Imperial College London
Department of Mathematics

Bayesian Inference For High Dimensional Lorenz '96 Equations Using Sequential Monte Carlo

Jesper Cedergren

CID: 01284936

Supervised by Nikolas Kantas

25th September 2017

Submitted in partial fulfilment of the requirements for the MSc in Statistics of Imperial College London

The work contained in this thesis is my own work unless otherwise stated.

Signed: Date:

Abstract

The particle filter is widely applicable to nonlinear dynamical systems. Inference in nonlinear systems of a high-dimensional nature is challenging due to the curse of dimensionality, the likelihood is highly peaked and the weights of the particle filter will vary widely. Furthermore, the performance of the particle filter depends on the stability of the underlying signal which needs to be observed accurately if the signal is chaotic. In this thesis we evaluate the particle filter's tracking of simulated data from the Lorenz '96 equations. We repeat this for different observations schemes under different regimes. Using Bayesian inference and Sequntial Monte Carlo (SMC) methods, we estimate the model parameters for different priors, likelihoods, observations schemes and evaluate the performance of these methods. We also infer the model parameters for high-dimensional systems using adaptive tempering to tackle the curse of dimensionality. We find that the tracking performance of the particle filter deteriorates as the system is observed more sparsly and that accurate observations are crucial for the performance. For an eight-dimensional Lorenz '96 model the SMC^2-algorithm is robust with respect to priors and likelihood evaluations. When observing the system less accurately, the posterior distribution converges at a slower rate. Bridging the posterior distribution in the algorithm with adaptive tempering mitigates the variance inflation of the weights. Together with an efficient transition kernel, tempering prevents the ensemble of particles from degenerating in systems of Lorenz '96 equations up to 96 dimensions.

Acknowledgements

I sincerely would like to thank my supervisor Nikolas Kantas for his contribution in terms of guidance and discussions throughout the term. I would also like to thank my fellow students for discussing and elaborating ideas, Xiaolin Jiang in particular who was always keen on discussing C++ programming which I did not have any experence with before this project. I would like to thank my family for their support over all the years leading up to the completion of my degree. I especially want to address a big thank-you to my brother Robert for all the encouragement he has given me and for putting up with me during intense work periods.

Table of Contents

1. Introduction

Chaotic systems are dynamical systems where small changes of the initial conditions can lead to very different future behavior and divergence in the difference of trajectories. Even though the nature of the system is deterministic it is chaotic since small initial changes makes it unpredictable. After Edward Lorenz discovered chaotic behavior in the mathematical modeling of weather systems [1] this concept later became known as the butterfly effect. Lorenz used the metaphor of whether two situations initially differing only by the influence of a single butterfly eventually would differ by as much as the presence of a tornado.

Being able to infer states and parameters of a turbulent system is in particular useful in weather forecasting [1]. Firstly, this task is challenging because of the unstable underlying signal which requires the system to be observed accurately. Secondly, the nature of a chaotic system is typically high-dimensional in which it is hard to have accurate numerical methods.

Inference about a dynamical system can be made using data assimilation. Data assimilation involves incorporating information from observations of a system state and information about the model dynamics in order to estimate the actual system states or certain model parameters. In a typical setting the observations arrive sequentially in time and some prior knowledge is available. This allows us to put the data assimilaion in a Bayesian framework and update the posterior distribution of interest sequentially in time using Bayes rule. For this purpose Sequential Monte Carlo methods (SMC) serve well. SMC methods are simulation based methods whose applicability is not contiditioned on the availability of analytical solutions. This makes them very flexible and applicable to non-Gaussian, nonlinear high-dimensional problems. The latter impose a challenge since the likelihood is highly peaked in high dimensions which results in large variance of the weights. The performance of the SMC algorithms depends on the stability of the underlying signal. Due the induced chaotic behavior this makes the Lorenz '96 equations baffling. These challenges calls for for advanced methods. In previous work [2] applied SMC methods for state prediction and the SMC^2-algorithm for parameter estimation in an eight-dimensional Lorenz '96 model. The SMC^2-algorithm is an SMC method in the parameter dimension with an attached particle filter in the state dimension. As in standard SMC the particles are resmapled in order to prevent weight degeneracy. In high dimensions the parameter weights typically degenerate even when resampling is used due to the mentioned highly peaked likelihood. To mitigate this the ensemble of particles needed grows exponentially with the variance of the log likelihood which quickly becomes computationally expensive, [3, 4]. The weight degeneracy can instead be reduced by introducing artificial intermediate steps using ideas from Annealed Importance Sampling to fix the effective sample size, [5, 6, 7, 8, 9].

The purpose of this thesis is to perform data assimilation for a chaotic system using SMC methods. In Chapter 2 a review of the necessary theory and algorithms is presented. In Chapter 3 we study the Lorenz '96 model and its behavior in different regimes. In Chapter 4, we first investigate the particle filter's performance to simulated trajectories from the model. In particular we compare the performance for different observation schemes in different regimes. The performance is measured in terms of tracking of actual system states and with respect to the variance of the normalising constant. In Section 2 in Chapter 4 we implement the SMC^2-algorithm to estimate the model parameters. We investigate the algorithm's sensitivity to different priors, likelihood evaluation and observation schemes. We also investigate how the SMC^2-algorithm perform when artificial intermediate distributions are introduced for high-dimensional problems. Finally, Chapter 5 contains conclusions and discussion of the result and ideas for further extensions of the thesis.

2. Literature Review

This chapter provides the theoretical framework for the SMC methods we use for the data assimilation in Chapter 4, starting with basic Monte Carlo and SMC methods leading up to the SMC^2-algorithm.

2.1. Definitions and notation

Some basic definitions that we will use throughout the thesis applied to the SMC framework is presented in this section.

2.1.1. Hidden Markov Model

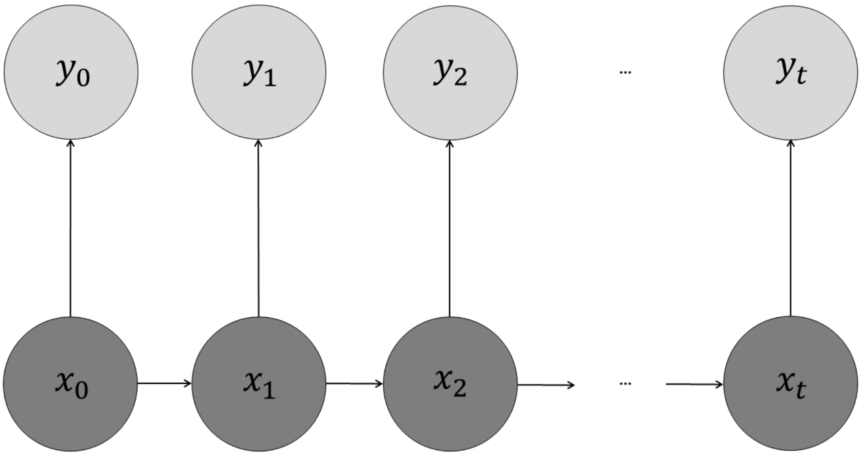

Figure 2.1.: Hidden Markov model

A state-space model, or equivalently a hidden markov model, consists of a hidden or latent process and an observed process where the latent process $\{X_t\}_{t\geq 0}$ is a Markov process with transition density

$$p(x_t|x_{0:t-1},\theta) = p(x_t|x_{t-1},\theta) = f_\theta(x_t|x_{t-1}), \quad p(x_0|\theta) = \eta_\theta(x_0).$$

The observed process $\{Y_t\}_{t\geq 0}$ has the conditional likelihood density

$$p(y_t|x_{0:t},\theta) = p(y_t|x_t,\theta) = g_\theta(y_t|x_t).$$

A general d-dimensional representation of a state-space model is given by

$$\begin{aligned}\mathbf{x}_t =& G(\mathbf{x}_{t-1},\mathbf{v}_t), \quad \mathbf{v}_t \sim \mathcal{V}_d,\\ \mathbf{y}_t =& H(\mathbf{x}_t,\mathbf{w}_t), \quad \mathbf{w}_t \sim \mathcal{W}_d,\end{aligned}$$

where G and H resepctively describes the dynamics of the specific model and the observation scheme. The noise components $\mathbf{v}_t$ and $\mathbf{w}_t$ are individually referred to as system noise and observation noise which are distributed according to some arbitrary d-dimensional distributions $\mathcal{V}_d$ and $\mathcal{W}_d$.

2.1.2. Filtering

The joint filtering distribution and its density for the whole path $X_{0:t}|Y_{0:t}$ is denoted

$$\pi_{0:t}(dx_{0:t}) = P(X_{0:t} \in dx_{0:t}|Y_{0:t}), \quad \pi_{0:t} = p_\theta(x_{0:t}|y_{0:t}).$$

Its marginal filtering distribution and density is denoted

$$\pi_t(dx_t) = P(X_t \in dx_t|Y_{0:t}), \quad \pi_t = p_\theta(x_t|y_{0:t}),$$

where $p_\theta(x_t|y_{0:t}) \propto \int p_\theta(x_{0:t}|y_{0:t})dx_{0:t-1}$.

2.1.3. Bayesian filtering recursions

The idea behind Bayesian filtering is to recursively update the posterior distribution in time as the observations become available using Bayes rule. Inference about $x_{0:t}$ given the observations $y_{0:t}$ may be based on the posterior density

$$p_\theta(x_{0:t}|y_{0:t}) = \frac{p_\theta(x_{0:t}, y_{0:t})}{p_\theta(y_{0:t})}, \tag{2.1}$$

where the joint density is

$$p_\theta(x_{0:t}, y_{0:t}) = \eta_\theta(x_0)\prod_{k=1}^{t} f_\theta(x_k|x_{k-1})\prod_{k=1}^{t} g_\theta(y_k|x_k).$$

The marginal density referred to as the normalising constant Z or evidence is

$$p_\theta(y_{0:t}) = \int p_\theta(x_{0:t}, y_{0:t})dx_{0:t}.$$

Using that the marginal density can be decomposed and written

$$p_\theta(y_{0:t}) = p(y_0)\prod_{k=1}^{t} p_\theta(y_t|y_{0:t-1}) = p_\theta(y_{0:t-1})p_\theta(y_t|y_{0:t-1}),$$

we can rewrite the joint filtering density in 2.1 as

$$\begin{aligned} p_\theta(x_{0:t}|y_{0:t}) &= \frac{p_\theta(x_{0:t-1}, y_{0:t-1})f_\theta(x_t|x_{t-1})g_\theta(y_t|x_t)}{p_\theta(y_{0:t-1})p_\theta(y_t|y_{0:t-1})} \\ &= p_\theta(x_{0:t-1}|y_{0:t-1})\frac{f_\theta(x_t|x_{t-1})g_\theta(y_t|x_t)}{p_\theta(y_t|y_{0:t-1})}. \end{aligned} \tag{2.2}$$

2.2. Basic Monte Carlo

For a more detailed introduction to the theory in this section, see [10, 11]. Consider the distribution π on $\mathcal{X}$ with resepct to dx which can be written

$$\pi(x) = \frac{\gamma(x)}{Z} \tag{2.3}$$

where Z is a normalising constant and let $\varphi : \mathcal{X} \to \mathbb{R}^{n_x}$ be a bounded measurable function. The expected value of φ with respect to π is

$$E_\pi[\varphi(X)] = \int_{\mathcal{X}} \varphi(x)\pi(dx). \tag{2.4}$$

2.2.1. Perfect Monte Carlo

Assuming we can obtain N independent and identically distributed samples from π

$$x^i \overset{\text{iid}}{\sim} \pi(\cdot), \quad i = 1, ..., N,$$

we can then approximate 2.4 by

$$\widehat{E_\pi[\varphi(X)]} = \frac{1}{N} \sum_{i=1}^{N} \varphi(x^i).$$

Referring to the samples as particles, the particle approximation of π is

$$\hat{\pi}(dx) = \frac{1}{N} \sum_{i=1}^{N} \delta_{x^i}(dx).$$

where $\delta_{x^i}(dx)$ is the Dirac delta function such that $\delta_{x^i}(\cdot) = \mathbb{I}(x^i \in \cdot)$.

2.2.2. Importance Sampling

It might either be hard to sample from π or areas for large values of $\varphi(x)$ might not coincide with areas of high density regions of $\pi(x)$. This could result in high variance when approximating π using perfect Monte Carlo. Assuming we can obtain N i.i.d. samples from an importance distribution q

$$x^i \overset{\text{iid}}{\sim} q(\cdot), \quad i = 1, ..., N, \tag{2.5}$$

for which $\varphi(x)\pi(x) > 0$ implies that $q(x) > 0$ the variance in the approximations can be reduced. We can write 2.3 as

$$\pi(x) = \frac{w(x)q(x)}{Z},$$

where $w(x) = \dfrac{\gamma(x)}{q(x)}$ weights the samples from q in order to compensate for the fact that the samples are not from π. Then 2.4 can be approximated with

$$\widehat{E_\pi[\varphi(X)]} = \sum_{i=1}^{N} W^i \varphi(x^i),$$

where $W^i = \frac{w(x^i)}{\sum_{i=1}^{N} w(x^i)}$. The particle approximation of π is now

$$\hat{\pi}(dx) = \sum_{i=1}^{N} W^i \delta_{x^i}(dx),$$

where $\hat{q}(dx) = \dfrac{1}{N} \displaystyle\sum_{i=1}^{N} \delta_{x^i}(dx)$ is the perfect Monte Carlo particle approximation of q. The normalising constant is approximated by

$$\hat{Z} = \int \frac{\gamma(x)}{q(x)} \hat{q}(dx) = \frac{1}{N} \sum_{i=1}^{N} \frac{\gamma(x^i)}{q(x^i)}.$$

It can be shown that the estimate of Z is unbiased with respect to the sampling distribution, $\mathrm{E}_q(\hat{Z}) = Z$. When choosing q in practice one should note that minimising the variance for $\hat{Z}$ is equivalent to minimising the variance of the importance weights. By manipulating the expression for the variance

$$\mathrm{Var}_q(\hat{Z}) = \mathrm{Var}\left(\int \frac{\gamma(x)}{q(x)}\hat{q}(dx)\right) = \frac{Z^2}{N}\int\left(\frac{\pi(x)^2}{q(x)}dx - 1\right),$$

we can see that the variance for $\hat{Z}$ and the importance weights is minimised when q is close to π.

2.2.3. Markov Chain Monte Carlo

Another method of sampling from π indirectly is Markov Chain Monte Carlo. See [12, 13, 14, 15, 16] for reference. The main idea of MCMC is to be able to obtain samples from the target distribution π by sampling from an ergodic Markov chain whose invariant distribution is π. This Markov chain is obtained by sampling from a proposal distribution $Q(\tilde{x}|x)$ invariant to π, i.e.

$$Q(\tilde{x}|x) \geq 0, \qquad \int Q(\tilde{x}|x)d\tilde{x} = 1, \qquad \int \pi(x)Q(\tilde{x}|x)dx = \pi(\tilde{x}) \quad \forall\tilde{x}, x \in \mathbb{R}.$$

The method produces an aperiodic and irreducible Markov chain $\{x_i\}_{i=1}^N$ with N samples from the target distribution. Let $q(\tilde{x}|x)$ be the density of the proposal distribution. Then the Markov chain has the transition kernel

$$Q(d\tilde{x}|x) = q(\tilde{x}|x)\alpha(x,\tilde{x})d\tilde{x} + r(x)\delta_x(d\tilde{x}),$$

where

$$\alpha(x,\tilde{x}) = \min\left(\frac{\pi(\tilde{x})q(\tilde{x}|x)}{\pi(x)q(x|\tilde{x})}, 1\right),$$
$$r(x) = 1 - \int q(\tilde{x}|x)\alpha(x,\tilde{x})d\tilde{x}.$$

The transition kernel is reversible and thus it satisfies the detailed balance condition

$$\pi(dx)Q(d\tilde{x}|x) = \pi(d\tilde{x})Q(dx|\tilde{x}).$$

When the proposal distribution $Q(\tilde{x}|x)$ is a random walk proposal the MCMC-algorithm is called Metropolis whilst when the $Q(\tilde{x}|x)$ is an independent proposal the resulting algorithm is referred to as Metropolis Hastings, described in Algorithm 1. By using samples from the MCMC-algorithm 2.4 can be approximated by

$$\widehat{E_\pi[\varphi(X)]} = \frac{1}{N}\sum_{i=1}^N \varphi(x^i),$$

and the chain of samples $\{x_i\}_{i=1}^N$ constitutes the particle approximation

$$\hat{\pi}(dx) = \frac{1}{N}\sum_{i=1}^N \delta_{x^i}(dx).$$

2.3. Sequential Monte Carlo

When assimilating data recursively by incorporating the latest sample standard Importane Sampling would require to use all the samples $x_{0:t}$ to recompute the importance weights as x_t becomes available. The computational cost of this procedure would increase with time. To circumvent this the weights can be computed sequentially by using Sequential Importance Sampling [11, 17].

Algorithm 1 Metropolis-Hastings MCMC

```
Initialise $x_0 \sim q_{x_0}(\cdot)$
For $i = 1, ..., N$ steps
    1. Propose $\tilde{x} \sim q(\cdot|x_{i-1})$
    2. Compute acceptance ratio $\alpha(x_{i-1}, \tilde{x}) \leq \min\left(\frac{\pi(\tilde{x})q(\tilde{x}|x_{i-1})}{\pi(x_{i-1})q(x_{i-1}|\tilde{x})}, 1\right)$
    3. With propability $\alpha(x_{i-1}, \tilde{x})$
           Set $x_i = \tilde{x}$ (accept proposal)
       otherwise
           Set $x_i = x_{i-1}$ (reject proposal)
```

2.3.1. Sequential Importance Sampling

Suppose the density $\pi_{t:0}(x_{0:t}) = \frac{\gamma(x_{0:t})}{Z}$ can be sequentially decomposed accordingly

$$\gamma(x_{0:t}) = \gamma(x_{0:t-1})\gamma(x_t|x_{0:t-1}) = ... = \gamma(x_0)\prod_{k=1}^{t}\gamma(x_t|x_{0:t-1}),$$

where $\gamma(x_t|x_{0:t-1}) = \frac{\gamma(x_{0:t})}{\gamma(x_{0:t-1})}$. Further suppose the importance density can be decomposed in a similar manner

$$q(x_{0:t}) = q(x_{0:t-1})q(x_t|x_{0:t-1}) = ... = q(x_0)\prod_{k=1}^{t}q(x_t|x_{0:t-1}).$$

where $q(x_t|x_{0:t-1}) = \frac{q(x_{0:t})}{q(x_{0:t-1})}$ Then the incremental importance weight can be written

$$w(x_{0:t}) = w(x_{0:t-1})w(x_{t-1}, x_t) = w(x_{0:t-1})\frac{\gamma(x_t|x_{0:t-1})}{q(x_t|x_{0:t-1})}, \tag{2.6}$$

or alternatively

$$w(x_{0:t}) = \frac{\gamma(x_{0:t})}{q(x_{0:t})} = \frac{\gamma(x_0)}{q(x_0)}\prod_{k=1}^{N}\frac{\gamma(x_t|x_{0:t-1})}{q(x_t|x_{0:t-1})} = \prod_{k=0}^{t}w_k(x_{k-1}, x_k). \tag{2.7}$$

This shows that as we assimilate a new sample point we only need to update the previous importance weight with the new incremental weight. Assume that i.i.d. samples can be obtained from the conditional importance density

$$x_t^i \overset{\text{iid}}{\sim} q_t(\cdot|x_{0:t-1}), \quad i = 1, ..., N.$$

We can then approximate 2.4 with

$$\widehat{E_{\pi_{0:t}}[\varphi(X)]} = \sum_{i=1}^{N} W_t^i \varphi(x^i),$$

where $W_t^i = \dfrac{w(x_{t-1:t}^i)}{\sum_{i=1}^{N} w(x_{t-1}^i)}$. The particle approximation of $\pi_{0:t}$ is

$$\hat{\pi}_{0:t}(dx_{0:t}) = \sum_{i=1}^{N} W_t^i \delta_{x_{0:t}^i}(dx_{0:t}).$$

The normalising constant is approximated by

$$\hat{Z}_t = \int \frac{\gamma(x_{0:t})}{q(x_{0:t})} \hat{q}_{0:t}(dx_{0:t}) = \frac{1}{N} \sum_{i=1}^{N} \frac{\gamma(x_{0:t}^i)}{q(x_{0:t}^i)},$$

where $\hat{q}(dx_{0:t}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{x_{0:t}^i}(dx_{0:t})$.

