Joint State and Parameter Estimation in Temporal Models

Yusuf Erol
Stuart J. Russell
Laurent El Ghaoui

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2018-28
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2018/EECS-2018-28.html

May 7, 2018

Joint State and Parameter Estimation in Temporal Models

by Yusuf Bugra Erol

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II.**

Approval for the Report and Comprehensive Examination:

Committee:

Professor Stuart Russell
Research Advisor

(Date)

* * * * * * *

Professor Laurent El Ghaoui
Second Reader

(Date)

Abstract

Many problems in science and engineering involve the modeling of dynamic processes using state-space models (SSMs). Online parameter and state estimation—computing the posterior probability for both static parameters and dynamic states, incrementally over time–is crucial for many applications. Many sequential Monte Carlo algorithms have been proposed for this problem; some apply only to restricted model classes, while others are computationally expensive. We propose two new algorithms, namely, the extended parameter filter and the assumed parameter filter, that try to close the gap between computational efficiency and generality. We compare our new algorithms with several state-of-the-art solutions on many benchmark problems. Finally, we discuss our work on joint state and parameter estimation for physiological models in intensive-care medicine.

Contents

1 Introduction

Many problems in scientific studies and real-world applications involve modeling of dynamic processes, which are often modeled by temporal models, namely state space models (SSMs) [Elmohamed et al., 2007, Arora et al., 2010]. Online parameter and state estimation –computing the posterior probability for both (static) parameters and (dynamic) states, incrementally over time– is crucial for many applications such as simultaneous localization and mapping [Montemerlo et al., 2002], object tracking [Ristic et al., 2004] and 3D design suggestion [Ritchie et al., 2015].

Dynamic Bayesian networks are widely used to model the processes underlying sequential data such as speech signals, financial time series, genetic sequences, and medical or physiological signals [Murphy, 2002]. State estimation or filtering—computing the posterior distribution over the state of a partially observable Markov process from a sequence of observations—is one of the most widely studied problems in control theory, statistics and AI. Exact filtering is intractable except for certain special cases (linear–Gaussian models and discrete HMMs), but approximate filtering using the *particle filter* (a sequential Monte Carlo method) is feasible in many real-world applications [Gordon et al., 1993, Arulampalam et al., 2002, Doucet and Johansen, 2011].

In the machine learning context, model parameters may be represented by static parameter variables that define the transition and sensor model probabilities of the Markov process, but do not themselves change over time. The posterior parameter distribution (usually) converges to a delta function at the true value in the limit of infinitely many observations.

Unfortunately, particle filters fail for such models: the algorithm samples parameter values for each particle at time $t = 0$, but these remain fixed; over time, the particle resampling process removes all but one set of values; and these are highly unlikely to be correct. The degeneracy problem is especially severe in high-dimensional parameter spaces, whether discrete or continuous. Hence, although learning requires inference, the most successful inference algorithm for temporal models is inapplicable.

Kantas et al. [2009, 2015] and Carvalho et al. [2010] describe several algorithms that have been proposed to solve this degeneracy problem, but the issue remains open because known algorithms either suffer from bias or computational inefficiency. Existing algorithms are either restricted to a particular class of models, such as the Storvik filter [Storvik, 2002] and the Liu-West filter [Liu and West, 2001], or very expensive in time complexity, such as particle MCMC [Andrieu et al., 2010], which utilizes an expensive MCMC kernel over the parameter space and typically requires a very large number of MCMC iterations to converge.

This thesis is structured as follows: In the following section, we will briefly discuss background and relevant material in the sequential Monte Carlo literature. Section 3 introduces the extended parameter filter (EPF) [Erol et al., 2013a] which generalizes the Storvik filter by introducing a larger model class. EPF utilizes a polynomial approximation scheme in order to handle arbitrary

models and performs well compared to many state-of-the-art solutions. Section 4 introduces the assumed parameter filter (APF) [Erol et al., 2017]. Instead of manual polynomial expansions which might be costly or hard to derive, we resort to assumed density filtering [Maybeck, 1982, Boyen and Koller, 1998, Opper and Winther, 1998] by projecting intractable posterior densities into convenient exponential family densities via KL-divergence minimization. APF streams data and is a nearly-black-box algorithm: when an appropriate approximate distribution for the parameter is chosen, APF can be applied to any SSM from which one can sample from and for which one can compute evidence likelihoods. We emphasize the nearly-black-box property of APF by developing it as an automatic inference engine for a probabilistic programming language. Experiment results show that APF converges much faster than existing algorithms on a variety of models.

