Time: a Survey”. *International Statistical Review* 72, pp. 337–354.

Szugat, S., J. Heinrich, R. Maurer, and Ch.H. Müller (2016). “Prediction Intervals for the Failure Time of Prestressed Concrete Beams”. *Advances in Materials Science and Engineering.* DOI: `10.1155/2016/9605450`.

Vidoni, P. (2004). “Improved Prediction Intervals for Stochastic Process Models”. *Journal of Times Series Analysis* 25, pp. 137–154.

Virkler, D. A., B. M. Hillberry, and P. K. Goel (1979). “The Statistical Nature of Fatigue Crack Propagation”. *Journal of Engineering Materials and Technology* 101, pp. 148–153.

Weinberg, J., L. D. Brown, and J. R. Stroud (2007). “Bayesian Forecasting of an Inhomogeneous Poisson Process with Application to Call Center Data”. *Journal of the American Statistical Association* 102, pp. 1185–1198.

Yu, J.-W., G.-L. Tian, and M.-L. Tang (2007). “Predictive Analyses for Nonhomogeneous Poisson Processes with Power Law Using Bayesian Approach”. *Computational Statistics & Data Analysis* 51, pp. 4254–4268.

Yuan, T. and Y. Ji (2015). “A Hierarchical Bayesian Degradation Model for Heterogeneous Data”. *IEEE Transactions on Reliability* 64, pp. 63–70.

Yuan, X.-X., D. Mao, and M. D. Pandey (2009). “A Bayesian Approach to Modeling and Predicting Pitting Flaws in Steam Generator Tubes”. *Reliability Engineering and System Safety* 94, pp. 1838–1847.

A. Notations

t	time in $[0,T]$
$\{Y_t, t\in[0,T]\}$	stochastic process
$\{W_t, t\in[0,T]\}$	Brownian motion
$\{N_t, t\in[0,T]\}$	Poisson process
$b(\phi,t,y)$	drift function
$s(\gamma,t,y)$	variance function
$h(\eta,t,y)$	jump height function
$\Lambda_\xi(t)$	cumulative intensity function
$\lambda_\xi(t)$	intensity function, derivative of $\Lambda_\xi(t)$
θ	parameter vector, e.g., (ϕ,γ^2,η,ξ), resp. dummy for arbitrary parameter
n	number of observations
$n_j,\ j=1,...,J$	in the hierarchical model: number of observations in series j
J	in the hierarchical model: number of series, e.g. test specimen
I	number of jumps
$y_0=y_0(\phi)$	starting point (almost surely) of the stochastic process $\{Y_t, t\in[0,T]\}$
$Y_{(n)}=(Y_0,...,Y_n)$	vector of Euler approximated variables
$Y_{(n_j,j)}$	in the hierarchical model: the jth observation vector
$Y_{(n\cdot,J)}$	in the hierarchical model: all the observations $\{Y_{ij}\}_{i=0,...,n_j,j=1,...,J}$
$t_{(n)}=(t_0,...,t_n)$	vector of time points
$t_{ij}, i=1,...,n_j; j=1,...,J$	observation time points in the hierarchical models
$\Delta_i=t_i-t_{i-1}, i=1,...,n$	observation time distances
$\Delta_{ij}=t_{ij}-t_{i-1,j}$	observation time distances in the hierarchical models, $i=1,...,n_j; j=1,...,J$
$T_1,...,T_I$	event times of the counting process
$q(\theta^*\|\theta)$	proposal density of MH algorithm
$p(\theta\|Y_{(n)})$	posterior density of θ
$p(Y_{m+1}\|Y_m,\theta)$	transition density of Markov process $\{Y_m, m=0,1,2,...\}$
$m_\theta, V_\theta, a_\theta, b_\theta$	prior parameters of the normal, resp. the IG distribution of θ
$\theta_k^*, k=1,...,K$	posterior samples
t^*, Y^*	Y^* variable to predict in t^*
$\tau_0\in\{t_0,t_n\}, \tau_1,..,\tau_M=t^*$	partition of interval $[t_0,t^*]$, resp. $[t_n,t^*]$
Δ^*	time difference $\frac{t^*-\tau_0}{M}$
$Y_0^*\in\{y_0,Y_n\}, Y_1^*,...,Y_M^*$	prediction variables in $\tau_0,\tau_1,..,\tau_M$

B. Further Details

Calculation to Section 7

Beginning with the function

$$w(t) = \frac{(1 - k(t)) \cdot (\Delta\sigma(t))^2 \cdot A_P(t)}{0.72 \cdot \pi \cdot f_{ctm} \cdot E_P \cdot \sqrt{A_P(t)}}$$

with

$$\begin{aligned}
k(t) &= k_l + (k_s - k_l) \cdot e^{t \cdot c} \\
A_P(t) &= A_P \cdot \left(1 - \frac{N_t}{35}\right) = A_p \cdot h(N_t)^{-1} \\
\Delta\sigma(t) &= \sigma_2 \cdot \frac{A_P}{A_P(t)} - \sigma_1 = \sigma_2 \cdot \frac{1}{1 - \frac{N_t}{35}} - \sigma_1 = \sigma_2 \cdot h(N_t) - \sigma_1
\end{aligned}$$

with $k_l, k_s, c, A_P, \sigma_1, \sigma_2, f_{ctm}, E_P$ constants and $h(x) = \frac{1}{1 - \frac{x}{35}}$. We abbreviate $\nu = 0.72 \cdot \pi \cdot f_{ctm} \cdot E_p$ and get