2.3.2. Sequential Importance Sampling for filtering

We can extend the Sequential Importance Sampling to the filtering problem of assimilating data points $\{y_t\}_{t=0}^T$ sequentially, [17, 18, 19, 20]. From 2.2 we have

$$p_\theta(x_{0:t}|y_{0:t}) \propto f_\theta(x_t|x_{t-1}) g_\theta(y_t|x_t) p_\theta(x_{0:k-1}|y_{1:k-1}).$$

Assuming i.i.d. samples can be obtained from the conditional importance density

$$x_t^i \overset{\text{iid}}{\sim} q_t(\cdot|x_{0:t-1}, y_{0:t}), \quad i = 1, ..., N.$$

Then the weights in 2.6 can be written

$$w(x_{0:t}) = w(x_{0:t-1}) w_k(x_{t-1}, x_t) \propto w(x_{0:t-1}) \frac{f_\theta(x_t|x_{t-1}) g_\theta(y_t|x_t)}{q_\theta(x_t|x_{0:t-1}, y_{1:t})},$$

where $w(x_{0:t-1}^i) \propto \frac{p_\theta(x_{0:t-1}|y_{1:t-1})}{q_\theta(x_{0:t-1}|y_{1:t-1})}$ so the weights in 2.7 can be expressed as

$$w(x_{0:t}) = \frac{p_\theta(x_{0:t})}{q_\theta(x_{0:t})} = \prod_{k=1}^{t} w_k(x_{t-1}, x_t) = \prod_{k=1}^{t} \frac{f_\theta(x_k|x_{k-1}) g_\theta(y_k|x_k)}{q(x_k|y_k, x_{k-1})}.$$

The particle approximations of $\pi_{0:t}$ and the filtering distribution π_t are similarly

$$\hat{\pi}_{0:t}(dx_{0:t}) = \sum_{i=1}^{N} W_t^i \delta_{x_{0:t}^i}(dx_{0:t}),$$
$$\hat{\pi}_t(dx_{0:t}) = \sum_{i=1}^{N} W_t^i \delta_{x_{0:t}^i}(dx_{0:t}).$$

The normalising constant is approximated by

$$\hat{Z}_t = \int w(x_{0:t}) \hat{q}_{0:t}(dx_{0:t}) = \frac{1}{N} \sum_{i=1}^{N} \prod_{k=1}^{t} w_k(x_{t-1}^i, x_t^i),$$

or equivalently

$$\hat{Z}_t = \left(\frac{1}{N}\right)^t \sum_{i=1}^{N} w_0(x_0^i) \prod_{k=1}^{t} \sum_{i=1}^{N} w_k(x_{k-1}^i, x_k^i) = \hat{p}_\theta(y_0) \prod_{k=1}^{t} \hat{p}_\theta(y_k|y_{0:k-1}). \tag{2.8}$$

This approximation is an unbiased estimate of $p(y_{0:t})$ which will prove to be useful [21].

Algorithm 2 Sequential Importance Sampling for filtering

```
For each time step $t = 0 : T$

1. At $t = 0$
   For each particle $i = 1 : N$ :
      a) Initialise $x_0^i \sim q_{x_0}(\cdot|y_0)$.
      b) Weight particle $w_0^i(x_0^i) = \frac{f_\theta(x_t^i|x_{t-1}) g_\theta(y_0|x_0^i)}{q(x_t^i|x_{0:t-1}^i, y_{0:t})}$ and normalise $W_0^i = \frac{w_0^i(x_0^i)}{\sum_{i'=1}^N w_0^{i'}(x_0^{i'})}$.
2. At $t \geq 1$
   For each particle $i = 1 : N$ :
      a) Propagate $x_t^i \sim q_\theta(x_t^i|x_{t-1:t}^i, y_{0:t})$ and set $x_{0:t}^i = (x_{0:t-1}^i, x_t^i)$.
      b) Weight particle $w_t(x_{t-1:t}) = \frac{f_\theta(x_t^i|x_{t-1}) g_\theta(y_t|x_t^i)}{q(x_t^i|x_{0:t-1}^i, y_{0:t})}$.
      c) Compute importance weight $w_t^i = W_{t-1}^i w_n(x_{t-1:t}^i)$ and normalise $W_t^i = \frac{w_t^i}{\sum_{i'=1}^N w_t^{i'}}$.
```

2.3.3. Resampling

In practice, after a couple of iterations a majority of the normalised importance weights degenerate and are very close to zero. This means that most of the computational effort is focused on particles that do not contribute to the final approximations. In addition the variance of the importance weights and thus the estimate of the normalising constant gets large. To measure this degree of weight degeneracy we use the effective sample size

$$\mathrm{ESS}_t = \frac{1}{\sum_{i=1}^N (W_t)^2}.$$

This measure approximates how many particles in a perfect Monte Carlo setting would yield the corresponding variance in the importance weights.

To prevent weight degeneracy weaker particles are eliminated by resampling, a sampling scheme that let particles generate new ones with a probability proportional to its weight. The ideas of resampling in the filtering context was originally proposed by [22]. The resampling procedure samples from the weighted set of particles $\{x_{0:t}^i, W_t^i\}_{i=1}^N$ and produces an equally weighted set of particles $\{\check{x}_{0:t}^i, \frac{1}{N}\}_{i=1}^N$. We will use the systematic resampling scheme described in Algorithm 3.

Algorithm 3 Systematic resampling, $\mathcal{R}(\{W_t^i\}_{i=1}^N)$

```
1. Sample $U \sim \mathcal{U}(0,1)$ and set $U_1 = \frac{U}{N}$.
   Set index $a_t^1 = \left\{k : \sum_{i=1}^{k-1} W_t^i \leq U_1 \leq \sum_{i=1}^{k} W_t^i \right\}$.
2. For i in 2:N:
   Set index $a_t^i = \left\{k : \sum_{i=1}^{k-1} W_t^i \leq U_1 \leq \sum_{i=1}^{k} W_t^i \right\}$.
```

Resampling effectively remedies weight degeneracy but the number of unique particles decrease as t grows large. In particular the joint filtering distribution $\pi_{0:T}$ will eventually be approximated by a single particle as $T - t$ increases and suffer from path degeneracy. This can partly be postponed by resorting to an adaptive resampling scheme and resample only when the effective sample size is below a certain threshold αN where α is some prespecified value on the unit interval. By combining the Importance Sampling algorithm with resampling gives us Sequential Importance Resampling which is the particle filter algorithm we will use throughout this thesis, see Algorithm 4.

Algorithm 4 Sequential Importance Resampling (Particle filtering), $\mathcal{PF}(Y_{0:T}, \theta)$

```
For each time step $t = 0 : T$ :
  1. Propagate and weight x.
     a) At $t = 0$
        For each particle $i = 1 : N$ :
        Initialise $x_0^i \sim q_\theta(x_0|y_0)$.
        Weight particle $w_0(x_0^i) = \frac{f_\theta(x_t^i|x_{t-1})g_\theta(y_0|x_0^i)}{q(x_t^i|x_{0:t-1}^i, y_{0:t})}$ and normalise $W_0^i = \frac{w_0^i}{\sum_{i'=1}^N w_0^{i'}}$.
     b) At $t \geq 1$
        For each particle $i = 1 : N$ :
        Propagate $x_t^i \sim q_\theta(x_t^i|x_{t-1:t}^{a_{t-1}^i})$ and set $x_{0:t}^i = (x_{0:t-1}^{a_{t-1}^i}, x_t^i)$.
        Weight particle $w_t(x_{t-1:t}) = \frac{f_\theta(x_t^i|x_{t-1})g_\theta(y_t|x_t^i)}{q(x_t^i|x_{0:t-1}^i, y_{0:t})}$.
        Compute importance weight $w_t^i = W_{t-1}^i w_n(x_{t-1:t}^i)$ and normalise $W_t^i = \frac{w_t^i}{\sum_{i'=1}^N w_t^{i'}}$.
        Set $a_t^i = i$.
  2. Compute statistics.
     ◦ Effective sample size $ESS_t = \frac{1}{\sum_{i=1}^N (W_t^i)^2}$.
     ◦ Evidence $\hat{p}_\theta(y_t|y_{0:t-1}) = \frac{1}{N}\sum_{i=1}^N w_n\left(x_{t-1:t}^i\right)$.
  3. Resample x. (if required)
     For each particle $i = 1 : N$ :
     a) Sample index $a_t^i \sim \mathcal{R}\left(\{W_t^i\}_{i=1}^N\right)$ of the ancestor to particle $i$.
     b) Set $W_t^i = \frac{1}{N}$.
  4. Compute filter estimate.
     $\hat{x}_t = \sum_{i=1}^N W_t^i x_t^{a_t^i}$.
Compute marginal likelihood $\hat{Z}_T = \prod_{t=0}^T \hat{p}_\theta(y_t|y_{0:t-1})$.
```

2.3.4. Annealed Importance Sampling

When dealing with problems of a high-dimensional nature the proposal distribution π_{t-1} and the target distribution π_t might differ significantly. This will cause the importance weights to have high variance. In order to prevent this variance inflation the transition between the proposal and the target can be bridged by introducing a sequence of intermediate distributions $\{\pi_{t,r}(x)\}_{r=0}^{R}$ from $\pi_{t-1}(x) = \pi_{t,0}(x)$ to $\pi_t(x) = \pi_{t,R}(x)$, [5, 6, 7, 8]. Between each of these intermediate distributions a Markov chain transition is simulated using a transition kernel $Q_{t,r}(\cdot|x_{t,r-1})$ that leaves $\pi_{t,r}(x)$ invariant. Let $\pi_{t,r}(x) \propto p_{t,r}(x)$ and

$$p_{t,r}(x) = p_{t,0}(x)^{1-\phi_{t,r}} p_{t,R}(x)^{\phi_{t,r}}, \tag{2.9}$$

for $r = 0, ..., R$. The sequence of temperatures $\{\phi_{t,r}\}_{r=0}^{R}$ satisfies $0 = \phi_{t,0} < \phi_{t,1} < \ldots < \phi_{t,R} = 1$ and are determined to provide a smooth transition from $\pi_{t,0}$ to $\pi_{t,R}$. Section 2.4.4 contains a brief discussion and description on how to choose these temperatures for our purposes.

The incremental importance weights can be written in a telescoping fashion

$$w_{t,R}(x_{t,0}, x_{t,R}) = \frac{p_{t,R}(x_{t,R})}{p_{t,0}(x_{t,R})} = \frac{p_{t,1}(x_{t,R})}{p_{t,0}(x_{t,R})} \frac{p_{t,2}(x_{t,R})}{p_{t,1}(x_{t,R})} \cdots \frac{p_{t,R}(x_{t,R})}{p_{t,R-1}(x_{t,R})},$$

where the intermediate incremental importance weights by plugging in 2.9 are

$$\begin{aligned} w_{t,r}(x_{t,r-1}, x_{t,r}) &= \frac{p_{t,r}(x_{t,r})}{p_{t,r-1}(x_{t,r})} = \frac{p_{t,0}(x_{t,r})^{1-\phi_{t,r}} p_{t,R}(x_{t,r})^{\phi_{t,r}}}{p_{t,0}(x_{t,r})^{1-\phi_{t,r-1}} p_{t,R}(x_{t,r})^{\phi_{t,r-1}}} \\ &= \left(\frac{p_{t,R}(x_{t,r})}{p_{t,0}(x_{t,r})}\right)^{\phi_{t,r}-\phi_{t,r-1}}. \end{aligned} \tag{2.10}$$

Algorithm 5 Sequential Annealed Importance Sampling

```
1. Sample $x_{t,0} \sim p_{t,0}(\cdot)$.
2. For $r$ in $1 : R$ :
    a) Generate $x_{t,r} \sim Q_{t,r}(\cdot|x_{t,r-1})$.
    b) Compute intermediate importance weight:
       $w_{t,r} = w_{t,r-1} \left(\frac{p_{t,r}(x)}{p_{t,r-1}(x)}\right)^{\phi_{t,r}-\phi_{t,r-1}}$.
```

2.4. Parameter inference

In this section we present theory for the parameter inference method we will use when estimating the parameters of the Lorenz' 96 model.

In order to perform particle filtering with Sequential Importance Resampling we need to use the parameter θ as an input. In applications this parameter is usually unknown and need to be estimated. The estimation of θ within the framework for state-space models is usually referred to as model calibration or system identification. This can be done using either a Bayesian or a frequentist approach, either off-line by using a batch of data points or on-line by incorporating the observations recursively in time. When using the Bayesian approach we need to assign prior distributions to the parameters and our goal is to obtain a posterior distribution for the parameters of interest in light of the data. In the frequentist, or equivalently the maximum likelihood (ML) approach we estimate θ by maximising the argument of the likelihood of the data.

For our purposes we use the SMC^2-algorithm described in 2.4.4 which is an off-line algorithm carried out within a Bayesian setting. An alternative approach would be to use the PMMH-algorithm. The main difference is that SMC^2 can incorporate new observations as they become available while PMMH has to be carried out fully to incorporate new information.

2.4.1. Bayesian approach

The task of performing Bayesian inference for $p(\theta|y_{0:t}) \propto p(\theta)\prod_{k=0}^{t} p(y_k|y_{0:k-1}\theta)$ can be done by targeting the posterior distributions $\{p(\theta|y_{0:t})\}_{t\geq 0}$ sequentially by introducing a population of N_θ θ-particles. Since $p(y_{0:t}|\theta)$ is intractable in

$$p(\theta|y_{0:t}) \propto p(\theta)p(y_{0:t}|\theta),$$

we can extend the state space to (x_t, θ_t) and use $x_{0:t}$ from an attached particle filter as auxiliary variables. So instead we approximate the joint posterior density

$$p(x_{0:t}, \theta|y_{0:t}) \propto p(\theta)p(x_{0:t}, y_{0:t}|\theta),$$

for which $x_{0:t}$ can be marginalised out to obtain the posterior of interest.

2.4.2. Iterated Batch Importance Sampling (IBIS)

We seek to explore the sequence of distributions $\{p(\theta|y_{0:t})\}_{t\geq 0}$ by incorporating the observations y_t sequentially. For a particle system of θ-particles at time t the target and importance distributions $p(\theta|y_{0:t})$ and $p(\theta|y_{0:t-1})$ respectively gives us the incremental weight for particle j

$$w^j_{\theta,t} \propto \frac{p(\theta^j|y_{1:t})}{p(\theta^j|y_{0:t-1})} \propto \frac{p(y_{0:t}|\theta^j)}{p(y_{0:t-1},|\theta^j)} = p(y_t|y_{0:t-1}, \theta^j). \tag{2.11}$$

By using the set of likelihood increments $\{p(y_t|y_{t-1}, \theta)\}_{j=1}^{N_\theta}$ the particles can be reweighed and resampled to get rid of the weak ones. This would however not allow the θ particles to explore the parameter space to a larger extent than at the initialisation. The resampling would cause the final approximation of the joint posterior to be based on a few distinct values of θ as t grows large. This path degeneracy can be mitigated by adding diversity into the θ-particles by introducing a move step which was proposed in [23] as Resample Move PF. The rejuvenation at time t can be done using a MCMC kernel invariant to $p(\theta|y_{0:t})$. Consider the two step recursion

$$p(\tilde{\theta}|y_{0:t}) \propto p(y_{0:t}|\tilde{\theta})p(\tilde{\theta}) = p(y_t|y_{0:t-1}, \tilde{\theta})p(\tilde{\theta}|y_{0:t-1}),$$

where by using an MCMC kernel $K_{t-1}(\theta'|\theta)$ invariant to $p(\tilde{\theta}|y_{0:t-1})$

$$p(\tilde{\theta}|y_{0:t-1}) = \int p(\tilde{\theta}, \theta|y_{0:t-1})d\theta = \int K_{t-1}(\tilde{\theta}|\theta)p(\theta|y_{0:t-1})d\theta,$$

then

$$p(\tilde{\theta}|y_{0:t})d\tilde{\theta} \propto p(y_t|y_{0:t-1}, \tilde{\theta}) \int K_{t-1}(d\tilde{\theta}|\theta)p(\theta|y_{0:t-1})d\theta.$$

This provides the foundation for what is known as Iterated Batch Importance Sampling, [6]. In analogy to Algorithm 3, IBIS targets $p(\tilde{\theta}|y_{0:t})$ with proposal distribution $K_{t-1}(d\tilde{\theta}|\theta)$ and incremental weight $w^j_{t,\theta} = p(y_t|y_{0:t-1}, \tilde{\theta}^j)$ for $j = 1, \dots N_\theta$.