Section 5 introduces an important application we are interested in which is model-based probabilistic inference for intensive care medicine. Modern intensive care units (ICUs) utilize a multitude of instrumentation devices to provide measurements of various important physiological variables and parameters. While data are valuable, understanding the data and acting on them is what yields the benefits in terms of improved health outcomes. Due to the uncertainty in our knowledge of the patient's physiology and the partial and noisy/artifactual nature of the observations we adopt a probabilistic, model-based approach. The core of the approach involves calculating a posterior probability distribution over a set of unobserved state variables, given a stream of data and a probabilistic model of patient physiology and sensor dynamics. The probability estimate of the state, which includes various physiological and pathophysiological variables, provides a diagnosis on which the nurse or the physician can act. The proposed approach is also capable of detecting artifacts, sensor failures, drug maladministration and other various problems in the ICU setting.

Here's a succinct description of our approach. We describe the patient's physiology and the sensor dynamics as a probabilistic model using the dynamic Bayesian network framework. We use existing works on human physiology to describe how the state of the patient evolves over time and employ a nontrivial sensor model that is capable of explaining various artifactual readings in the ICU setting. Then the main task of the ICU monitoring system is estimating the state of the patient accurately, given a sequence of observations. Due to the nonlinear, non-Gaussian behavior of our model, exact inference is intractable. Hence we resort to approximate inference via sequential Monte Carlo (SMC) methods described in this thesis. We present experimental results on simulated data for intracranial hemodynamics and real data for blood pressure monitoring. This work is the result of a collaboration with SFGH BASIC (Brain and Spinal Injury Center) and UCSF [Erol et al., 2013b, Sivaganesan et al., 2012, Erol et al., 2015].

2 Background and Related Material

A state space model (SSM) consists of the parameters Θ, latent states $\{X_t\}$ and the observations $\{Y_t\}$. Let Θ be a parameter space for a partially observable Markov process $\{X_t\}_{t\geq 0}, \{Y_t\}_{t\geq 0}$ as shown in Figure 1 and defined as follows:

$$X_0 \sim p(x_0 \,|\, \theta) \tag{1}$$

$$X_t \,|\, x_{t-1} \sim p(x_t \,|\, x_{t-1}, \theta) \tag{2}$$

$$Y_t \,|\, x_t \sim p(y_t \,|\, x_t, \theta) \tag{3}$$

Here the state variables X_t are unobserved and the observations Y_t are assumed conditionally independent of other observations given X_t. We assume in this section that states X_t, observations Y_t, and parameters θ are real-valued vectors in d, m, and p dimensions respectively. Here both the transition and sensor models are parameterized by θ.

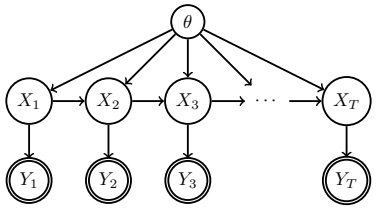


Figure 1: DBN depiction of a state-space model with static parameters θ. $X_{1:T}$ are latent states and $Y_{1:T}$ are observations.

The filtering density $p(x_t \,|\, y_{0:t}, \theta)$ obeys the following recursion:

$$\begin{aligned} p(x_t \,|\, y_{0:t}, \theta) &= \frac{p(y_t \,|\, x_t, \theta) p(x_t \,|\, y_{0:t-1}, \theta)}{p(y_t \,|\, y_{0:t-1}, \theta)} \\ &= \frac{p(y_t \,|\, x_t, \theta)}{p(y_t \,|\, y_{0:t-1}, \theta)} \int p(x_{t-1} \,|\, y_{0:t-1}, \theta) p(x_t \,|\, x_{t-1}, \theta) dx_{t-1} \end{aligned} \tag{4}$$

where the update steps for $p(x_t \,|\, y_{0:t-1}, \theta)$ and $p(y_t \,|\, y_{0:t-1}, \theta)$ involve the evaluation of integrals that are not in general analytically tractable.