$$\begin{aligned}
w(t) = &\frac{(1 - k(t)) \cdot (\Delta\sigma(t))^2 \cdot A_p(t)}{0.72 \cdot \pi \cdot f_{ctm} \cdot E_p \cdot \sqrt{A_p(t)}} \\
= &\frac{1}{\nu} \cdot (1 - k(t)) \cdot (\Delta\sigma(t))^2 \cdot \sqrt{A_p(t)} \\
= &\frac{1}{\nu} \cdot (1 - k(t)) \cdot (\sigma_2 \cdot h(N_t) - \sigma_1)^2 \cdot \sqrt{A_p \cdot h(N_t)^{-1}} \\
= &\frac{\sqrt{A_p}}{\nu \cdot \sqrt{h(N_t)}} \cdot (1 - k(t)) \cdot (\sigma_2 \cdot h(C_t) - \sigma_1)^2 \\
= &\frac{\sqrt{A_p}}{\nu \cdot \sqrt{h(N_t)}} \cdot (1 - k_l - (k_s - k_l) \cdot e^{t \cdot c}) \cdot \sigma_2^2 \cdot \left(h(N_t) - \frac{\sigma_1}{\sigma_2}\right)^2 \\
= &\frac{\sqrt{A_p} \cdot \sigma_2^2}{\nu} \cdot \Big((1 - k_l) - (k_s - k_l) \cdot e^{t \cdot c}\Big) \cdot \frac{1}{\sqrt{h(N_t)}} \cdot \left(h(N_t) - \frac{\sigma_1}{\sigma_2}\right)^2 \\
= &\left(\phi_1 - \phi_2 \cdot \exp(-t \cdot \phi_3)\right) \cdot \tfrac{1}{\sqrt{h(N_t)}} \left(h(N_t) - \phi_4\right)^2
\end{aligned}$$

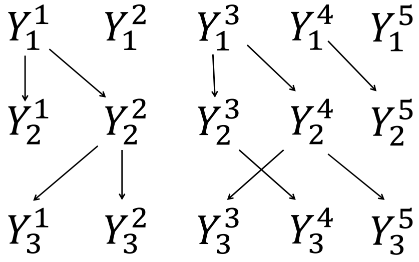
with

- $\phi_1 = \frac{(1-k_l) \cdot \sqrt{A_p} \cdot \sigma_2^2}{0{,}72 \cdot \pi \cdot f_{ctm} \cdot E_p}$,
- $\phi_2 = \frac{(k_s-k_l) \cdot \sqrt{A_p} \cdot \sigma_2^2}{0{,}72 \cdot \pi \cdot f_{ctm} \cdot E_p}$,
- $\phi_3 = -c$,
- $\phi_4 = \frac{\sigma_1}{\sigma_2}$.

Conditional SMC

We here recall to the SMC algorithm in Section 4 and add the changes for the conditional SMC. Remember, we draw $j \sim p_{1:L}(\cdot|W_n)$ and set $Y_{(n)} = (Y_0^j, ..., Y_n^j)$ with the weights and the particles from the SMC before.

We introduce the notion of the ancestral lineage $B_{(n)}^l := (B_0^l, ..., B_n^l), l = 1, ..., L,$. It is recursively defined by $B_n^l = l$ and $B_i^l := A_i^{B_{i+1}^l}, i = n-1, ..., 0$. In Figure 51, we see an illustration of the SMC with $L = 5$ particles and $n = 3$ iteration steps.



$$A_1 = (A_1^1, ..., A_1^5) = (1, 1, 3, 3, 4)$$

$$A_2 = (A_2^1, ..., A_2^5) = (2, 2, 4, 3, 4)$$

Figure 51: Illustration of SMC; $B_{(3)}^1 = (1, 2, 1), B_{(3)}^2 = (1, 2, 2), B_{(3)}^3 = (3, 4, 3), B_{(3)}^4 = (3, 3, 4), B_{(3)}^5 = (3, 4, 5)$.

We denote with $\mathcal{M}(L-1, (p_1, ..., p_L))$ the multinomial distribution with parameter $L-1$ and probability vector $(p_1, ..., p_L)$. For the CSMC in the next iteration of the Gibbs sampler, set $Y_0^{B_0^j} = Y_0$ and replace (i) and (iii) in the description of the SMC by

(ia) – Sample offspring variables $O_{i-1}^l \sim \mathcal{M}(L-1, W_{i-1})$ and set $o := O_{i-1, B_{i-1}^j}^l + 1$, with $O_{i-1, B_{i-1}^j}^l$ the B_{i-1}^jth component

– sample $h_1, ..., h_o \sim \mathcal{U}\{1, ..., B_{i-1}^j - 1, B_{i-1}^j + 1, ..., L\}$

– $A_{i-1}^l = B_{i-1}^j$ for $l \in \{B_i^j, h_1, ..., h_o\}$
$A_{i-1}^l \sim p_{1,...,B_i^j-1,B_i^j+1,...,L}(\cdot|W_{i-1}^{-B_i^j})$ for $l \in \{1, ..., L\} \backslash \{B_i^j, h_1, ..., h_o\}$,

whereby $W_{i-1}^{-B_i^j} = (W_{i-1}^1, ..., W_{i-1}^{B_i^j-1}, W_{i-1}^{B_i^j+1}, ..., W_{i-1}^L)$.

(iiia) Set $Y_i^{B_i^j} = Y_i$ and draw $Y_i^l \sim p(Y_i|Y_{i-1}^l, Z_i, \theta),\ l = 1, ..., B_i^j - 1, B_i^j + 1, ..., L$.

See Andrieu et al. (2009), and remark the errata at the end.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Dissertation selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Dissertation ist bisher keiner anderen Fakultät vorgelegt worden. Ich erkläre, dass ich bisher kein Promotionsverfahren erfolglos beendet habe und dass keine Aberkennung eines bereits erworbenen Doktorgrades vorliegt.

Simone Hermann