Algorithm 6 Iterated Batch Importance Sampling

```
For $t$ in $0, ..., T$ :
  1. At $t = 0$
     For $j$ in $1 : N_\theta$ :
       a) Initialise $\theta^j_t \sim p(\theta_0)$ and set $w^j_{\theta,0} = \frac{1}{N}$.
       b) Compute importance weight $w_{\theta,0} = p(y_0|\theta^j_t)$.
  2. At $t \geq 1$
  3. Compute importance $\theta$-weights
       a) $\hat{p}(y_t|y_{0:t-1}, \theta^j_{t-1})$.
       b) $w^j_{\theta,t} = w^j_{\theta,t-1}\hat{p}_t(y_t|y_{0:t-1}, \theta^j_{t-1})$ and normalise $W^j_{\theta,t} = \frac{w^j_{\theta,t}}{\sum_{j'=1}^{N_\theta} w^{j'}_{\theta,t}}$.
  4. Resample (if required)
     For $j$ in $1 : N_\theta$:
       a) Sample indices of $\theta^j_{t-1}$-particle $o^j_t \sim \mathcal{R}\left(\{W^j_{\theta,t}\}_{j=1}^{N_\theta}\right)$.
       b) Set: $\left(\check{\theta}^j_{t-1}, W_{\theta,t}\right) \leftarrow \left(\theta^{o^j_t}_{t-1}, \frac{1}{N_\theta}\right)$.
  5. $\theta^j_t$ invariant mutation
     For $j$ in $1 : N_\theta$:
       a) Sample $\tilde{\theta}^j_t \sim K(\cdot|\check{\theta}^j_{t-1})$ according to Algorithm 1.
       b) Set $\theta^j_t \leftarrow \tilde{\theta}^j_t$
```

2.4.3. Particle MCMC

The incremental weights in 2.11 can be calculated when the state-space model of interest is linear Gaussian or when the states x_t takes values in a finite set. In other cases we need to resort to particle methods. We can construct a transition kernel invariant to $p(\theta|y_{0:t})$ by using ideas from particle MCMC, [24]. Such ideas uses a batch of observations $y_{0:T}$ to target the posterior density $p(x_{0:T}, \theta|y_{0:T})$. Choosing the proposal density

$$q((\tilde{x}_{0:T}, \tilde{\theta})|(x_{0:T}, \theta)) = q(\tilde{\theta}|\theta)p(\tilde{x}_{0:T}|y_{0:T}, \tilde{\theta}),$$

and by using that

$$p(x_{0:T},\theta|y_{0:T}) = \frac{p(x_{0:T},y_{0:T},\theta)}{p(y_{0:T})} = \frac{p(x_{0:T},y_{0:T}|\theta)p(\theta)}{p(y_{0:T})},$$

$$\frac{p(x_{0:T},y_{0:T}|\theta)p(\theta)}{p(x_{0:T}|y_{0:T},\theta)} = \frac{\frac{p(x_{0:T},y_{0:T},\theta)}{p(\theta)}p(\theta)}{\frac{p(x_{0:T},y_{0:T},\theta)}{p(y_{0:T},\theta)}} = \frac{p(y_{0:T},\theta)p(\theta)}{p(\theta)} = p(y_{0:T}|\theta)p(\theta),$$

the acceptance ratio in Algorithm 1 will be

$$\begin{aligned}\alpha((x_{0:T},\theta),(\tilde{x}_{0:T},\theta)) &= 1 \wedge \frac{p(\tilde{x}_{0:T},\tilde{\theta}|y_{0:T})q((x_{0:T},\theta)|(\tilde{x}_{0:T},\tilde{\theta}))}{p(x_{0:T},\theta|y_{0:T})q((\tilde{x}_{0:T},\tilde{\theta}|x_{0:T},\theta))} \\ &= 1 \wedge \frac{p(\tilde{x}_{0:T},y_{0:T}|\tilde{\theta})p(\tilde{\theta})q(\theta|\tilde{\theta})p(x_{0:T}|y_{0:T},\theta)}{p(x_{0:T},y_{0:T}|\theta)p(\theta)q(\tilde{\theta}|\theta)p(\tilde{x}_{0:T}|y_{0:T},\tilde{\theta})} \\ &= 1 \wedge \frac{p(y_{0:T}|\tilde{\theta})p(\tilde{\theta})q(\theta|\tilde{\theta})}{p(y_{0:T}|\theta)p(\theta)q(\tilde{\theta}|\theta)}. \end{aligned} \tag{2.12}$$

If it is not possible to sample from the proposal $p(\tilde{x}_{0:T}|y_{0:T},\tilde{\theta})$ nor to compute the likelihood $p(y_{0:t}|\tilde{\theta})$ we can use all sampled particles and offsprings in the particle filter up to and including time T, $\{\{x_t^i,a_t^i\}_{i=1}^N\}_{t=0}^T$ as auxiliary variables. We can then sample from the law of the particle filter

$$\hat{p}\left(\left\{\{\tilde{x}_t^i,\tilde{a}_t^i\}_{i=1}^{N_x}\right\}_{t=0}^T \middle| y_{0:T},\tilde{\theta}\right),$$

and as a by-product we obtain an unbiased estimate of the marginal likelihood $p(y_{0:T}|\theta)$ and its increments. By plugging in this estimate in 2.12 the acceptance ratio instead becomes

$$\alpha((x_{0:T},\theta),(\tilde{x}_{0:T},\theta)) = 1 \wedge \frac{\hat{p}(y_{0:T}|\tilde{\theta})p(\tilde{\theta})q(\theta|\tilde{\theta})}{\hat{p}(y_{0:T}|\tilde{\theta})p(\theta).q(\tilde{\theta}|\theta)}.$$

The use of the particle filter within Algorithm 1 like this is referred to as the Particle Marginal Metropolis Hastings (PMMH) sampler which targets

$$p\left(\theta,\left\{\{x_t^i,a_t^i\}_{i=1}^{N_x}\right\}_{t=0}^T \middle| y_{0:T}\right).$$

The likelihood estimate $\hat{p}(y_{0:T}|\theta)$ is unbiased so as N_x grows large we obtain our target of interest $p(\theta|y_{0:T})$ by marginalising out $\{\{x_t^i,a_t^i\}_{i=1}^{N_x}\}_{t=0}^T$ from 2.4.3, [24]. Hence a transition kernel based on the PMMH sampler is invariant to $p(\theta|y_{0:T})$.

Algorithm 7 PMCMC transition kernel $\mathcal{T}\left(\cdot|(\theta,\{x_{0:t}^i,a_{0:t}^i\}_{i=1}^{N_x})\right)$

```
Let $\left(\theta,\{x_{0:t}^i,a_{0:t}^i\}_{i=1}^{N_x}\right) = \mathcal{I}(\theta)$.
  1. Propose $\tilde{\theta} \sim T(\cdot|\theta)$
  2. Run a particle filter for $\tilde{\theta}$ with $\mathcal{PF}(y_{0:t},\tilde{\theta})$ according to Algorithm 4:
     a) Sample $\{\tilde{x}_{0:t}^i,\tilde{a}_{0:t}^i\}_{i=1}^{N_x}$ independently.
     b) Compute $\hat{Z}_t(\tilde{\theta},\{\tilde{x}_{0:t}^i,\tilde{a}_{0:t}^i\}_{i=1}^{N_x}) = p_0(y_0|\tilde{\theta})\prod_{k=1}^{t}\hat{p}_t(y_t|y_{0:t-1},\tilde{\theta})$.
```

```
 3. With probability

    $$1 \wedge \frac{\hat{Z}_t(\tilde{\theta}, \{\tilde{x}^i_{0:t}, \tilde{a}^i_{0:t}\}_{i=1}^{N_x})}{\hat{Z}_t(\theta, \{x^i_{0:t}, a^i_{0:t}\}_{i=1}^{N_x})} \frac{p(\tilde{\theta})}{p(\theta)} \frac{T(\theta|\tilde{\theta})}{T(\tilde{\theta}|\theta)}.$$

    accept the proposal and so replace the current particle system and its attached particle filter:
    $$\left(\theta, \left\{x^i_{0:t}, a^i_{0:t}\right\}_{i=1}^{N_x}\right) \leftarrow \left(\tilde{\theta}, \left\{\tilde{x}^i_{0:t}, \tilde{a}^i_{0:t}\right\}_{i=1}^{N_x}\right).$$
```

2.4.4. SMC^2

Combining IBIS with the particle MCMC transition kernel gives us the the SMC^2-algorithm. It is the particle equivalent of IBIS, SMC algorithm in the θ-dimension which propagates new particles with an attached particle filter in the x-dimension. The theoretical justification for the SMC^2-algorithm can be found in [25].

Algorithm 8 SMC^2

```
  For $t$ in $0 : T$ :

 1. At $t = 0$
    For $j$ in $1 : N_\theta$ :
      a) Initialise $\theta^j_0 \sim p_{\theta_0}(\cdot)$ and set $w^j_{\theta,0} = \frac{1}{N}$.
      b) For $i$ in $1 : N_x$ :
         Propagate $x^i_0 \sim q_{x_0}(\cdot|y_0)$.
      c) Compute importance weight $w_{\theta,0} = p(y_0|\theta^j_0)$.

 2. At $t \geq 1$
    For $j$ in $1 : N_\theta$ :
      a) For $i$ in $1 : N_x$ :
         Resample, propagate and weight the $x$-particles as in algorithm 3 for the particle filter to obtain $\left(x^{i,j}_{0:t}, a^{i,j}_{t-1}, w^{i,j}_{x,t}\right)$.
      b) Compute importance $\theta$-weights:
           i. $\hat{p}(y_t|y_{0:t-1}, \theta^j_{t-1}) = \frac{1}{N} \sum_{t=1}^{N_x} w^{i,j}_{x,t}$
          ii. $w^j_{\theta,t} = w^j_{\theta,t-1} \hat{p}_t(y_t|y_{0:t-1}, \theta^j_{t-1})$ and normalise $W^j_{\theta,t} = \frac{w^j_{\theta,t}}{\sum_{j'=1}^{N_\theta} w^{j'}_{\theta,t}}$.

 3. Resample (if required)
    For $j$ in $1 : N_\theta$:
      a) Sample indices of $\theta^j_{t-1}$-particle $o^j_t \sim \mathcal{R}\left(\{W^j_{\theta,t}\}_{j=1}^{N_\theta}\right)$.
      b) Set: $\left(\check{\theta}^j_{t-1}, W_{\theta,t}, \left\{\check{x}^{i,j}_{0:t}, \check{a}^{i,j}_{t-1}\right\}_{i=1}^{N_x}\right) \leftarrow \left(\theta^{o^j_t}_{t-1}, \frac{1}{N_\theta}, \left\{x^{i,o^j_t}_{0:t}, a^{i,o^j_t}_{t-1}\right\}_{i=1}^{N_x}\right)$.
```

4. θ_t^j invariant mutation

   For j in $1 : N_\theta$:

   a) Sample $\left(\tilde{\theta}_t^j, \left\{ \tilde{x}_{0:t}^{i,j}, \tilde{a}_{t-1}^{i,j} \right\}_{i=1}^{N_x}\right) \sim \mathcal{T}\left(\cdot \middle| \left(\check{\theta}_{t-1}^j, \left\{ \check{x}_{0:t}^{i,j}, \check{a}_{t-1}^{i,j} \right\}_{i=1}^{N_x}\right)\right)$ according to the PMCMC transition kernel in Algorithm 7.

   b) Set $\left(\theta_t^j, \left\{ x_{0:t}^{i,j}, a_{t-1}^{i,j} \right\}_{i=1}^{N_x}\right) \leftarrow \left(\tilde{\theta}_t^j, \left\{ \tilde{x}_{0:t}^{i,j}, \tilde{a}_{t-1}^{i,j} \right\}_{i=1}^{N_x}\right)$.

SMC^2 with adaptive tempering

As pointed out in Section 2.3.4 the transition from a proposal distribution to its target distribution can be smoothed. More specifically in the context of SMC^2 by adpoting the convention that $t, -1 = t-1, R$ we can introduce a sequence of intermediate distributions $\{p_{t,r}(\theta_{t,r-1}|y_{0:t}))\}_{r=0}^R$ from $p_{t-1}(\theta_{t-1}|y_{0:t}) = p_{t,0}(\theta_{t-1,R}|y_{0:t})$ to $p_t(\theta_{t-1}|y_{0:t}) = p_{t,R}(\theta_{t,R-1}|y_{0:t}))$.
In analogy to 2.9 and using that $p_{t,r}(\theta_{t,r-1}|y_{0:t}) \propto p_{t,r}(y_{0:t}|\theta_{t,r-1})$ if we let

$$p_{t,r}(y_{0:t}|\theta_{t,r-1}) = p_{t,0}(y_{0:t}|\theta_{t,r-1})^{1-\phi_{t,r-1}} p_{t,R}(y_{0:t}|\theta_{t,r-1})^{\phi_{t,r}},$$

then the intermediate incremental importance θ-weights corresponding to 2.10 are given by

$$\begin{aligned} p_{t,r}(y_t|y_{0:t-1}, \theta_{t,r-1}) &= \frac{p_{t,r}(y_{0:t}|\theta_{t,r-1})}{p_{t,r}(y_{0:t-1}|\theta_{t,r-1})} = \frac{p_{t,0}(y_{0:t}|\theta_{t,r-1})^{1-\phi_{t,r-1}} p_{t,R}(y_{0:t}|\theta_{t,r-1})^{\phi_{t,r}}}{p_{t,0}(y_{0:t-1}|\theta_{t,r-1})^{1-\phi_{t,r-1}} p_{t,R}(y_{0:t-1}|\theta_{t,r-1})^{\phi_{t,r}}} \\ &= \left(\frac{p_{t,R}(y_{0:t}|\theta_{t,r-1})}{p_{t,0}(y_{0:t-1}|\theta_{t,r-1})}\right)^{\phi_{t,r}-\phi_{t,r-1}} = p_t(y_t|y_{1:t-1}, \theta_{t,r-1})^{\phi_{t,r}-\phi_{t,r-1}}. \end{aligned}$$

The temperatures can be chosen in various ways. Since the difference between the proposal and the target distribution are greatest for early time points one might argue for a scheme using temperatures closer to 0 during initial time points. Then as more data points are assimilated and the proposal gets closer to the target gradually increase the temperatures. In the sequential context we dont't need to specify these temperatures in advance. Each temperature $\phi_{t,r}$ can be determined adaptively on-the-fly based on a criterion for the effective sample size as in [26] by calculating the normalised weights

$$W_{\theta,t,r}^j = \frac{p_t(y_t|y_{0:t-1}, \theta_{t,r-1}^j)^{\phi_{t,r}-\phi_{t,r-1}}}{\sum_{j'=1}^{N_\theta} p_t(y_t|y_{0:t-1}, \theta_{t,r-1}^{j'})^{\phi_{t,r}-\phi_{t,r-1}}},$$

and solving the following equation for the temperature $\phi_{t,r}$

$$\mathrm{ESS}_{t,r}(\phi_{t,r}) = \sum_{j=1}^{N_\theta} \frac{1}{(W_{\theta,t,r}^j)^2} \approx N_{tresh},$$

starting with $\phi_{t,r-1} = \phi_{t,0} = 0$ for some prespecified threshold N_{thresh}. It should be noted that $\hat{p}(y_t|y_{0:t}, \theta_{t,r-1}^j)^{\phi_{t,r}-\phi_{t-1,r}}$ is not an unbiased estimate of $p(y_t|y_{0:t}, \theta_{t,r-1}^j)^{\phi_{t,r}-\phi_{t-1,r}}$.

Algorithm 9 Adaptive Sequential Annealed Importance Sampling

```
1. Sample $\theta_{t,0} \sim p_0(\cdot)$ and set $\phi_{t,0} = 0$.
2. For $r$ in $1 : R$ :
   a) For $j$ in $1 : N_\theta$:
      Compute: $W^j_{\theta,t,r} = \dfrac{p(y_t|y_{0:t-1}, \theta^j_{t,r-1})^{\phi_{t,r}-\phi_{t,r-1}}}{\sum_{j'=1}^{N_\theta} p(y_t|y_{0:t-1}, \theta^{j'}_{t,r-1})^{\phi_{t,r}-\phi_{t,r-1}}}.$
   b) Solve the following equation for $\phi_{t,r}$:
      $\mathrm{ESS}_{t,r}(\phi_{t,r}) = \sum_{j=1}^{N_\theta} \dfrac{1}{(W^j_{\theta,t,r})^2} \approx N_{tresh}.$
   c) Compute intermediate importance weight:
      $w_{t,r} = w_{t,r-1} p_t(y_t|y_{1:t-1}, \theta_{t,r-1})^{\phi_{t,r}-\phi_{t,r-1}}.$
   d) Generate $\theta_{t,r} \sim K_{t,r}(\cdot|\theta_{t,r-1})$.
3. Set $\theta_t \leftarrow \theta_{t,R}$
```

Transition kernel with adaptive tempering within SMC^2

It can be a rather challenging task to construct a Markov transition kernel that mixes well. It might call for a seemingly countless number of pilot runs in particular when repeating the same procedure for different experimental configurations. Therefore we opt to use an adaptive kernel which incorporates the most recent information about the particle approximation $\pi_t(d\theta_{t,r-1}) = p(d\theta_{t,r-1}|y_{0:t})$ at time t, [27, 28, 9, 16, 29]. We estimate the first two moments of this particle approximation by

$$\hat{\mu}_{t,r} = \sum_{j=1}^{N_\theta} W^j_{\theta,t,r}\theta^j_{t,r-1}, \quad \hat{\Sigma}_{t,r} = \sum_{j=1}^{N_\theta} W^j_{\theta,t,r}(\theta^j_{t,r-1} - \hat{\mu}_{t,r})(\theta^j_{t,r-1} - \hat{\mu}_{t,r})^T.$$

This estimates will be used in the following proposal for particle j:

$$\tilde{\theta}^j_{t,r} = \hat{\mu}_{t,r} + \rho(\theta^j_{t,r-1} - \hat{\mu}_{t,r}) + \sqrt{1-\rho^2}\mathcal{N}(0, \hat{\Sigma}_{t,r}), \tag{2.13}$$

where ρ is the only parameter subject to tuning. The proposal in 2.13 can be seen as a hybrid between an independent and a random walk proposal. When ρ approaches 0 the proposal resembles an independent proposal and when ρ approaches 1 it resembles a random walk with steps that become increasingly smaller. The use of the proposal within an MCMC transition kernel leaves $p(\theta_{t,r}|y_{0:t})$ invariant since

$$\mathrm{E}(\tilde{\theta}_{t,r}) = \hat{\mu}_{t,r} = \mathrm{E}(\theta_{t,r-1}), \quad \mathrm{Cov}(\tilde{\theta}_{t,r}) = \hat{\Sigma}_{t,r} = \mathrm{Cov}(\theta_{t,r-1}).$$

Algorithm 10 Transition kernel $\mathcal{K}(\cdot|\{\theta^j_{t,r-1}, \mathcal{I}(\theta^j_{t,r-1})\})$

```
Let $\left(\theta^j_{t,r}, Z^j_{t-1}, \hat{p}_{t,r}(y_t|y_{0:t-1}, \theta^j_{t,r}), \left\{x^{i,j}_{0:t}, a^{i,j}_{0:t-1}, W^{i,j}_{x,t}\right\}_{i=1}^{N_x}\right) = \{\theta^j_{t,r-1}, \mathcal{I}(\theta^j_{t,r-1})\}$ and let $\hat{\mu}_{t,r}$ and $\hat{\Sigma}_{t,r}$ be known approximations.

For $i = 1, ..., M$ steps
   1. Propose $\tilde{\theta}^j_{t,r} = \hat{\mu}_{t,r} + \rho(\theta^j_{t,r} - \hat{\mu}_{t,r}) + \sqrt{1-\rho^2}\mathcal{N}(0, \hat{\Sigma}_{t,r})$.
   2. Run a particle filter for $\tilde{\theta}^j_{t,r}$ with $\mathcal{PF}(y_{0:t}, \tilde{\theta}^j_{t,r})$ to:
```

a) Obtain $\left\{\tilde{x}^{i,j}_{0:t}, \tilde{a}^{i,j}_{0:t-1}, \tilde{W}^{i,j}_{x,t}\right\}_{i=1}^{N_x}$.

b) Compute $\hat{p}_t(y_t|y_{0:t-1}, \tilde{\theta}^j_{t,r})$ and $\tilde{Z}^j_{t-1} = \hat{p}_0(y_0|\tilde{\theta}^j_{t,r}) \prod_{k=1}^{t-1} \hat{p}_t(y_t|y_{0:t-1}, \tilde{\theta}^j_{t,r})$.

3. With probability

$$1 \wedge \frac{\hat{p}_t(y_t|y_{0:t-1}, \tilde{\theta}^j_{t,r})^{\phi_{t,r}} \tilde{Z}^j_{t-1}}{\hat{p}_t(y_t|y_{0:t-1}, \theta^j_{t,r})^{\phi_{t,r}} Z^j_{t-1}} \frac{p(\tilde{\theta}^j_{t,r})}{p(\theta^j_{t,r})} \frac{q(\theta^j_{t,r}|\theta^{\tilde{j}}_{t,r})}{q(\theta^{\tilde{j}}_{t,r}|\theta^j_{t,r})}$$

accept the proposal, replace the particle system and its attached particle filter:

- ○ $\theta^j_{t,r} \leftarrow \tilde{\theta}^j_{t,r}$
- ○ $\left(Z^j_{t-1}, \hat{p}_t(y_t|y_{0:t-1}, \theta^j_{t,r})\right) \leftarrow \left(\tilde{Z}^j_{t-1,r}, \hat{p}_t(y_t|y_{0:t-1}, \tilde{\theta}^j_{t,r})\right)$
- ○ $\left\{x^{i,j}_{0:t}, a^{i,j}_{0:t-1}, W^{i,j}_{x,t}\right\}_{i=1}^{N_x} \leftarrow \left\{\tilde{x}^{i,j}_{0:t}, \tilde{a}^{i,j}_{0:t-1}, \tilde{W}^{i,j}_{x,t}\right\}_{i=1}^{N_x}$.