$$p(x_t \,|\, y_{0:t-1}, \theta) = \int p(x_t \,|\, x_{t-1}, \theta) p(x_{t-1} \,|\, y_{0:t-1}, \theta) dx_{t-1} \tag{5}$$

$$p(y_t | y_{0:t-1}) = \int p(x_{t-1} | y_{0:t-1}, \theta) p(x_t | x_{t-1}, \theta) p(y_t | x_t) dx_{t-1:t} \tag{6}$$

2.1 Particle Filtering

Sequential Monte Carlo (SMC) is a widely adopted class of methods for inference on SSMs. Given the observed values $Y_{0:T} = y_{0:T}$, the posterior distribution $p(x_{0:t}, \theta | y_{0:t})$ is approximated by a set of K particles, with each particle k denoted by X_t^k for $1 \leq k \leq K$. X_t^k consists of a particular assignment of the state

Algorithm 1: Sequential importance sampling-resampling (SIR)

Input: N: number of particles;
$y_0, \ldots, y_T$: observation sequence
Output: $\bar{x}_{1:T}^{1:N}$
initialize $\{x_0^i\}$;
for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, N$ **do**
    sample $x_t^i \sim p(x_t \,|\, x_{t-1}^i)$;
    $w_t^i \leftarrow p(y_t \,|\, x_t^i)$;
  sample $\{\frac{1}{N}, \bar{x}_t^i\} \leftarrow$Multinomial$\{w_t^i, x_t^i\}$;
  $\{x_t^i\} \leftarrow \{\bar{x}_t^i\}$;

variables and the parameter denoted by $x_{0:t}^k$ and θ^k. Particles are propagated forward through the proposal density $q(x_t^k | x_{t-1}^k, \theta^k)$ and resampled at each time step t according to the weights w_t^k. The canonical example is the sequential importance sampling-resampling algorithm (SIR) (Algorithm 1) which uses the transition model as the proposal density.

The SIR filter has various appealing properties. It is modular, efficient, and easy to implement. The filter takes constant time per update, regardless of time T, and as the number of particles $N \to \infty$, the empirical filtering density converges to the true marginal posterior density under suitable assumptions [Crisan and Doucet, 2002].

Particle filters can accommodate unknown parameters by adding parameter variables into the state vector with an "identity function" transition model. As noted in Section 1 this approach leads to degeneracy problems—especially for high-dimensional parameter spaces. Due to the resampling step, which "kills" the particles with low weights, the Θ-diversity of the particles reduces at every time step (parameter particles are only sampled once at the beginning of the inference procedure). Hence, K needs to be sufficiently large to prevent an early convergence to a degenerate particle set. To ensure that *some* particle has initial parameter values with bounded error, the number of particles must grow exponentially with the dimension of the parameter space.

There have been many solutions proposed throughout the years to cope with the degeneracy problem via rejuvenating the particle set. In this section we will focus on the Liu-West filter [Liu and West, 2001], Resample-Move algorithm [Gilks and Berzuini, 2001], the Storvik filter [Storvik, 2002], and particle Markov chain Monte Carlo (PMCMC) [Andrieu et al., 2010].

2.2 Liu-West Filter

Liu and West [2001] introduce a stochastic transition model for the parameter variables, allowing exploration of the parameter space. This so-called *artifical dynamics* is designed to preserve the first and second order moments of the

particle distribution. The parameter particles are jittered at each time step as follows:

$$\theta_t^i = \rho\theta_{t-1}^i + (1-\rho)\bar{\theta}_{t-1} + \sqrt{1-\rho^2}\sigma(\theta_{t-1})\mathcal{N}(0, I) \tag{7}$$

where $\bar{\theta}_t$ and $\sigma(\theta_t)$ represent mean and standard deviations of parameter particles respectively. Liu-West filter is depicted in Algorithm 2.

Algorithm 2: Liu-West Filter (LW)

Input: N: number of particles;
$y_0, \ldots, y_T$: observation sequence
Output: $\bar{x}_{1:T}^{1:N}$
initialize $\left\{x_0^i\right\}$;
for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, N$ **do**
    $\theta_t^i = \rho\theta_{t-1}^i + (1-\rho)\bar{\theta}_{t-1} + \sqrt{1-\rho^2}\sigma(\theta_{t-1})\mathcal{N}(0, I)$;
    sample $x_t^i \sim p(x_t \,|\, x_{t-1}^i, \theta^i)$;
    $w_t^i \leftarrow p(y_t \,|\, x_t^i, \theta^i)$;
  sample $\left\{\frac{1}{N}, \bar{x}_t^i\right\} \leftarrow$Multinomial$\left\{w_t^i, x_t^i\right\}$;
  $\left\{x_t^i\right\} \leftarrow \left\{\bar{x}_t^i\right\}$;

The Liu-West filter is simple and fast, however, the amount of bias introduced by the artificial dynamics is hard to quantify and for models with high dimensional parameter spaces or complex nonlinear dynamics with multimodal posterior densities, it fails. Furthermore, the kernel parameter ρ needs to be manually tuned and the filter is not applicable to models with discrete parameter spaces.

2.3 Resample-Move Algorithm

In order to avoid artificial dynamics or other approximations, an unbiased method that utilizes MCMC steps to rejuvenate particles is introduced by Gilks and Berzuini [2001]. To introduce diversity to the population of particles, an MCMC kernel with invariant density $p(x_{0:t}, \theta \mid y_{0:t})$ is utilized.

$$\left(X_{0:t}^i, \theta_t^i\right) \sim K_t\left(\cdot, \cdot \mid \bar{X}_{0:t}^i, \bar{\theta}_t^i\right) \tag{8}$$

where by construction the kernel K_t satisfies

$$p(x'_{0:t}, \theta' \mid y_{0:t}) = \int p(x_{0:t}, \theta \mid y_{0:t}) K_t\left(x'_{0:t}, \theta' \mid x_{0:t}^i, \theta_t^i\right) d(x_{0:t}, \theta).$$

Utilizing such an MCMC move requires $O(t)$ computation per time step, leading Gilks and Berzuini to propose a move at a rate proportional to $1/t$ so as to have asymptotically constant-time updates. Resample-Move algorithm is unbiased and accurate, but the design and execution of MCMC kernels pose a serious problem. The proposal densities to be utilized by MCMC need to be hand tuned

for reasonable acceptance ratios and it is usually hard to gauge the convergence of the MCMC samples. Although asymptotically constant-time per update via $1/t$ rate moves, making an MCMC move when t is large will be prohibitive as the MCMC kernel will need to operate on thousands of time steps of state variables.

2.4 Storvik Filter

Similar to Resample-Move algorithm, Storvik [2002], Polson et al. [2008], and Carvalho et al. [2010] propose a Gibbs move to rejuvenate only the parameter particles via

$$K_t\left(x'_{0:t}, \theta' \mid x^i_{0:t}, \theta^i_t\right) = \delta(x'_{0:t}) p(\theta' \mid x_{0:t}, y_{0:t}). \tag{9}$$

Sampling from $p(\theta \mid x_{0:t}, y_{0:t})$ is $O(t)$ per time step and prohibitive for online applications. Storvik [2002] focuses on models that accept fixed-dimensional sufficient statistics such that $p(\theta \mid x_{0:t}, y_{0:t}) = p(\theta \mid s_t(x_{0:t}, y_{0:t}))$. The important property of this algorithm is that the parameter value simulated at time t does not depend on the values simulated previously. This property prevents the impoverishment of the parameter values in particles. Storvik filter is depicted in Algorithm 3.

Algorithm 3: Storvik's filter.

Input: N: number of particles;
$y_0, \ldots, y_T$: observation sequence
Output: $\bar{x}^{1:N}_{1:T}$, $\theta^{1:N}$
initialize $\left\{x^i_0\right\}$;
for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, N$ **do**
    sample $\theta^i \sim p(\theta | x^i_{0:t-1}, y^i_{0:t-1})$;
    sample $x^i_t \sim p(x_t | x^i_{t-1}, \theta^i)$;
    $w^i \leftarrow p(y_t | x^i_t, \theta^i)$;
  sample $\left\{\frac{1}{N}, \bar{x}^i_t\right\} \leftarrow$Multinomial$\left\{w^i_t, x^i_t\right\}$;
  $\left\{x^i_t\right\} \leftarrow \left\{\bar{x}^i_t\right\}$;

Storvik [2002] shows how to obtain a sufficient statistic in the context of what he calls the *Gaussian system process*, a transition model satisfying the equation

$$x_t = \mathbf{F}_t^T \theta + \epsilon_t, \;\; \epsilon_t \sim N(0, \mathbf{Q}) \tag{10}$$

where θ is the vector of unknown parameters with a prior of $N(\theta_0, \mathbf{C}_0)$ and $\mathbf{F}_t = \mathbf{F}(x_{t-1})$ is a matrix where elements are possibly nonlinear functions of x_{t-1}. An arbitrary but known observation model is assumed. Then the standard theory states that $\theta \mid x_{0:t} \sim N(m_t, \mathbf{C}_t)$. Thus, m_t and $\mathbf{C}_t$ constitute a fixed-dimensional sufficient statistic for θ.