The prior and transition densities used are the Inverse Gamma and Multivariate Normal:
$p(\theta) = \dfrac{\beta^\alpha}{\Gamma(\alpha)} x^{-\alpha-1} \exp\left(-\frac{\beta}{\theta}\right)$,
$q(\tilde{\theta}|\theta) = \exp\left\{-\frac{1}{2(1-\rho^2))} \sum_{j=1}^{N_\theta} (\tilde{\theta}^j - \hat{\mu} - \rho(\theta^j - \hat{\mu}))^T \hat{\Sigma}^{-1} (\tilde{\theta}^j - \hat{\mu} - \rho(\theta^j - \hat{\mu}))\right\}$.

The final algorithm that combines SMC^2 with adaptive tempering and the transition kernel in Algorithm 10 is presented in A.2.1.

To summarise, the SMC methods enables us to assimilate observations sequentially and update the importance weights at time t without having to use all observations up to time t. Choosing a proposal close the target will minimse the variance of the weights. The variance of the weights will increase with time and this weight degeneracy can be mitigated by resampling in order to get rid of the weak particles. This will result in low variance in the estimate of the normalising constant. In the filtering case this estimate is an unbiased estimate of the marginal density $p(y_{0:t})$.

When using a Bayesian approach in parameter inference we target the posterior distribution $p(\theta|y_{0:t})$. Due to the intractability of $p(y_{0:t}|\theta)$ this can be carried out using an SMC algorithm in the θ-dimension with an attached particle filter in the x-dimension, targeting $p(\theta, x_{0:t}|y_{0:t})$. To provide dynamics for the θ-particles and to prevent path degeneracy we use an MCMC kernel invariant to $p(\theta|y_{0:t})$. To be able to compute the acceptance ratio in the MCMC kernel we can obtain unbiased esimates for $p(y_{0:t}|\theta)$ from the attached particle filter. This gives us the SMC^2-algorithm. For problems of a higher dimensional nature the distances between two subsequent distribution $p(\theta|y_{0:t-1})$ and $p(\theta|y_{0:t})$ might cause the variance of the weighs to be inflated. To limit this variance inflation and smooth the transition we can use adaptive tempering. In addition we can use an adaptive transition kernel for the rejuvenation of θ-particles.

3. Lorenz '96 Equations

In this chapter we present the model for which we will perform tracking and parameter estimation in the state-space framework. We depict the behavior under different regimes in bifurcation plots and illustrate the rate of separation of the trajectories.

3.1. Definition

The Lorenz '96 equations is commonly applied to simulation of one dimensional atmospheric behavior. The model is a mathematical representation of a chaotic dynamical system where the future states evolve from the present states and where small changes in the initial states can lead to diverging outcomes [1]. It is governed by the system of D ordinary differential equations (ODEs)

$$\frac{dx_d(t)}{dt} = x_{d-1}(t)(x_{d+1}(t) - x_{d-2}(t)) - x_d(t) + F, \tag{3.1}$$

where x_d is assumed to be cyclic, i.e. $x_{d-D} \equiv x_{d+D} = x_d$. In the atmospheric context the variables can be thought of as being placed on an equispaced grid on a latitudinal circle where the constant F induces external force, the linear term dissapation and the quadratic term convection.

3.2. State-space representation

To be able to perform inference using the state-space framework we can convert the ODEs in 3.1 to stochastic differential equations (SDEs) and assume that noisy observations of the continuous process are available at discrete time points $0 \leq t_0 \leq t_1 \leq \ldots \leq t_n = T$. The SDEs are given by

$$dX_d(t) = X_{d-1}(t)(X_{d+1}(t) - X_{d-2}(t)) - X_d(t) + F)dt + \sigma dW_d(t),$$

where dW_d is an increment of a standard Wiener process and σ its scaling parameter. By interpreting these SDEs in the Stratonovic sense they can be written as the ODEs [2, 30]

$$\frac{dx_k(t)}{dt} = x_{k-1}(t)(x_{k+1}(t) - x_{k-2}(t)) - x_k(t) + F + \sigma\frac{\Delta W_k(t)}{\Delta t}, \tag{3.2}$$

where $\Delta W_d \sim \mathcal{N}(0, \Delta t)$. We assume the state-space model is given by

$$\begin{aligned} dX_d(t) =& X_{d-1}(t)(X_{d+1}(t) - X_{d-2}(t)) - X_d(t) + F)dt + \sigma dW_d(t), \quad X(0) \in \mathbb{R}^D, \\ Y_d(t) =& H(X_d(t), \varepsilon_d(t)), \quad \varepsilon_d(t) \sim \mathcal{N}(0, \sigma_\varepsilon^2). \end{aligned} \tag{3.3}$$

for $d = 0, \ldots, D$ and where the function H determines which elements d of the vector X_t that are observed. In the following we will use the convention $X_t = (X_1(t), \ldots, X_D(t))$, $\mathbf{x}_t = (x_1(t), \ldots, x_D(t))$ and similar for Y_t and $\mathbf{y}_t$.

3.3. Visualisations of trajectories for different F

If otherwise not stated we assume that the forcing parameter $F = 8$, the standard deviation of the system noise $\sigma = 0.5$ and the observation noise $\sigma_\varepsilon = 1$. To obtain the trajectory of the process $\{X_t\}_t^T$ we integrate the ODEs in 3.2 numerically with the 4th order Runge Kutta scheme described in A.2.2 using step size $\Delta t = 0.05$, [31, 32]. The observational process $\{Y_t\}_t^T$ is obtained by adding the i.i.d. noise ε in 3.3 to the latent process. We assume that $\mathbf{y}_0$ is unobserved and that the initial state is uniformly distributed, $\mathbf{x}_0 \sim \mathcal{U}(-3, 3)$. Simulated trajectories of a an eight-dimensional Lorenz '96 system for $T = 100$ is plotted in Figure A.1 in section A.3.

A good illustration of the behavior of the deterministic model in 3.1 is to plot the state values for a grid of 5000 possible values of the forcing parameter F using the same set of simulated initial values. We can see in 3.1 that as F grows larger the possible set of state values grow larger. The evident structure of the bifurcation points in the deterministic model when $F \leq 5$ gets more and more blurry as F grows beyond 5. Figure 3.2 shows the same plots for the stochastic model where σ for each trajectory is drawn from $\mathcal{IG}(1, 1/3)$.

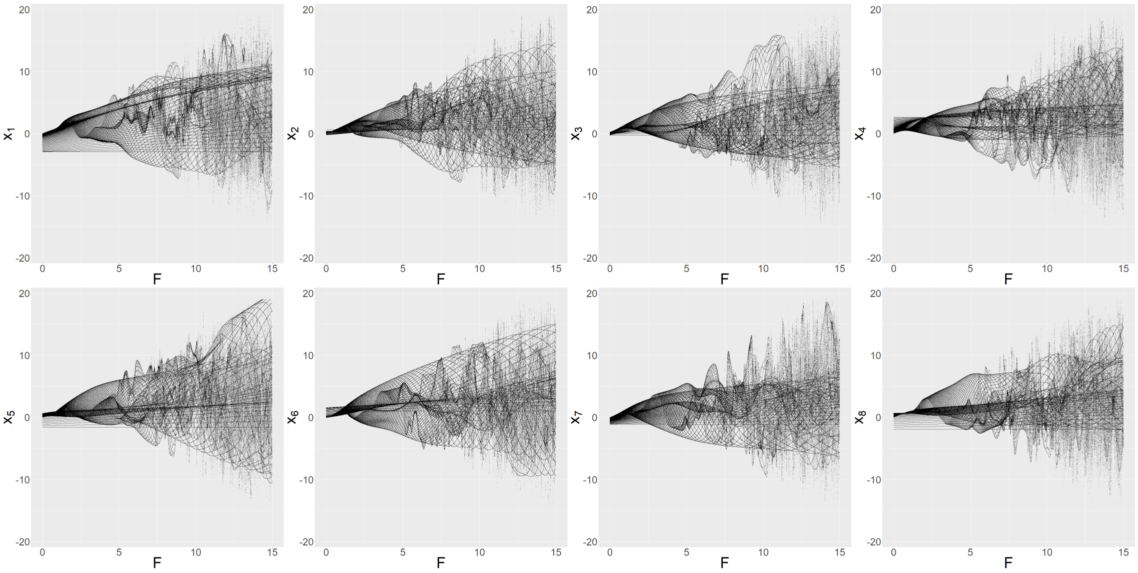

Figure 3.1.: Bifurcation plots of the individual components of the state vector in a deterministic eight-dimensional Lorenz '96 model. The elements of the vector are plotted rowwise.

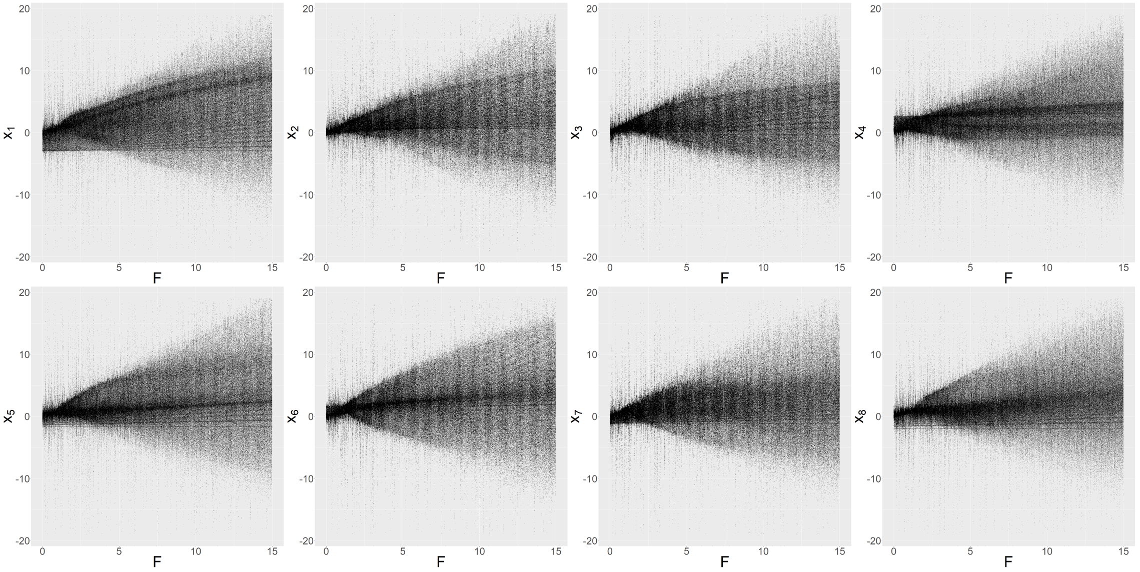

Figure 3.2.: State vector in an eight-dimensional Lorenz '96 model, with the elements of the vector rowwise from 1 to 8.

In Figure 3.4 we plot bifurcation plots for the state vector marginalised over its components both for the deterministic and the stochastic model. In both plots the values show decaying behavior for $F \leq 1$, periodic behavior for $F \geq 1$ and as F grows beyond 5 there is no apparent structure which indicates a chaotic behavior.

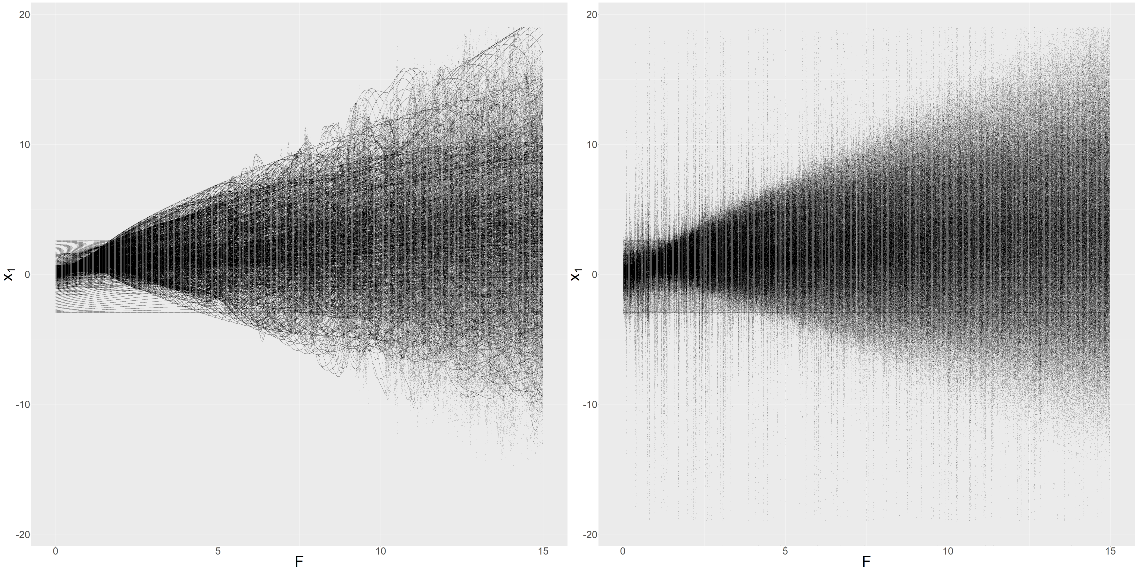

Figure 3.3.: States in an eight-dimensional Lorenz '96 model marginalised for the coordinates d.

3.4. Lyapunov's exponents and rate of separation

As mentioned in Chapter 1 one characteristic trait of a chaotic system is its sensitivity to initial conditions. Predictability of the Lorenz '96 model is of special interest especially because of this inherent sensitivity. In particular if two states initially are separated by an infinitesimally small distance we can use the largest Lyapunov exponent to quantify the rate at which these two states further separate through time [33].

For some distance measure two states $\mathbf{x}$ and $\dot{\mathbf{x}}$ with initial separation $|\delta Z(0)| = |\mathbf{x}(0) - \dot{\mathbf{x}}(0)|$

diverge at a rate given by

$$|\delta Z(t))| \approx e^{\lambda t}|\delta Z(0)|,$$

where λ is the largest Lyapunov exponent. A positive value of Lyapunov's largest exponent is an indication of chaos. To illustrate this growth rate and how it changes over time in the Lorenz '96 model as in [1] we simulate an initial state $\mathbf{x}_0 \sim \mathcal{U}(-3, 3)$. We then we add a random perturbation corresponding to the initial separation $|\delta Z(0)|$. For these two states we integrate equation 3.1 forward in time with time step $\Delta t = 0.05$ and calculate the Euklidean distance between the points of the two trajectories at every time point. This is done until the states at time t are so far apart they could have been randomly chosen, i.e. when the error reaches saturation. This procedure is repeated 500 times and averaged over the log values of these distances. See Algorithm 13 for details.

Figure 3.4 plots the difference between the average log distances at time $t = 0$ and time t as a function of t. The slope of the curves should be interpreted as an estimate of Lyapunov's largest exponent. From the figure we can se that for $F = 1$, λ is negative which corresponds to an exponentially decaying growth rate. In the same figure we can see that an intermediate value $F = 3$ induces a quasiperiodic behavior of the growth rate, λ alternates between both positive and negative values. When $F = 8$, λ is a positive constant corresponding to an exponentially increasing growth rate in time which is an indication of that the system is chaotic. When $F = 8$ the estimated slope of the curve is approximately 0.11 which gives an estimated Lyaponov time around 9. Lyapunov time is the inverse of the Lyapunov's exponent, defined as the time for the distance between the trajectories to increase by a factor of e.

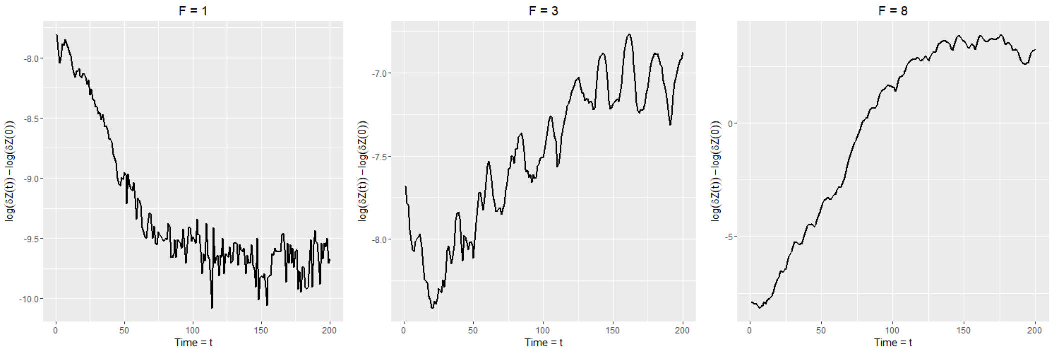

Figure 3.4.: Averaged log separation of 500 pair of trajectories over time for different values of the forcing parameter F.

4. Results

In this Chapter we present the results from applying the particle filter and the SMC^2-algorithm to simulated data from the Lorenz '96 model.

In Section 4.1 outline the experimental setup for the implementations and the measures of performance that we use.

In Section 4.3.1 we show the particle filter's performance in terms of tracking MSE and compare it for different number of particles N_x. We present how the variance of the normalising constant changes for different number of particles. This will serve as a guidance for how many particles that is needed within the SMC^2-algorithm for the parameter inference. We also investigate the tracking performance of the particle filter when the system is observed less accurate, i.e. when not all components of the state vector are observed and when the the observation noise is larger.

In Section 4.3.2 Then we present the resulting posterior distributions for different priors, likelihood evaluations and observation schemes together with the mean squared error for the parameters. This is followed by diagnostics for the SMC^2-algorithm. The diagnostics include monitoring of the effective sample size, acceptance rate and decorrelation of the rejuvenation steps. To conclude the result we present some results for adaptive tempering in a similar manner.

4.1. Experimental setup

4.1.1. Particle filtering

If nothing else stated we initialise the particle filter $\mathbf{x}_0 \sim \mathcal{U}(-3, 3)$ for all its implementations and assume that this state is unobserved. We will use a bootstrap proposal and sample from the dynamics $f(\mathbf{x}_t|\mathbf{x}_{t-1})$ hence the incremental weights are given by the likelihood only, $w_{x,t}(\mathbf{x}_{t-1:t}) = g_\theta(\mathbf{y}_t|\mathbf{x}_t)$.

4.1.2. Parameter inference

As pointed out in 2.4 the posterior of interest when performing parameter inference in the Bayesian setting is $p(\theta|\mathbf{y}_{0:t})$, in particular we are interested in the marginal posterior distributions $p(F|\mathbf{y}_{0:t})$ and $p(\sigma|\mathbf{y}_{0:t})$. We target the joint posterior density given by

$$p(\mathbf{x}_{0:t}, F, \sigma|\mathbf{y}_{0:t}) \propto p(F)p(\sigma)p(\mathbf{x}_0) \prod_{k=1}^{t} p(\mathbf{x}_k|\mathbf{x}_{k-1}, F, \sigma) \prod_{k=1}^{t} p(\mathbf{y}_t|\mathbf{x}_t, F, \sigma). \tag{4.1}$$

It is common practice to use an $\mathcal{IG}$ prior for σ^2 and the conjugacy of the Normal and the Inverse Gamma distributions to get a closed form expression for the joint posterior in 4.1. For our purposes we do not need such and it suffices to do without a closed form expression.