2.5 Particle MCMC

For joint parameter and state estimation, the "gold standard" approaches are particle Markov chain Monte Carlo (PMCMC) algorithms [Andrieu et al., 2010], such as particle independent Metropolis-Hastings (PIMH), particle marginal Metropolis-Hastings (PMMH), particle Gibbs (PGibbs) [Andrieu et al., 2010] and particle Gibbs with ancestor resampling (PGAS) [Lindsten et al., 2014]. PMCMC algorithms utilize an MCMC transition kernel over the parameter space and a classical particle filter for state estimation and likelihood computation. PMCMC methods are favored due to their theoretical guarantees as an unbiased estimator as well as their "black-box" property: the only requirement for PMCMC methods is that one needs to sample from the SSM and compute likelihood for the evidence, which is in most cases straightforward.

The usual approach to approximate the target density $p(x_{0:T}, |y_{0:T})$ is MCMC. However, it is difficult to design efficient MCMC sampling algorithms for arbitrary state-space models. Particle MCMC (PMCMC) is an MCMC technique which relies on sequential Monte Carlo to build efficient high dimensional proposal distributions. The elegance of PMCMC arises from the fact that it can use the unbiased SMC estimate for the evidence likelihood $\hat{p}_\theta(y_{0:T})$ to build an accurate MCMC algorithm. Particle Marginal Metropolis-Hastings (PMMH) algorithm is depicted in Algorithm 4.

Algorithm 4: Particle Marginal Metropolis-Hastings (PMMH)

Input: K: number of particles;
N: number of MCMC steps;
$y_0, \ldots, y_T$: observation sequence
Set $\theta(0)$ randomly ;
Run an SMC algorithm targeting $p_{\theta(0)}(x_{0:T} \mid y_{0:T})$;
for $n = 1, \ldots, N$ **do**
  Sample a proposal $\theta' \sim q(\theta \mid \theta(n-1))$;
  Run an SMC algorithm targeting $p_{\theta'}(x_{0:T} \mid y_{0:T})$;
  Set $\theta(n) = \theta'$ with probability
  $\min\left\{1, \frac{\bar{p}_{\theta'}(y_{0:T})p(\theta')q(\theta(n-1)|\theta')}{\bar{p}_{\theta(n-1)}(y_{0:T})p(\theta(n-1))q(\theta'|\theta(n-1))}\right\}$
  otherwise reject ;

One significant drawback of PMCMC algorithms is the computational budget. Suppose there are T time steps and we perform N MCMC steps with K particles. Then the time complexity for PMCMC algorithms is $O(NKT)$. Note that for adequate mixing, it is necessary for N to be sufficiently large. For a real-world application with a large number of time steps and complex dynamics, the mixing problem becomes critical. Moreover, since PMCMC algorithms require multiple sweeps over observations, T must be fixed in advance and the full history of the particles must be stored. This "offline" property of PMCMC algorithms is infeasible for online/streaming applications, such as real-time object

tracking and signal monitoring, for which constant time per update is required and storing the whole history is prohibitive.

2.6 Concluding Remarks

Sequential Monte-Carlo (particle filter) based algorithms have been introduced for real-world applications [Gordon et al., 1993, Arulampalam et al., 2002, Cappé et al., 2007]. However, classical particle filter algorithms suffer from the path degeneracy problem, especially for parameters, and leave it a challenge to jointly estimate parameters and states for SSMs with complex dependencies and nonlinear dynamics. Real-world models can involve both discrete and continuous variables, arbitrary dependencies and a rich collection of nonlinearities and distributions. Existing algorithms are either restricted to a particular class of models, such as the Storvik filter [Storvik, 2002], or biased as in the Liu-West filter [Liu and West, 2001], or very expensive in time complexity, such as particle MCMC [Andrieu et al., 2010], which utilizes an expensive MCMC kernel over the parameter space and typically requires a very large number of MCMC iterations to converge.

In the upcoming sections we will explore two algorithms, namely, the extended parameter filter and the assumed parameter filter that try to close the gap between computational efficiency, accuracy, and applicability to arbitrary model classes.

3 The Extended Parameter Filter

Storvik [2002] and Polson et al. [2008] observe that a fixed-dimensional sufficient statistic (if one exists) for θ can be updated in constant time. Storvik describes an algorithm for a specific family of linear-in-parameters transition models.

We show that Storvik's algorithm is a special case of the Kalman filter in parameter space and identify a more general class of *separable* systems to which the same approach can be applied. By analogy with the extended Kalman filter, we propose a new algorithm, the *extended parameter filter* (EPF) [Erol et al., 2013a], that computes a separable approximation to the parameter posterior and allows a fixed-dimensional (approximate) sufficient statistic to be maintained. The method is quite general: for example, with a polynomial approximation scheme such as Taylor expansion any analytic posterior can be handled.

Section 3.1 briefly revisits Storvik filter and in the next section we introduce our notion of separable models. Section 3.3 describes the EPF algorithm, and Section 3.4 discusses the details of a polynomial approximation scheme for arbitrary densities, which Section 3.5 then applies to estimate posterior distributions of static parameters. Section 3.6 provides empirical results comparing the EPF to other algorithms.

3.1 Revisiting Storvik Filter

As discussed in section 2, Storvik [2002] shows how to obtain a sufficient statistic in the context of what he calls the *Gaussian system process*, a transition model satisfying the equation

$$x_t = \mathbf{F}_t^T \theta + \epsilon_t, \;\; \epsilon_t \sim N(0, \mathbf{Q}) \tag{11}$$

where θ is the vector of unknown parameters with a prior of $N(\theta_0, \mathbf{C}_0)$ and $\mathbf{F}_t = \mathbf{F}(x_{t-1})$ is a matrix where elements are possibly nonlinear functions of x_{t-1}. An arbitrary but known observation model is assumed. Then the standard theory states that $\theta \,|\, x_{0:t} \sim N(m_t, \mathbf{C}_t)$ where the recursions for the mean and the covariance matrix are as follows:

$$\begin{aligned} \mathbf{D}_t &= \mathbf{F}_t^T \mathbf{C}_{t-1} \mathbf{F}_t + \mathbf{Q} \\ \mathbf{C}_t &= \mathbf{C}_{t-1} - \mathbf{C}_{t-1} \mathbf{F}_t \mathbf{D}_t^{-1} \mathbf{F}_t^T \mathbf{C}_{t-1} \\ m_t &= m_{t-1} + \mathbf{C}_{t-1} \mathbf{F}_t \mathbf{D}_t^{-1} (x_t - \mathbf{F}_t^T m_{t-1}) \end{aligned} \tag{12}$$

Thus, m_t and $\mathbf{C}_t$ constitute a fixed-dimensional sufficient statistic for θ.

These updates are in fact a special case of Kalman filtering applied to the parameter space. Matching terms with the standard KF update equations [Kalman, 1960], we find that the transition matrix for the KF is the identity matrix, the transition noise covariance matrix is the zero matrix, the observation matrix for the KF is $\mathbf{F}_t$, and the observation noise covariance matrix is $\mathbf{Q}$. This correspondence is of course what one would expect, since the true parameter values are fixed (i.e., an identity transition). The derivation is given in Section 3.8.

3.2 Separability

In this section, we define a condition under which there exist efficient updates to parameters. Again, we focus on the state-space model as described in Figure 1 and Equation (3). The model in Equation (3) can also be expressed as

$$\begin{aligned} x_t &= f_\theta(x_{t-1}) + v_t \\ y_t &= g(x_t) + w_t \end{aligned} \tag{13}$$

for some suitable f_θ, g, v_t, and w_t.

Definition 1. *A system is* separable *if the transition function $f_\theta(x_{t-1})$ can be written as $f_\theta(x_{t-1}) = l(x_{t-1})^T h(\theta)$ for some $l(\cdot)$ and $h(\cdot)$ and if the stochastic i.i.d. noise v_t has log-polynomial density.*

Theorem 1. *For a separable system, there exist fixed-dimensional sufficient statistics for the Gibbs density, $p(\theta \mid x_{0:T})$.*

The proof is straightforward by the Fisher–Neyman factorization theorem; more details are given in Section 3.8.