4.1.3. Experimental configurations

The algorithm is implemented for the different sets of particles, priors, likelihoods, observations schemes shown in Table 4.1. The number of times the algorithm is repeated for same configuration is denoted N_{runs}. The constant γ determines when tempering is used by fixing the effective sample size above γN_θ. The Cauchy likelihood is denoted with $\mathcal{C}$. For the first four configurations in the table $\{\mathbf{y}_t\}_{t=1}^{40}$ are assimilated and for the adaptive tempering scheme $\{\mathbf{y}_t\}_{t=1}^{10}$ due to the computational cost. For the high-dimensional problems we use a more narrow prior for the initial state to diminish the effect of the initialisation error.

The configuration in the first row of the table serves as a benchwark and to ease the presentation of the results the other configurations are referred to what makes them different from this benchmark. The configurations in row 1-4 is respectively referred to as: Benchmark, Prior $\mathcal{IG}(1, 1/3)$, Likelihood $\mathcal{C}$, Observation scheme $d_y = 2$.

Table 4.1.: Experiment configurations

d_x	d_y	N_θ	N_x	Prior x_0	Prior F	Prior σ	Likelihood	γ	ρ	N_{runs}
8	8	500	2000	$\mathcal{U}(-3,3)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{N}(0,1)$	1.0	0.6	10
8	8	500	2000	$\mathcal{U}(-3,3)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1)$	$\mathcal{N}(0,1)$	1.0	0.5	10
8	8	500	2000	$\mathcal{U}(-3,3)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{C}(0,1)$	1.0	0.6	10
8	2	500	2000	$\mathcal{U}(-3,3)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{N}(0,1)$	1.0	0.6	10
8	8	400	2000	$\mathcal{U}(-1,1)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{N}(0,1)$	0.5	0.0001	1
16	16	400	2000	$\mathcal{U}(-1,1)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{N}(0,1)$	0.5	0.99	1
32	32	400	2000	$\mathcal{U}(-1,1)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{N}(0,1)$	0.5	0.99	1
64	64	400	2000	$\mathcal{U}(-1,1)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{N}(0,1)$	0.5	0.99	1
96	96	400	2000	$\mathcal{U}(-1,1)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{N}(0,1)$	0.5	0.99	1
128	128	400	2000	$\mathcal{U}(-1,1)$	$\mathcal{U}(6,11)$	$\mathcal{IG}(1,1/3)$	$\mathcal{N}(0,1)$	0.5	0.99	1

4.2. Measures of performance and efficiency

When measuring the performance of the particle filter we average the estimated measures over R runs, R being sufficiently large in order to reduce the Monte Carlo variance the estimates.

4.2.1. Tracking performance

To compare the tracking performance of the particle filter for different number of particles N_x we track the d-dimensional latent process $\{X_t\}_{t=1}^T$ and calculate the mean squared error (MSE) of its estimates

$$AMSE(\mathbf{x}_t) = \sum_{d=1}^{D} \frac{1}{N_x} \sum_{i=1}^{N_x} (x_d^i(t) - x_d(t))^2,$$

$$MSE(\mathbf{x}_{1:T}) = \sum_{t=1}^{T} AMSE(\mathbf{x}_t).$$

We estimate these quantities by averaging over R multiple runs and calculate

$$\widehat{AMSE}(\mathbf{x}_t) = \frac{1}{R} \sum_{t=1}^{R} (AMSE(\mathbf{x}_t))_i,$$

$$\widehat{MSE(x_{1:T}}) = \sum_{t=1}^{T} \widehat{AMSE(x_t})}.$$

4.2.2. Variance of the normalising constant

For the parameter inference withing SMC^2 we need to estimate the marginal likelihood $p(\mathbf{y}_{0:t})$. This estimate need to be precise but at the same time it cannot be too expensive as the computational cost within the algorithm grows with N_x. We calculate the variance of this estimate which is expected to grow linearly in time [34] accordingly

$$\mathrm{Var}(\hat{p}(\mathbf{y}_{0:T})) = \mathrm{Var}\left(\sum_{t=1}^{T}\hat{p}(\mathbf{y}_k|\mathbf{y}_{0:k-1})\right) = \sum_{k=1}^{T}\mathrm{Var}(\hat{p}(\mathbf{y}_k|\mathbf{y}_{0:k-1})). \tag{4.2}$$

The variance of the likelihood increments is calculated over a set of R multiple runs

$$\mathrm{Var}(\hat{p}(\mathbf{y}_k|\mathbf{y}_{0:k-1})) = \frac{1}{R-1}\sum_{i=1}^{R}(p(\mathbf{y}_k|\mathbf{y}_{0:k-1}) - \bar{p}(\mathbf{y}_k|\mathbf{y}_{0:k-1}))^2, \tag{4.3}$$

where

$$\bar{p}(\mathbf{y}_k|\mathbf{y}_{0:k-1}) = \frac{1}{R}\sum_{i=1}^{R}p(\mathbf{y}_k|\mathbf{y}_{0:k-1}). \tag{4.4}$$

4.2.3. Observation schemes

Since the underlying process is chaotic the performance of the particle filter is expected to substantially decrease when the signal is not observed very accurately, i.e. when the observation noise is large or when not all components of the state vector are observed. When observing fewer elements of the state vector the particle filter tracks a process in a smaller dimensional space. Hence the probability mass is larger compared to the entire space with lower variance of the weights as a result. The more informative the better the performance will be up to some degree as long as the observation noise is not too small.

4.2.4. Decorrelation of MCMC moves

The scaling parameter of our transition kernel ρ is tuned in order to obtain an average acceptance rate of the move steps in the region $0.2 - 0.3$ at time T. The purpose of the rejuvenation and its move steps is to jitter the particle population. The extent of the jitter will partly depend on how many move steps that are done in the rejuvenation. In the best of worlds the particles after M steps $\theta^j_{t,r}(M)$ are uncorrelated with the initial particles $\theta^j_{t,r}(0)$. In order to judge if enough diversity has been inserted, in addition to scatterplots, we use the measure of decorrelation

$$J_{t,r} = \frac{\sum_{j=1}^{N_\theta}\left|\breve{\theta}^j_{t,r} - \tilde{\theta}_{t,r}\right|^2}{\sum_{i=1}^{N_\theta}\left|\theta^j_{t,r} - \bar{\theta}_{t,r}\right|^2}. \tag{4.5}$$

The measure should converge to $1 - \mathrm{corr}(\theta^j_{t,r}(M), \theta^j_{t,r}(0))$ as N_θ grows large and should be above 0.01-0.05 [9].

4.2.5. Computational cost

The computational cost of the SMC^2-algorithm is $\mathcal{O}(tN_\theta N_x M(1+\kappa))$ where κ is the the number of tempering steps. Not using tempering at all corresponds to setting N_{thresh} equal to 0 and the algorithm collapses to a standard SMC^2.

4.3. Numerical results

4.3.1. Particle filtering

Tracking

In plots (a) and (b) in Figure 4.1 we can see that the tracking MSE is improved when the number of particles are increased. The improvement is not proportional to the increase in the number of particles and the MSE increases linear in time.

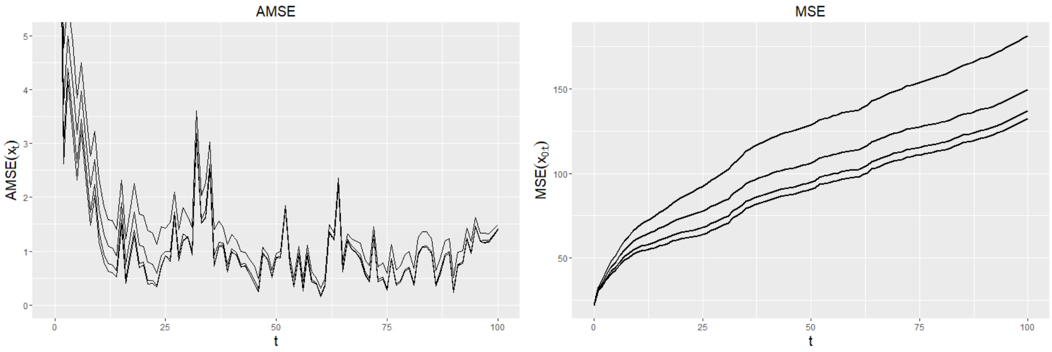

Figure 4.1.: Tracking MSE averaged over 1000 runs when using $N_x = 500, 1000, 2000$ and 3000 (black lines from top to bottom) number of particles up to time $T = 100$. Left: $AMSE(\mathbf{x}_t)$. Right: $MSE(\mathbf{x}_{0:t})$.

Variance of the normalising constant

From Figure 4.2 we can observe that the variance of the normalising constant increases linear in time. As in the case for the tracking the increase in precision is diminishing and the gain is not proportional to the cost. For the SMC^2-algorithm using 2000 particles seem to be a conservative choice.

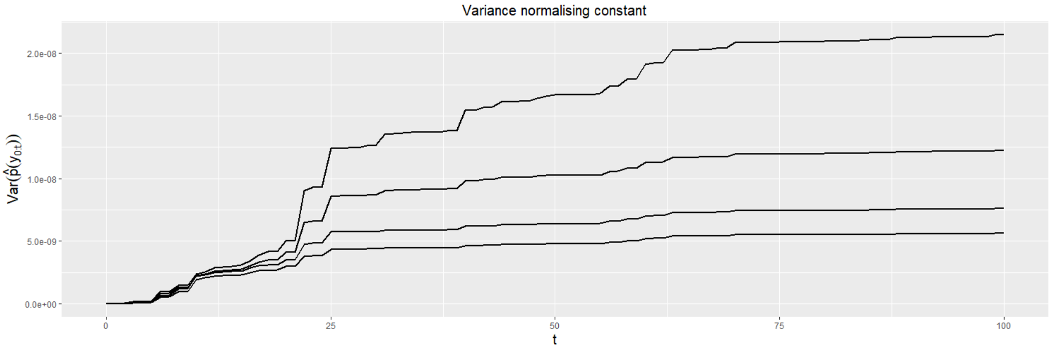

Figure 4.2.: Variance of the normalising constant averaged over 1000 runs when using $N_x = 500, 1000, 2000$ and 3000 (black lines from top to bottom) number of particles for a trajectory up to $T = 100$.

Observation schemes

Figure 4.3 depicts a typical tracking of the Lorenz '96 model when only the first component is observed. For some of the components that are not observed the marginal filtering distribution at a given time point is not centered around one region. This shows that the performance of the particle filter heavily deteriorates when $d_y = 1$. The plots also illustrate the cyclic nature of the model, the tracking of the components worsens the farther away the components are from the observed one.

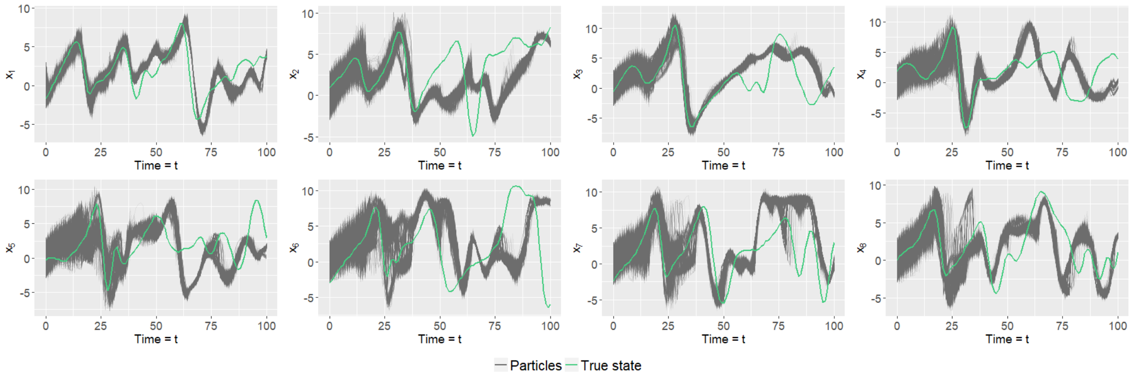

Figure 4.3.: Trajectories of each coordinate of the latent process X_t and its marginal filtering distributions $\hat{p}(dx_d(t)|\mathbf{y}_{0:t})$ for $t = 0, \ldots, 100$. The coordinates are plotted row-wise.

From plot (a) in 4.4 we can see that the tracking gradually gets worse as we observe fewer elements of the observation vector. Based on the same plot it appears that the MSE increase faster than linear for $d_y = 1$. After ploting the same plot for a longer time horizon we concluded that this increase was just local in time and that the tracking MSE actually increases linear in time for all considered observation schemes. When observing twice as many components of the state vector the tracking error is improved by more than a factor of two.

Plot (b) in the same figure indicates that the variance of the normalising constant decreases as we observe fewer elements of the state vector. This indicates that we can expect a higher effective sample size in the SCM^2-algorithm when we observe fewer coordinates. The high Monte Carlo variance suggests that 1000 is a conservative number of multiple runs for this measure.

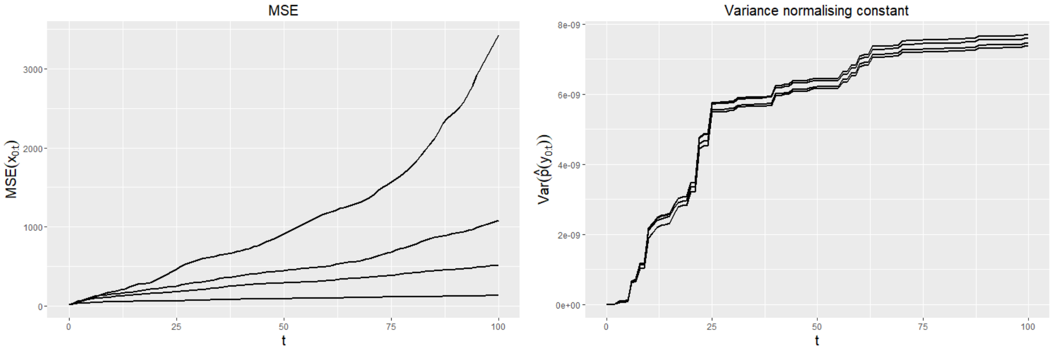

Figure 4.4.: Averages over 1000 multiple runs when using $N_x = 2000$ particles for different dimensions of the observation vector. Left: MSE, $MSE(\mathbf{x}_{0:t})$ with $d_y = 1, 2, 4, 8$ respectively from high to low. Right: Variance of the normalising constant, $\text{Var}(\hat{p}(\mathbf{y}_{0:t}))$ with $d_y = 1, 2, 4, 8$ respectively from low to high.

In plot (a) in Figure 4.5 the tracking MSE decreases as the number of particles increase for all combinations of observation schemes and noise. The improvement of an increase in the number of particles is smaller for larger observation noise when the system is not fully observed. This is illustrated by the fact that for constant values $d_y = 1, 2, 4$ the dotted lines are farther apart from the solid lines for larger values of N_x than for smaller ones. Furthermore, when the system is not observed fully the decrease in tracking MSE is relatively small when the observation noise is increased from $\sigma_\varepsilon = 1$ to $\sigma_\varepsilon = 2$ compared to when $d_y = 8$. This is suggested by the shifts of the solid curves to the the dotted curves are relatively small for $d_y = 1, 2, 4$ in comparison to when $d_y = 8$.

In plot (b) in the same figure we can see that when using 5000 particles the particle filter perfoms equally well for $d_y = 8, \sigma = 2$ and $d_y = 4, \sigma = 1$. So for a large number of particles the performance in this case is the same if the noise is doubled and when d_y is halfed. This is illustrated by the two lines almost intersecting.

Figure 4.5 is based on Table A.1 and Table A.2 in Appendix. These tables also include the standard deviation of the MSE. Overall, the standard deviation changes dramatically as the system goes from being partially observed to being fully observed. The difference is greater for smaller observation noise.

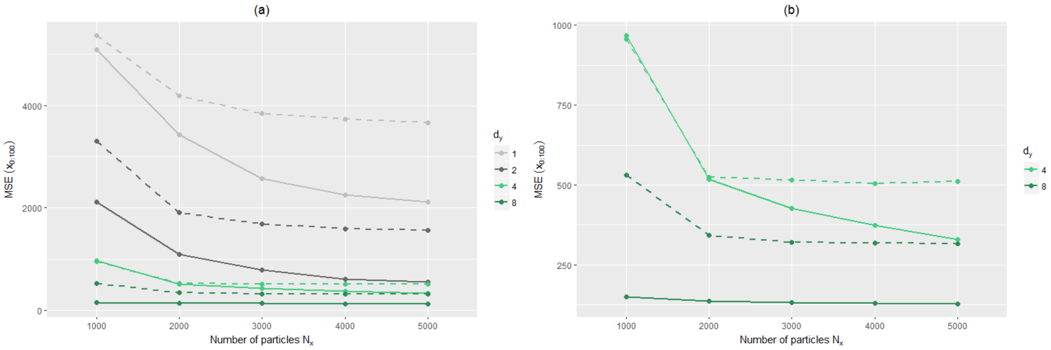

Figure 4.5.: Tracking MSE averages over 1000 multiple runs for different combinations of number of particles, observations schemes and noise. The different observation noises $\sigma_\varepsilon = 1$ and $\sigma_\varepsilon = 2$ are indicated with solid and dotted lines respectively.

Observation schemes in different regimes

As we saw in the bifurcation plots in Figure 3.4 and section 3.4 the model is clearly placed in different regimes for different values of the forcing parameter F. It would not be valid to asusme that the conclusion regarding the performance of the particle filter in one regime would be valid in other regimes.

Table 4.2 shows the ratio of the tracking MSE when misspecified values of the forcing parameters are used in the particle filer compared to when the true values are used. To place the model in the middle of the "fork" in the chaotic regime we use the true values $F = 3$ and $F = 8$.

The left table in Table 4.2 shows that the ratio of the tracking MSE when the true forcing parameter $F = 8$ is greater when it is assumed that $F = 9$ compared to $F = 7$. This is true irregardless of the observation scheme. This indicates that when the model is in a chaotic regime it is advantageous to perform tracking with the particle filter using an underestimated value of the forcing parameter rather than an overestimated one. For $d_y = 1$ the tracking is so bad that the relative decrease in performance by assuming lower or higher values of F rather than the true one is almost the same as when observing the state vector fully.

The right table shows that for observation schemes when most of the components are observed the particle filter performs slightly better when the overestimated value of the forcing parameter is used. However for $d_y = 2, 1$ is preferred to use an underestimated value. The table also shows that when using $F = 2$ for $d_y = 2$ the tracking is even better than when the true value is used. This suggests there is a great deal of Monte Carlo variance present even though the estimates are averaged over 1000 runs.

The table below is based Table A.3 and Table A.4 in Appendix. Overall, these tables show that the variance of the tracking MSE is very high when the system is not fully observed and it is much higher in the chaotic regime.

Table 4.2.: Ratio of tracking MSE averages over 1000 multiple runs when tracking $\{X_t\}_{t=0}^{100}$ with system noise $\sigma = 0.5$ and observation noise $\sigma = 1$. The ratio is calculated with the MSE for the misspecified forcing parameter in the numerator and with the true value in the denominator.

d_y	$\frac{MSE_{F=7}(\mathbf{x}_{0:100})}{MSE_{F=8}(\mathbf{x}_{0:100})}$	$\frac{MSE_{F=9}(\mathbf{x}_{0:100})}{MSE_{F=8}(\mathbf{x}_{0:100})}$
8	1.50	1.80
4	1.84	2.66
2	2.35	3.10
1	1.51	1.87

d_y	$\frac{MSE_{F=2}(\mathbf{x}_{0:t})}{MSE_{F=3}(\mathbf{x}_{0:t})}$	$\frac{MSE_{F=4}(\mathbf{x}_{0:t})}{MSE_{F=3}(\mathbf{x}_{0:t})}$
8	1.62	1.45
4	1.61	1.38
2	0.89	3.49
1	1.05	2.66

4.3.2. Parameter inference

Posterior distributions

The kernel density estimates of the marginal posterior distributions at time point $T = 40$ $p(F|y_{0:t})$ and $p(\sigma|y_{0:t})$ are plotted in Figure 4.6. They are all centered around the true value of the forcing parameter and slightly positively biased for σ. By comparing the plots in the first two columns there is no indication that the algorithm is sensitive to the choice between the two priors $\mathcal{IG}(1, 1/3)$ and $\mathcal{IG}(1, 1)$. The only difference is that the kernel density estimate in plot (e) is slightly peakier than in plot (f). The plots in the right column show that when only two components of the observation vector is observed the kernel density estimates are much flatter than for the other posterior distributions.

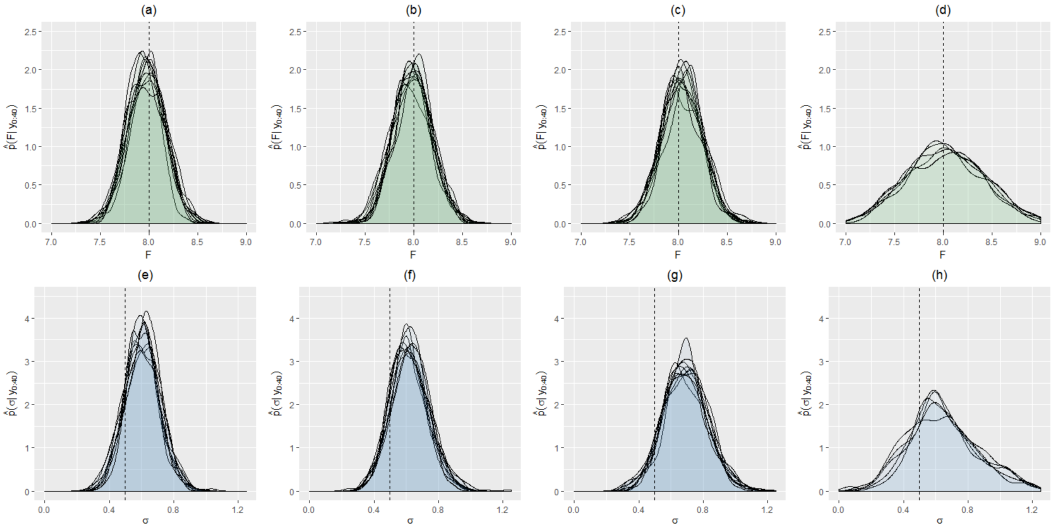

Figure 4.6.: Kernel density estimates of the marginal posterior distributions $p(F|\mathbf{y}_{0:40})$ and $p(\sigma|\mathbf{y}_{0:40})$ in the first and second row respectively. Configurations in columns from left to right: Benchmark, Prior $\mathcal{IG}(1, 1)$, Likelihood $\mathcal{C}$, Observations scheme $d_y = 2$.

In Figure 4.7 we plot the 95% credible interval for the average marginal posterior distributions over time. The plots in the second column show that the wider $\mathcal{IG}(1, 1)$ prior yields wider credible intervals during earlier time points than the $\mathcal{IG}(1, 1/3)$ prior in the first column. This effect of the wider prior has vanished at time point $T = 40$ where the credible intervals are roughly equally narrow. The marginal posteriors obtained using the Cauchy likelihood in plots (c) and (g) does not differ significantly from the benchmark configuration using a standard Normal in the first column. It is however seem to be more responsive when estimating σ suggested by the larger bump at time $t = 10$. In the case of the restricted observation scheme in the right column the credible intervals are wider. Judging by the MAP-estimate the evolution of these marginal posteriors is not as responsive as when the state vector is observed fully.

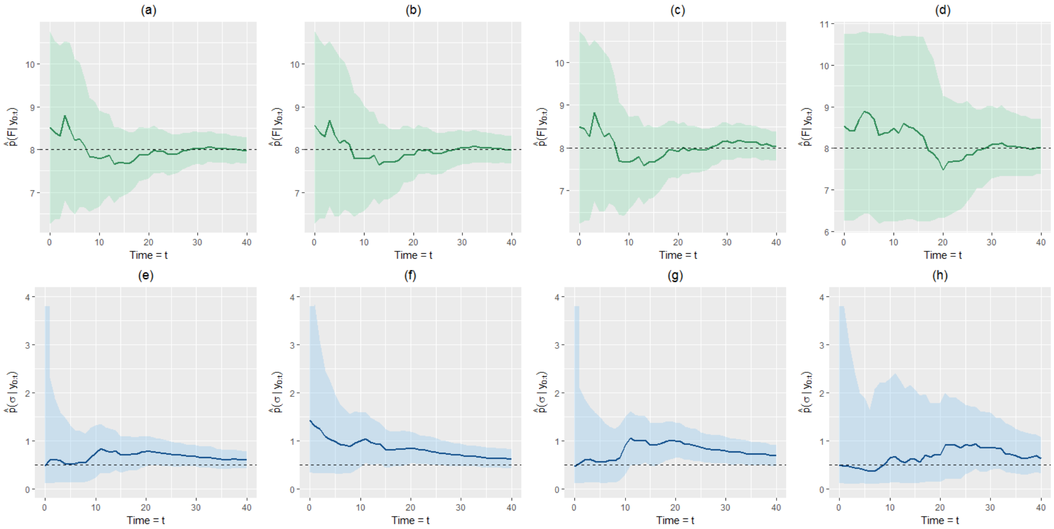

Figure 4.7.: Marginal posterior distributions $p(F|y_{0:t})$ and $p(\sigma|y_{0:t})$ over time in the first and second row respectively. Configurations in columns from left to right: Benchmark, Prior $\mathcal{IG}(1,1)$, Likelihood $\mathcal{C}$, Observations scheme $d_y = 2$.

MSE

The plots of the MSE in Figure 4.8 naturally show similar patterns to the marginal posterior distributions in Figure 4.7. Considering the MSE for F all the confidence intervals overlap each of the other estimates at every time point except when $d_y = 2$. For this case the MSE starts to improve after 15 time points which is in line with the fact that the marginal posterior distribution is very wide until then. The MSE for this observation scheme is significantly higher than for the other experimental configurations.

The MSE for σ is not surprisingly high in the beginning for the $\mathcal{IG}(1,1)$ prior. After 20 time points the difference between the different priors have vanished as the blue intervals overlap in plot (c). The use of the heavy-tailed likelihood results in a significantly higher MSE as the confidence intervals do not overlap in plot (d). It however overlaps the confidence interval for the configuration that only observes two components.

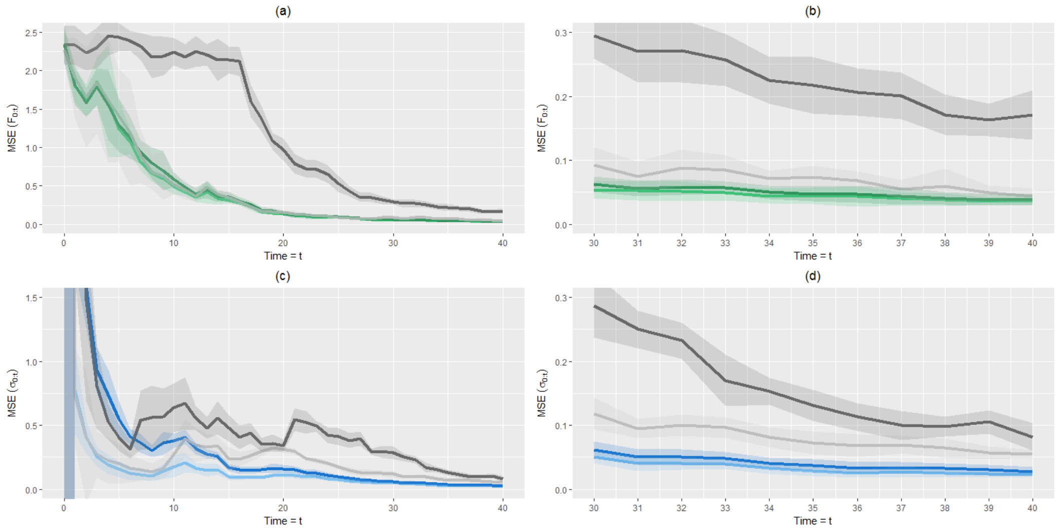

Figure 4.8.: $MSE(F_t)$ and $MSE(\sigma_t)$ in the first and second row respectively together with confidence bounds within 1.96 times its standard deviation. Configurations in colors: Light green: Benchmark, Dark green: Prior $\mathcal{IG}(1,1)$, Light grey: Likelihood $\mathcal{C}$, Light grey: Observations scheme $d_y = 2$.

Diagnostics

The use of different priors does not seem to effect the effective sample size, see plots (a) and (b) in Figure 4.9. On averge the effective sample size is lower at initial time points for the Cauchy likelihood in plot (c) on. The average acceptances rates in plots (e)-(g) gradually decrease. This indicates that the posterior distribution gradually from the time point of the initialisation gets closer and closer to its target distribution. Plot (d) shows that the effective sample size as expected on average is higher for the restricted observation scheme is compared to the others. As a consequence of the fact that the variance of the weights are smaller these diagnostic measures in the right column exhibit less Monte Carlo variance.

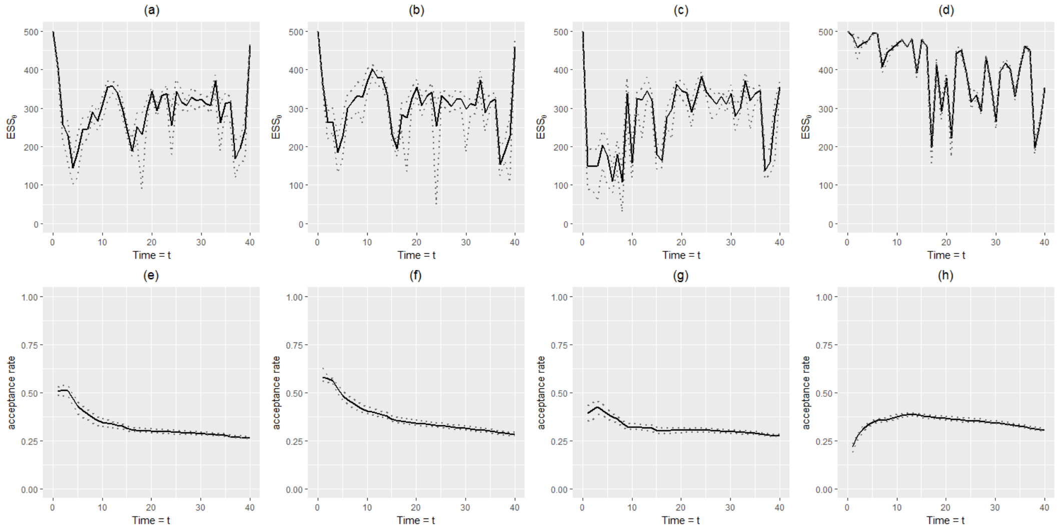

Figure 4.9.: ESS_θ and average acceptance rate over time in the first and second row respectively. Configurations in columns from left to right: Benchmark, Prior $\mathcal{IG}(1,1)$, Likelihood $\mathcal{C}$, Observations scheme $d_y = 2$. Solid and dotted lines correspond to average and minimum/maximum values respectively for the 10 runs.

Rejuvenation diagnostics

In Figure 4.10 we plot the values of the θ-particles before and after the rejuvenation for a typical run of the benchmark configuration. Most importantly, overall the dots do not lie on a straight line which would correspond to that no moves has been made. The dots are to a large extent randomly spread. This suggests that $M = 5$ move steps corresponding to roughly one actual move in every rejuvenation suffice to provide enough jitter. The posterior gets closer to its target and the acceptance rate gradually goes down. As an effect a straight line becomes more and more apparent as time increases.

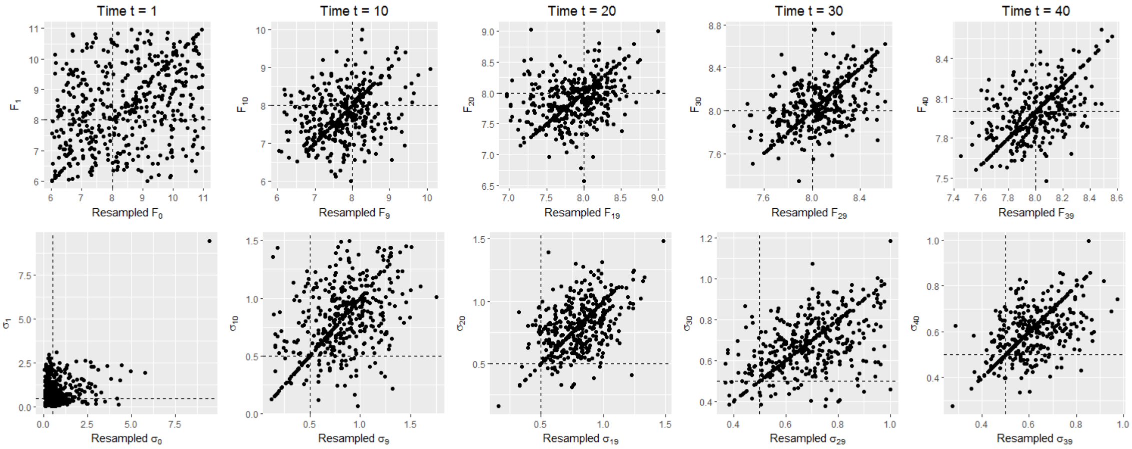

Figure 4.10.: Move steps at different time points for a typical run of the benchmark configuration. The moves for F and σ are respectively in the first and second row.

The observation made from the previous figure is further cemented by the plots in Figure 4.11. The plots show the average measure of decorrelation over time for the multiple runs using the benchmark configuration, together with its minimum and maximum. These values are clearly above the prescribed value of 0.01-0.05. We can also see that the measure of decorrelation gradually goes down as the average acceptance rate goes down. It should be noted that the plots in this section look very similar for the other experimental configurations and are therefore left out.

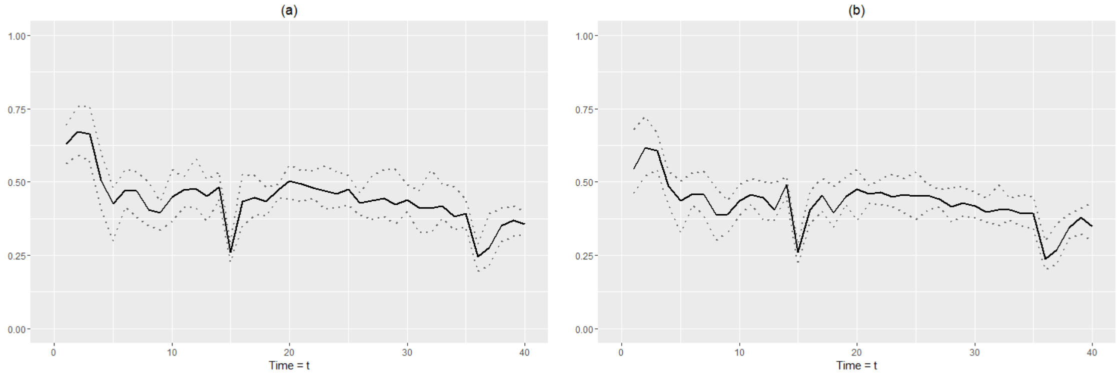

Figure 4.11.: Average decorrelation of the rejuvenation over time for the benchmark configuration. Dotted lines indicate minumum and maximum.

Graphics: Posterior evolution

The plots in 4.12 show the evolution of the kernel density estimates for the marginal posteriors over time. Each column shows a typical run for the different experimental configurations. Plots

(e) and (f) further illustrates the differences between the use of the different priors. Initially the kernel density estimates are very different but gradually they resemble each other. The density estimate using $\mathcal{IG}(1,1)$ again being slightly less peaky. In line with the observation from Figure 4.7 that, in addition to being less peaky, the marginal posterior for σ is prone to move towards higher values when the Cauchy likelihood is used.

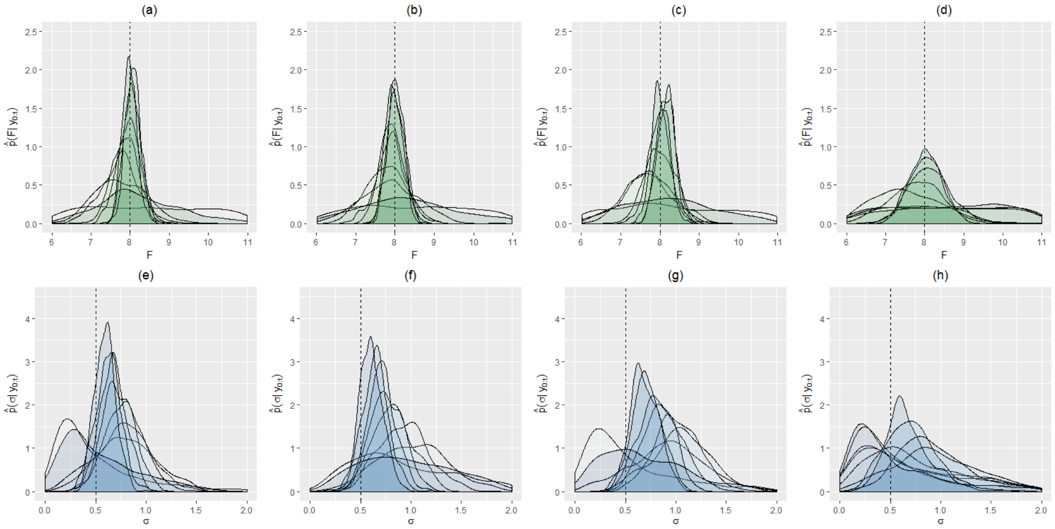

Figure 4.12.: Marginal posterior distributions for every fifth time point in a typical run, rows from top to bottom: $p(F|\mathbf{y}_{0:t})$, $p(\sigma|\mathbf{y}_{0:t})$. Configurations in columns from left to right: Benchmark, Prior $\mathcal{IG}(1,1)$, Likelihood $\mathcal{C}$, Observations scheme $d_y = 2$.

Figure 4.13 shows the evolution of the joint posterior distribution over time for a typical run for the benchmark configuration. After 30 time points the true value of the forcing parameter is in the center of the joint posterior while the true value of σ gets closer and closer.

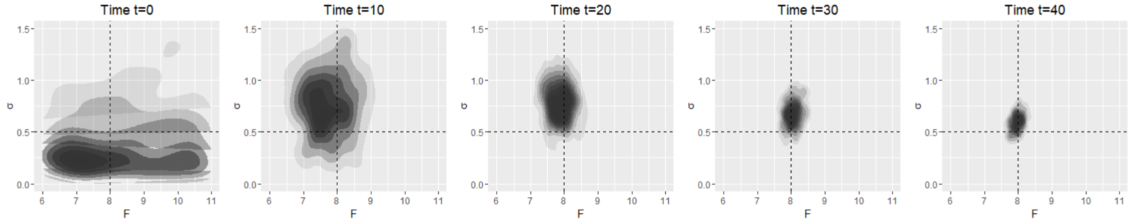

Figure 4.13.: Joint posterior distribution $p(F,\sigma|\mathbf{y}_{0:t})$ over time in a typical run of the benchmark configuration.

4.3.3. Parameter inference with adaptive tempering

The variance of the weights increase in higher dimensions. Hence the algorithm introduces more intermediate steps between the posterior distribution and its target distribution as can be seen in Table 4.3.

Table 4.3.: Average number of tempering steps for a single run of different dimensions d_x of SMC^2 with adaptive tempering. Every individual run assimilated ten data points $\{\mathbf{y}_i\}_{i=1}^{10}$ and the effective sample size was set to be above $N_{thresh} = 0.5N_\theta$.

d_x	8	16	32	64	96	128
κ	0	0.54	1.18	2.18	3.00	4.10

For higher dimensional systems at least one tempering step between every assimilation is done as seen in Figure 4.14. As more intermediate steps are done lower values of the cooling temperatures are used. When more than two tempering steps on average are used the cooling temperatures are roughly equally spaced. For lower dimensional cases the cooling temperatures increase with time, see the plot for the 16-dimensional case.

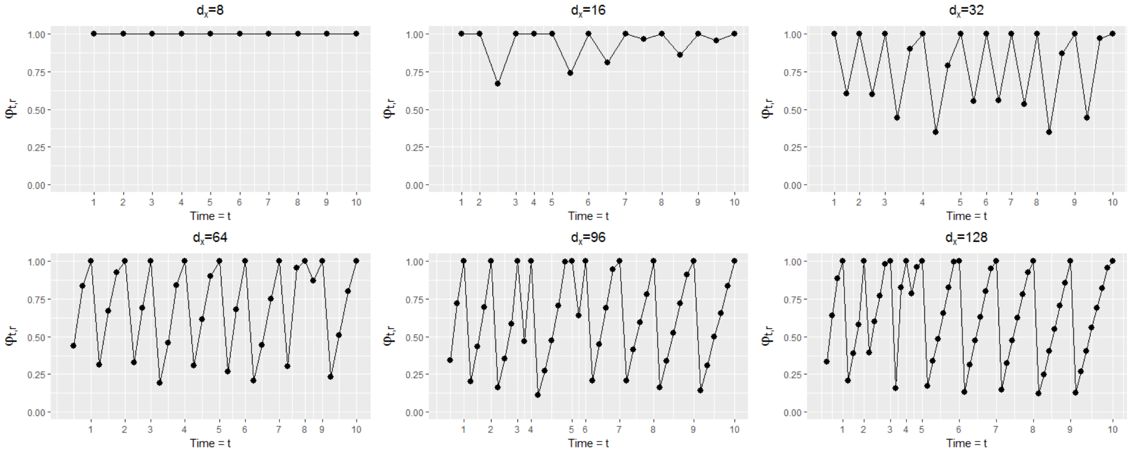

Figure 4.14.: Cooling temperatures over time in SMC^2 with adaptive tempering when assimilating ten observations $\{\mathbf{y}_t\}_{t=1}^{10}$. Plots for different sets of dimensions $d_x = 8, 16, 32, 64, 96, 128$.

The reults for the 64-dimensional Lorenz ’96 equations show that the tempering works when assimilating ten data points. It prevents the θ-weights from degenerating and the posterior distribution includes the true values for most of the time points until the very last ones, see 4.15. The results for the 96 dimensional case can be found in Appendix section A.3.

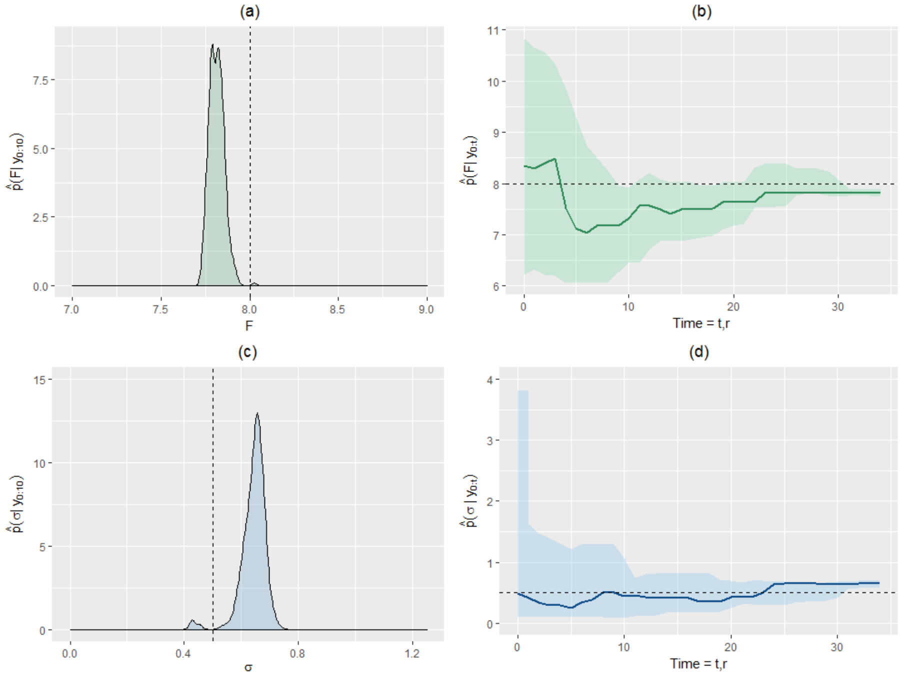

Figure 4.15.: Results for a single run of the SMC^2-algorithm with adaptive tempering, assimilating ten observations $\{\mathbf{y}_i\}_{i=1}^{10}$ where $d_x = 64$. Kernel density estimates of the marginal posterior distributions at time $T = 10$ in plots (a) $p(F|\mathbf{y}_{0:10})$ and (c) $p(\sigma|\mathbf{y}_{0:10})$. Marginal posterior distributions in plots (b) $p(F|\mathbf{y}_{0:t})$ and (d) $p(\sigma|\mathbf{y}_{0:t})$ over time.

Path degeneracy is an issue that rears its head as the number of dimensions increase. The transition kernel is put to test and manages to effectively reinsert diversity when the resampling filters out the particle population, illustrated in 4.16 for $d_x = 96$. By the end timepoint $T = 10$ the posterior is even less concentrated than in the 64-dimensional case. 128 dimensions turn out to be too challenging for our fixed computational cost and the we refrain from showing the results.

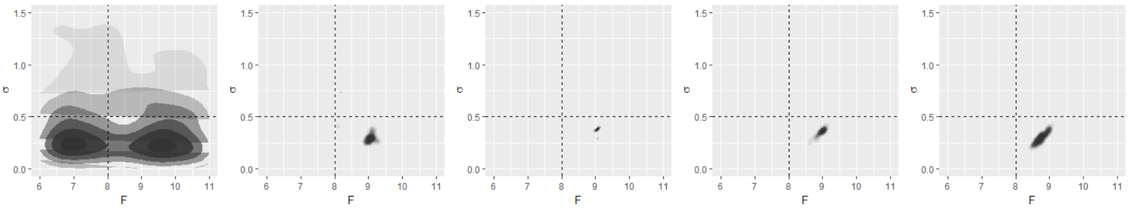

Figure 4.16.: Joint posterior distribution $p(F, \sigma|\mathbf{y}_{0:t})$ over time in SMC^2 with adapative tempering for $d_x = 96$ when assimilating $\{\mathbf{y}_t\}_{i=1}^{10}$.

To summarise, as the state vector are observed more sparsely the MSE increases exponentially whereas the variance of the normalising decrease. At the same time the increase in performance is relatively small when increasing the number of particles compared to when observing the system fully. An increase in the observation noise has a smaller deteriorating effect in a more sparse observation scheme. The standard deviation of the MSE increase more for smaller observation noise when the system goes from observing all component to fewer.

When the system is in the chaotic regime it is less bad to use an underestimated value of the forcing parameter rather than an overestimated. When the system is in the periodic regime the indications is not that unanimous. However, for or a very sparse observation scheme it is better to underestimate the forcing parameter. Overall, the variance of the MSE goes down with the forcing parameter.

The SMC^2-aglorithm manages to estimate the posterior distributions of the parameters indepedent of the configuration used. The posteriors for σ is however slightly biased and when the system is partially observed the posterior is flatter than for the other configurations, see Figure 4.6. Even though there initially is a difference for the different priors this difference have vanished after 20 time points.

The MSE for the forcing parameter are roughly equal for the different configurations except when the system is partially osberved, see Figure 4.8. The same can be said when inferring σ except the MSE is higher for heavier tailed likelihood. Observing fewer components seem to have more negative influence on the MSE for F rather than σ. For the Cauchy likeliood it is the other way around.

The acceptance rate overall gradually decerase which suggests that the joint posterior distributions get closer and closer to its target distributions.

Tempering prevents the variance inflation of the weights and the number of tempering steps increase as the dimensionality of the state vector increase. Together with the transition kernel it prevents the particle population from degenerating in up to 96 dimensional systems.

5. Discussion

5.1. Limitations

The MSE estimates in the tracking evaluation exhibit high Monte Carlo variance when all coordinates of the state vector is not observed. The assessment would preferably have been done for a larger number of multiple runs. The results suggests that the particle filter is of no good use if not all coordinates of the state vector are observed.

For the parameter inference in high-dimensional problems we only assimilate ten data points which is too few to perform consistent parameter estimation. As mentioned in chapter 1 the number of x-particles needed grows exponentially with the variance of the log likelihood. Bearing in mind that the variance of the normalising constant is expected to increase at least at a linear rate in time [34], we would need to increase the number of particles as we assimilate more data points at least at an exponential rate. This would be very computationally expensive. To tackle the path degeneracy in the high-dimensional problems we would need to increase the number of θ-particles too.

By sampling from the bootstrap proposal we did not use Importance Sampling so there is room for improvement with respect to the variance of the normalising constant.

5.2. Conclusions

When tracking the system of the Lorenz '96 model it is overall less bad to underestimate rather than overestimate the forcing parameter, see Table 4.2. The tracking MSE has lower variance when the system is less chaotic, see Table A.3 and Table A.4. An increase in the observation noise has a smaller deteriorating effect in a more sparse observation scheme, see Figure 4.5. When we observe less components of the system the tracking performance deteriorates drastically.

The resulting posteriors are wider for the wider priors and likelihoods, see Figure 4.7. The marginal posterior of σ converges at a slower rate than the marginal posteriors for F. The latter has for all configurations converged to its true values within 20-30 time points while the marginal posterior for σ has not converged at the end time point $t = 40$. Less informative likelihoods and observation schemes cause the convergence rate of the posterior to slow down, see Figure 4.7.

When using a wider prior the posterior is farther away from its target. This is suggested since the average acceptance rate is higher for the wider prior even though the scaling parameter ρ in the transition kernel is smaller, see Figure 4.9. When the distance between the posterior and its target is larger, a more informative likelihood distinguishes more between different particles than a heavier tailed one. This is indicated by the fact that for earlier time points the average acceptance rate is lower for the heavier tailed likelihood, see Figure 4.9. When we observe fewer components of the observation vector it appears that the posterior distribution is closer to its target distribution than it actually is, with a high effective sample size and a low acceptance rate in comparison to the other ones. As mentioned in 4.3.1 this is because as the observation system is in a lower dimensional space with lower variance of the weights as a result. For the same reason the algorithm exhibit less Monte Carlo variance aswell, see Figure 4.9. Although it provides information on how the performance of the SMC^2-algorithm changes over time, the absolute value of the effective sample size cannot be compared when d_y differ.

Observing fewer components of the state vector seem to effect the estimate of F in particular, see Figure 4.8. The uncertainty in the posterior for F is even higher than the prior allows it to be, indicated by the seemingly abrupt decrease in the MSE in the same figure.

On average it suffices to do sligtly more than one move per rejuvenation to jitter the particle population efficiently in the eight-dimensional case, see Figure 4.10 and Figure 4.11.

Since only single runs was made for the different high dimensional systems we cannot draw any general conclusions from these about the tempering schemes. We can however infer that the runs indicate that tempering improves the performance of the SMC^2-algorithm in high-dimensional spaces and prevents the variance inflation of the weights.

For 400 particles it effectively reduces the variance inflation of the θ-weights but as the dimensions increase the particle population suffer from path degeneracy. It can be noted that path degeneracy seem to result in similar sequences of cooling temperatures several timepoints in a row. This is due to the fact that the particle population is stuck in the same region, see the bottom right plot in Figure 4.14.

5.3. Ideas of extension and further work

This paper provides early results and the work can be extended in numerous ways.

As for the tracking performance we can also explore the changes in the variance of the normalising constant under the different regimes. It can also be investigated whether the tracking performance in higher dimensions is constant scaled for the number of dimensions. This can be extended for different observation noise and schemes to see if the performance is constant and if not infer at what rate the number of particles need to be increased. In a real world application we might have limited knowledge of the observation noise. We could look at the cost of misspecifying this quantity in a region around the true value. Since the underlyig signal is chaotic the particle filter will at no point be stable. The effect of this instability versus observation quality could be investigated further by using smaller and larger observation nosie.

When partially observing the state vector the algoirthm is fed by less information and there is naturally higher uncertainty in the estimated posterior distributions. The estimate of the posterior for the forcing parameter is however more negatively influenced than the posterior for the system noise. By adopting a less informative prior for the system noise we could see whether this is a result of using a more informative prior for σ than for F.

We need the system noise to be informtive but not too informative since large jumps in the distribution will require a large amount of particles or a better importance proposals for the states. In any case it would be interesting to look at different importance proposals both in terms of state estimation and parameter inference. In case the importance proposal significantly would increase the variance of the normalising constant we might be able to decrease the number of particles and thereby decrease the computational cost drastically in the SMC^2-algorithm.

We could assimilate more data points and see whether the bias in the marginal posterior distributions stems from observing to few observations. It would be interesting to see whether the posteriors resemble the same normal approximations as more observations are assimilated. A natural extension of this would be to look at what rate the posterior distributions converge to this normal approximatons for different priors and likelihoods, [35].

In a real world setting we might not be able to make any assumptions regarding the observation noise and the likelihood. Such scenarios might call approximations of the likelihood for which we could use Approximate Bayesian Computation (ABC) methods [36].

To improve the SMC^2-algorithm we can instead of assuming the same prior for the inital state try to infer this with the particle filter too. We can make use of the the marginal likelihoods in the particle filter and associate all estimates with our simulated initial values of $\mathbf{x}_0$. Another approach would be to to assume that the observation of the initial state $\mathbf{y}_0$ is available and make use of the assumed properties of the likelihood.

We approached the curse of dimensionality with tempering but we could also try the Equivalent-Weights Particle Filter, [4, 37, 38, 39, 40]. Yet another approcah would be to use ideas of dimensionality reduction and Principal Component Analysis (PCA), [41, 42].

Given the positive results of tempering in high dimensions it may be worth exploring whether an increase in the number of particles will prevent the degeneracy for dimensions exceeding 100. To extend this even further one could compare the performance of increasing the number of particles versus the threshold N_{thresh}.

Since the cooling temperatures are roughly equally spaced when more than two intermediate steps are used we could try and combine the adaptive scheme with a deterministic one. Say we determine only the first cooling temperature between two time points using the adaptive scheme. Then the following are chosen so that set the set of temperatures are equally spaced up to the target distribution.

In the 128-dimensional system the transition kernel did not reinsert enough diversity into the particle population. We could construct other kernels and compare their performance in this high-dimensional setting. When comparing the performance of the transition kernels we could in addition to using decorrelation measures and scatterplots use ideas similar to the rate of separation of trajectories.

Bibliography

[1] E. Lorenz, "Predictability: a problem partly solved," in *Seminar on Predictability, 4-8 September 1995*, vol. 1, (Shinfield Park, Reading), pp. 1–18, ECMWF, ECMWF, 1995.

[2] L. Murray, "Bayesian state-space modelling on high-performance hardware using libbi," vol. 67, 06 2013.

[3] C. Snyder, T. Bengtsson, P. Bickel, and J. Anderson, "Obstacles to high-dimensional particle filtering," *Monthly Weather Review*, vol. 136, pp. 4629–4640, 2008.

[4] P. Van Leeuwen, "Particle filtering in geophysical systems," *Monthly Weather Review*, vol. 137, pp. 4089–4114, 2009.

[5] R. Neal, "Annealed importance sampling," *Statistics and Computing*, vol. 11(2), pp. 125–139, 2001.

[6] N. Chopin, "A sequential particle filter for static models," *Biometrika*, vol. 89, p. 539–552, 2002.

[7] C. Jarzynski, "Nonequilibrium equality for free energy differences," *Phys. Rev. Lett.*, vol. 78, pp. 2690–2693, Apr 1997.

[8] P. Del Moral, A. Doucet, and A. Jasra, "Sequential Monte Carlo samplers," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, vol. 68, no. 3, pp. 411–436, 2006.

[9] N. Kantas, A. Beskos, and A. Jasra, "Sequential Monte Carlo methods for high-dimensional inverse problems: A case study for the Navier-Stokes equations," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 2(1), pp. 464–489, 2014.

[10] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods.* Springer. New York. Chicago, 2004.

[11] J. S. Liu, *Monte Carlo Strategies in Scientific Computing.* New York, 2001.

[12] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The journal of chemical physics*, vol. 21(6), pp. 1087–1092, 1953.

[13] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57(1), pp. 97–109, 1970.

[14] A. E. Gelfand and A. F. Smith, "Sampling-based approaches to calculating marginal densities," *Journal of the American statistical association*, vol. 85(410), pp. 398–409, 1990.

[15] A. Smith and G. Roberts, "Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society*, vol. Series B (Methodological), pp. 3–23, 1993.

[16] L. Tierney, "Markov chains for exploring posterior distributions," *The Annals of Statistics*, vol. 0, pp. 1701–1728, 1994.

[17] J. Liu and R. Chen, “Sequential Monte Carlo methods for dynamic systems,” *Journal of the American Statistical Association*, vol. 93(443), pp. 1032–1044, 09 1998.

[18] A. Doucet, S. Godsil, and C. Andrieu, “On sequential Monte Carlo sampling methods for Bayesian filtering,” *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[19] A. Doucet, N. De Freitas, and N. Gordon, eds., *Sequential Monte Carlo Methods in Practice.* Springer-Verlag, 2001.

[20] G. Kitagawa, “Monte Carlo filter and smoother for non-Gaussian nonlinear state space models,” vol. 1, pp. 1–25, 03 1996.

[21] P. Del Moral, “Feynman-kac formulae,” in *Feynman-Kac Formulae*, pp. 47–93, Springer, 2004.

[22] N. Gordon, D. Salmond, and A. Smith, “Novel approach to nonlinear/non-Gaussian Bayesian state estimation,” *IEE Proceedings F - Radar and Signal Processing*, vol. 140(2), pp. 107–113, 1993.

[23] W. R. Gilks and C. Berzuini, “Following a moving target - Monte Carlo inference for dynamic bayesian models,” *J. Royal Stat. Soc. B*, vol. 63(1), pp. 127–146, 2001.

[24] C. Andrieu, A. Doucet, and R. Holenstein, “Particle Markov chain Monte Carlo methods,” *J. R. Statist. Soc. B*, vol. 72(3), p. 269–342, 2010.

[25] N. Chopin, P. Jacob, and O. Papaspiliopoulos, “Smc^2: A sequential Monte Carlo algorithm with particle Markov chain Monte Carlo updates,” *J. Royal Stat. Soc. B*, vol. 75, pp. 397–426, 2013.

[26] A. Jasra, D. Stephens, A. Doucet, and T. Tsagaris, “Inference for Lévy-driven stochastic volatility models via adaptive sequential Monte Carlo,” *Scandinavian Journal of Statistics*, vol. 38(1), p. 1–22, 2011.

[27] A. Stuart, “Inverse problems: A Bayesian perspective,” *Acta Numerica*, vol. 19(1), pp. 451–559, 2010.

[28] S. Cotter, G. Roberts, A. Stuart, and D. White, “MCMC methods for functions: Modifying old algorithms to make them faster,” *Statistical Science*, vol. 28, no. 3, pp. 424–446, 2013.

[29] R. Neal, J. M. editors, Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, “Regression and classification using Gaussian process priors,” *Bayesian Statistics 6: Proceedings of the Sixth Valencia International Meeting*, vol. 6, pp. 475–491, 1998.

[30] B. Øksendal, *Stochastic differential equations.* Springer Berlin Heidelberg, 2003.

[31] C. Runge, “Ueber die numerische auflösung von differentialgleichungen,” *Math. Ann*, vol. 46, p. 167–178, 1895.

[32] W. Kutta, “Beitrag zur naherungsweisen integration von differentialgleichungen,” *Z. Math. und Phys.*, vol. 46, pp. 435–453, 1901.

[33] P. C. Parks, “A. m. Lyapunov’s stability theory—100 years on,” vol. 9, pp. 275–303, 01 1992.

[34] F. Cerou, P. Del Moral, and A. Guyader, “A nonasymptotic theorem for unnormalized Feynman-Kac particle models,” *Ann. Inst. Henri Poincaré*, vol. 47(3), pp. 629–649, 2011.

[35] R. Michel and C. Hipp, “On the Bernstein von Mises approximation of posterior distributions,” *The Annals of Statistics*, vol. 4(5), pp. 972–980, 1976.

[36] A. Jasra, S. Singh, J. Martin, and E. McCoy, "Filtering via approximate Bayesian computation," *Statistics and Computing*, vol. 22, pp. 1223–1237, Nov 2012.

[37] P. Van Leeuwen, "Nonlinear data assimilation in geosciences: an extremely efficient particle filter," *Quarterly Journal of the Royal Meteorological Society*, vol. 136, pp. 1991–1999, 2010.

[38] M. Ades and P. J. Van Leeuwen, "An exploration of the equivalent weights particle filter," *Quartely Journal of Meteorology*, vol. 139, pp. 820–840, 2013.

[39] M. Ades and P. J. Van Leeuwen, "The equivalent weights particle filter in a high 662 dimensional system," *Quartely Journal of Meteorology*, vol. 139, pp. 820–840, 2014a.

[40] M. Ades and P. J. Van Leeuwen, "The effect of the equivalent-weights particle flter on dynamical balance in a primitive equation model," *Quartely Journal of Meteorology*, vol. 139, pp. 820–840, 2014a.

[41] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2 (6), pp. 559–572, 1901.

[42] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," vol. 2, pp. 37–52, 08 1987.

A. Appendix

A.1. Code

The C++ code for the main implementations can be found on GitHub, https://github.com/cedergren89.

A.2. Algorithms

A.2.1. SMC^2 with adaptive tempering

Algorithm 11 SMC^2 with adaptive tempering

```
At $t = 0$
Initialise $\theta_0^j \sim p_\theta(\cdot)$ for each $\theta$-particle $j = 1, .., N_\theta$.

For each time step $t = 0, ...T$

  1. **Propagate and resample x**
     For each $\theta$-particle $j = 1, .., N_\theta$
       a) if $t = 0$
            i. Initialise $x_0^{i,j} \sim q_{x_0}(\cdot|y_0)$.
           ii. Weight $x$-particle $w_{x,0}(x_0^{i,j}) = \frac{1}{N_x}$.
          iii. Compute incremental $\theta$-weight $\hat{p}_0(y_0|\theta_0^j) = \frac{1}{N_x}\sum_{i=1}^{N_x} w_{x_0}^{i,j}$.
           iv. Set importance $\theta$-weights $w_{\theta,0}^j = \hat{p}_0(y_0|\theta_0^j)$ and $W_{\theta,0}^{i,j} = \frac{1}{N_\theta}$.
            v. Set evidence $Z_0^j = \hat{p}_0(y_0|\theta_0^j)$.
       b) if $t \geq 1$
            i. Compute effective sample size $ESS_{x,t-1} = \frac{1}{\sum_{i=1}^{N_x}(W_{x,t-1}^{i,j})^2}$.
           ii. if $ESS_{x,t-1} < \gamma N_\theta$
                 A. Sample index $a_t^{i,j} \sim \mathcal{R}(\{W_{x,t-1}^{i,j}\}_{i=1}^N)$ of the ancestor to particle $i$.
                 B. Set $W_{x,t-1}^{i,j} \leftarrow \frac{1}{N_x}$ and $x_{0:t-1}^{i,j} \leftarrow x_{0:t-1}^{a_t^{i,j},j}$.
          iii. Propagate $x_t^{i,j} \sim p(\cdot|x_{0:t-1}^{i,j}, \theta_{t,0})$ and set $x_{0:t}^{i,j} = (x_{0:t-1}^{i,j}, x_t^{i,j})$.
           iv. Weight $x$-particles $w_{x,t}^{i,j}(x_{t-1}^{i,j}, x_t^{i,j}) = \frac{f_{\theta_{t,0}^j}(x_t^{i,j}|x_{t-1}^{i,j})g_{\theta_{t,0}^j}(y_t|x_t^{i,j})}{q_{\theta_{t,0}^j}(x_t^{i,j}|y_t, x_{t-1}^{i,j})}$.
            v. Compute importance $x$-weight $w_{x,t}^{i,j} = W_{x,t-1}^{i,j} w_{x,t}(x_{t-1}^{i,j}, x_t^{i,j})$ and normalise $W_{x,t}^{i,j} = \frac{w_{x,t}^{i,j}}{\sum_{i=1}^N w_{x,t}^{i,j}}$.
            vi. Compute incremental $\theta$-weight $\hat{p}_t(y_t|y_{0:t-1}, \theta^j_{t,0}) = \frac{1}{N_x}\sum_{i=1}^{N_x} w^{i,j}_{x,t}$.

2. Set $r = 0$ and $\phi_{t,0} = 0$.
   While $\phi_{t,r} < 1$ :
    a) Compute temperatures
       Set $r \leftarrow r + 1$.
       if $\phi \in (\phi_{t,r-1}, 1]\mathrm{ESS}_{t,r}(\phi) > N_{thresh}$, set $\phi_{t,r} = 1$.
       else compute $\phi_{t,r}$ such that $\mathrm{ESS}_{\theta,t,r}(\phi_{t,r}) \approx N_{thresh}$.

    b) Compute statistics for $\theta$
        i. For each $\theta$-particle $j = 1, .., N_\theta$
              Compute $\theta$-weights $W^j_{\theta,t,r} = \dfrac{W^j_{\theta,t,r-1}\hat{p}_t(y_t|y_{0:t-1}, \theta^j_{t,r-1})^{\phi_{t,r}-\phi_{t,r-1}}}{\sum_{j'=1}^{N_\theta} W^j_{\theta,t,r-1} p_t(y_t|y_{0:t-1}, \theta^{j'}_{t,r-1})^{\phi_{t,r}-\phi_{t,r-1}}}$
           where $W^j_{\theta,t,0} = W^j_{\theta,t-1}$
        ii. Effective sample size $ESS_{\theta,t,r} = \dfrac{1}{\sum_{j=1}^{N_\theta}(W^j_{\theta,t,r})^2}$.
           Moment estimates
           $\hat{\mu}_{t,r} = \sum_{j=1}^{N_\theta} W^j_{\theta,t,r}\theta^j_{t,r-1}$ and $\hat{\Sigma}_{t,r} = \sum_{j=1}^{N_\theta} W^j_{\theta,t,r}(\theta^j_{t,r-1} - \hat{\mu}_{t,r})(\theta^j_{t,r-1} - \hat{\mu}_{t,r})^T$.
    c) Resample $\theta$ (if required)
       For each $\theta$-particle $j = 1, .., N_\theta$
        i. Sample index of $\theta^j_{t,r-1}$-particle $o^j_t \sim \mathcal{R}(\{W^j_{\theta,t,r}\}_{j=1}^{N_\theta})$:
        ii. Set:
           ∘ $W^{i,j}_{\theta,t,r} \leftarrow \dfrac{1}{N_\theta}$.
           ∘ $\left(\theta^j_{t,r-1}, Z^j_{t-1}, \left\{x^{i,j}_{0:t}, a^{i,j}_{0:t-1}, W^{i,j}_{x,t}\right\}_{i=1}^{N_x}\right) \leftarrow \left(\theta^{o^j_t}_{t,r-1}, Z^{o^j_t}_{t-1}, \left\{x^{o^j_t,j}_{0:t}, a^{o^j_t,j}_{0:t-1}, W^{o^j_t,j}_{x,t}\right\}_{i=1}^{N_x}\right)$.
    d) Rejuvenate $\theta$
       Sample $\theta^j_{t,r} \sim \mathcal{K}\left(\cdot| \left(\theta^j_{t,r-1}, Z^j_{t-1}, \hat{p}_{t,r}(y_t|y_{0:t-1}, \theta^j_{t,r-1}), \left\{x^{i,j}_{0:t}, a^{i,j}_{0:t-1}, W^{i,j}_{x,t}\right\}_{i=1}^{N_x}\right)\right)$ according to Algorithm #.
3. Set for next time step
    a) For each $\theta$-particle $j = 1, .., N_\theta$;
       ∘ $\theta^j_{t+1,0} = \theta^j_{t,r} (:= \theta^j_t)$
       ∘ $W^j_{\theta,t} = W^j_{\theta,t,r}$
       ∘ $Z^j_t = \hat{p}(y_t|y_{0:t-1}, \theta^j_{t,r})Z^j_{t-1}$
    b) ∘ $ESS_{\theta,t+1,0} = ESS_{\theta,t,r} (:= ESS_{\theta,t})$
```

A.2.2. 4th order Runge Kutta scheme

Consider the system of ODEs:

$$\frac{df(x,t)}{dt} = f(x,t).$$

The 4th order Runge Kutta scheme is a numerical integrating scheme solving the initial value problem

$$x = f(x,t), \quad x(t_0) = x_0$$

with step size Δt for $k = 0, \Delta t, 2\Delta t, ..., T/\Delta t$.

Algorithm 12 4th order Runge Kutta

```
Set $t_0 = 0$;
For $k$ in $0 : T/\Delta t$:
    1. $m_1 = f(x_k, t_k)$
    2. $m_2 = f(t_k, x_k + m_1 \frac{\Delta t}{2})$
    3. $m_3 = f(t_k, x_k + m_2 \frac{\Delta t}{2})$
    4. $m_4 = f(t_k, x_k + m_3 \Delta t)$
    5. $x_{k+1} = \frac{1}{6}(m_1 + m_2 + m_3 + m_4)$
    6. $t_{k+1} = t_k + \Delta t$
In the case when the system of ODEs has an additive diffusion process then a standard Wiender increment $\Delta W_k \sim \mathcal{N}(0, \Delta t)$ is added in step 5 above.
```

A.2.3. Lyapunov's exponents and rate of separation

Algorithm 13 Lyapunov's exponent and rate of separation

```
1. Simulated initial true state, $x_d^0 \sim \mathcal{U}(-3, 3)$.
2. Simulate initial infinitesimally distance $(|\delta Z_0|) = e_d^1 \sim \mathcal{N}(0, 0.001^2)$.
3. Set perturbated state $\dot{x}_d^1 = x_d^1 + e_d^1$.
4. For $n$ in $1 : N$ repetitions:
    a) Integrate 3.1 forward in time for T steps using the 4th order Runge Kutta scheme in A.2.2.
    b) For $t$ in $1 : T$ time steps:
       Calculate Euklidean distance $e_n^t = \sqrt{\sum_{d=1}^{D}(x_d^t - \dot{x}_d^t)^2}$.
    c) Set $x_d^0 \leftarrow x_d^T$, $\dot{x}_d^0 \leftarrow \dot{x}_d^T$.
5. For $t$ in $1 : T$ time steps:
   Estimate infinitesmall distance $\widehat{|\delta Z_t|} = \frac{1}{N} \sum_{i=1}^{N} \ln e_n^t$.
```

6. Plot $\widehat{\ln \frac{|\delta Z_t|}{|\delta Z_0|}}$ against t.

7. Estimate λ by approximating $\hat{\lambda} = \lim_{t \to 0} \frac{1}{t} \widehat{\ln \frac{|\delta Z_t|}{|\delta Z_0|}}$.

A.3. Figures

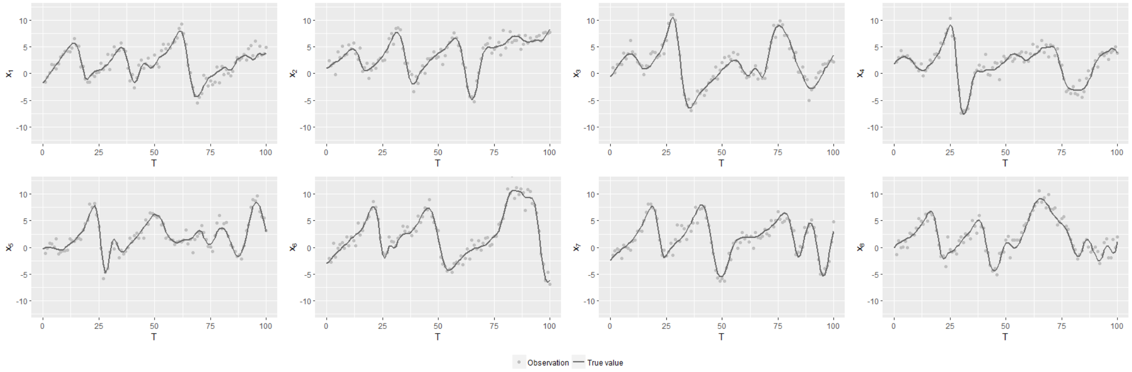

Figure A.1.: Trajectories and observations in an eight-dimensional Lorenz '96 model from $t = 0$ to $T = 100$. The coordinates $d = 1, \cdots, 8$ are plotted rowwise from left ro right. The dark grey lines correspond to the true state and the grey dots the observations.

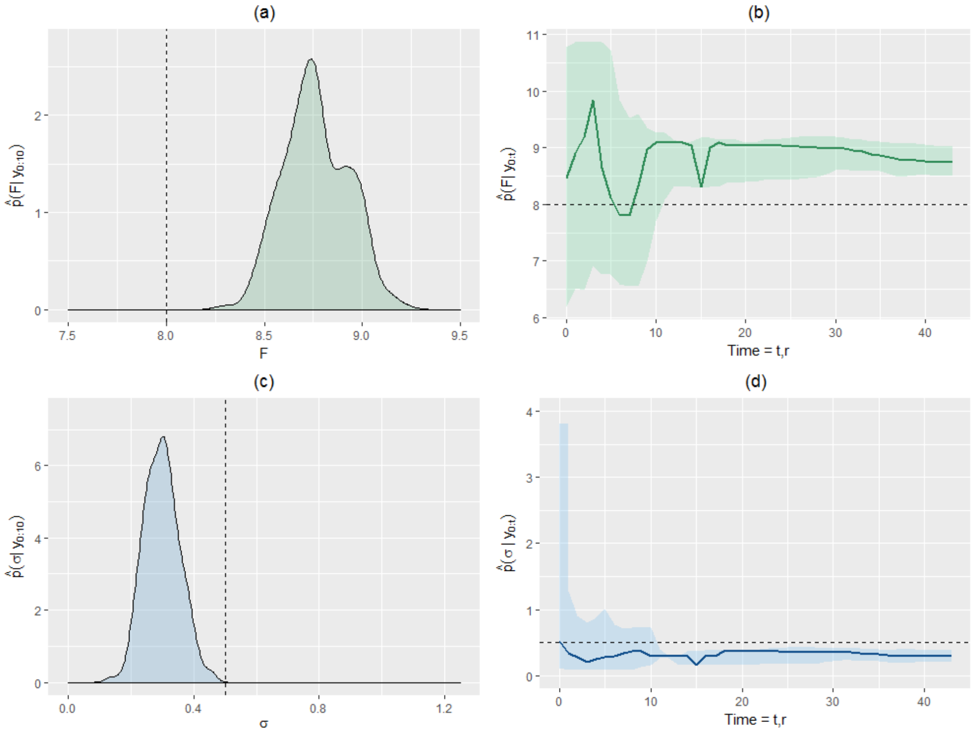

Figure A.2.: Results for a single run of the SMC^2-algorithm with adaptive tempering, assimilating ten observations $\{\mathbf{y}_i\}_{i=1}^{10}$ where $d_x = 96$. Kernel density estimates of the marginal posterior distributions at time $T = 10$ in plots (a) $p(F|y_{0:10})$ and (c) $p(\sigma|y_{0:10})$. Marginal posterior distributions in plots (b) $p(F|y_{0:t})$ and (d) $p(\sigma|y_{0:t})$ over time.

A.4. Tables

A.4.1. Tracking MSE

Table A.1.: $MSE(x_{0:100})$ averaged over 1000 runs when the observation noise is $\sigma_\varepsilon = 1$ for different number of particles and observation schemes. The values in brackets denotes the standard deviation.

	N_x				
d_y	1000	2000	3000	4000	5000
8	149.7 (17.4)	137.0 (11.2)	132.4 (8.4)	129.9 (7.4)	128.3 (6.5)
4	1072.3 (1817.0)	517.1 (833.8)	427.8 (679.5)	374.4 (354.5)	330.3 (124.0)
2	2117.1 (2856.3)	1085.9 (1648.1)	792.8 (1080.4)	614.2 (927.7)	552.7 (650.5)
1	5088.1 (3549.2)	3429.8 (2724.6)	2566.3 (1933.0)	2251.7 (1295.6)	2116.8 (1072.0)

Table A.2.: $MSE(x_{0:100})$ averaged over 1000 runs when the observation noise is $\sigma_\varepsilon = 2$ for different number of particles and observation schemes. The values in brackets denotes the standard deviation.

	N_x				
d_y	1000	2000	3000	4000	5000
8	531.1 (1140.9)	341.8 (131.1)	320.8 (19.4)	317.3 (17.0)	316.0 (15.8)
4	955.6 (1422.9)	524.5 (557.6)	515.4 (191.5)	505.2 (28.8)	512.0 (25.4)
2	3309.3 (3210.4)	1909.7 (1183.7)	1686.5 (656.4)	1594.3 (319.4)	1556.7 (220.8)
1	5371.3 (2521.4)	4192.1 (1310.3)	3849.8 (887.1)	3739.6 (717.3)	3658.2 (586.7)

Table A.3.: $MSE(x_{0:100})$ averaged over 1000 runs when the observation noise is $\sigma_\varepsilon = 1$ and the forcing parameter is $F = 8$. Shown for different combinations of values of the forcing parameter in the particle filter and observation schemes. The values in brackets denotes the standard deviation.

	F		
d_y	7	8	9
8	205.5 (40.1)	137.0 (11.2)	246.2 (153.1)
4	949.9 (1076.3)	517.1 (833.8)	1377.4 (2142.7)
2	2555.9 (2955.7)	1085.9 (1648.1)	3364.8 (4055.3)
1	5176.0 (2271.5)	3429.8 (2724.6)	6407.9 (4035.8)

Table A.4.: $MSE(x_{0:100})$ averaged over 1000 runs when the observation noise is $\sigma_{\varepsilon} = 1$ and the forcing parameter is $F = 3$. Shown for different combinations of values of the forcing parameter in the particle filter and observation schemes. The values in brackets denotes the standard deviation.

	F		
d_y	2	3	4
8	219.1 (15.0)	134.9 (12.0)	195.4 (13.0)
4	540.5 (104.24)	335.22 (110.18)	462.3 (152.2)
2	726.0 (60.6)	814.3 (175.2)	2839.1 (394.1)
1	1137.5 (138.53)	1078.9 (148.4)	2873.0 (398.3)