The Gaussian system process models defined in Equation (11) are separable, since the transition function $\mathbf{F}_t^T\theta = (F_t)^T\theta$, but the property—and therefore Storvik's algorithm—applies to a much broader class of systems. Moreover, as we now show, non-separable systems may in some cases be well-approximated by separable systems, constructed by polynomial density approximation steps applied to either the Gibbs distribution $p(\theta \mid x_{0:t})$ or to the transition model.

3.3 The extended parameter filter

Let us consider the following model.

$$x_t = f_\theta(x_{t-1}) + v_t;\ v_t \sim N(0, \Sigma) \tag{14}$$

where $x \in \mathbb{R}^d$,$\theta \in \mathbb{R}^p$ and $f_\theta(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ is a vector-valued function parameterized by θ. We assume that the transition function f_θ may be non-separable. Our algorithm will create a polynomial approximation to either the transition function or to the Gibbs distribution, $p(\theta \mid x_{0:t})$.

To illustrate, let us consider the transition model $f_\theta(x_{t-1}) = \sin(\theta x_{t-1})$. It is apparent that this transition model is non-separable. If we approximate the transition function with a Taylor series in θ centered around zero

$$f_\theta(x_{t-1}) \approx \hat{f}_\theta(x_{t-1}) = x_{t-1}\theta - \frac{1}{3!}x_{t-1}^3\theta^3 + \dots \tag{15}$$

and use $\hat{f}$ as an approximate transition model, the system will become separable. Then, Storvik's filter can be applied in constant time per update. This Taylor approximation leads to a log-polynomial density of the form of Equation (25).

Our approach is analogous to that of the extended Kalman filter (EKF). EKF linearizes nonlinear transitions around the current estimates of the mean

and covariance and uses Kalman filter updates for state estimation [Welch and Bishop, 1995]. Our proposed algorithm, which we call the extended parameter filter (EPF), approximates a non-separable system with a separable one, using a polynomial approximation of some arbitrary order. This separable, approximate model is well-suited for Storvik's filter and allows for constant time updates to the Gibbs density of the parameters.

Although we have described an analogy to the EKF, it is important to note that the EPF can effectively use higher-order approximations instead of just first-order linearizations as in EKF. In EKF, higher order approximations lead to intractable integrals. The prediction integral for EKF

$$p(x_t \mid y_{0:t-1}) = \int p(x_{t-1} \mid y_{0:t-1}) p(x_t \mid x_{t-1}) dx_{t-1}$$

can be calculated for linear Gaussian transitions, in which case the mean and the covariance matrix are the tracked sufficient statistic. However, in the case of quadratic transitions (or any higher-order transitions), the above integral is no longer analytically tractable.

In the case of EPF, the transition model is the identity transition and hence the prediction step is trivial. The filtering recursion is

$$p(\theta \mid x_{0:t}) \propto p(x_t \mid x_{t-1}, \theta) p(\theta \mid x_{0:t-1}). \tag{16}$$

We approximate the transition $p(x_t \mid x_{t-1}, \theta)$ with a log-polynomial density $\hat{p}$ (log-polynomial in θ), so that the Gibbs density, which satisfies the recursions in equation 16, has a fixed log-polynomial structure at each time step. Due to the polynomial structure, the approximate Gibbs density can be tracked in terms of its sufficient statistic (i.e., in terms of the coefficients of the polynomial). The log-polynomial structure is derived in Section 3.5. Pseudo-code for EPF is shown in Algorithm 3.

Note that the approximated Gibbs density will be a log-multivariate polynomial density of fixed order (proportional to the order of the polynomial approximation). Sampling from such a density is not straightforward but can be done by Monte Carlo sampling. We suggest slice sampling [Neal, 2003] or the Metropolis-Hastings algorithm [Robert and Casella, 2005] for this purpose. Although some approximate sampling scheme is necessary, sampling from the approximated density remains a constant-time operation when the dimension of $\hat{p}$ remains constant.

It is also important to note that performing a polynomial approximation for a p-dimensional parameter space may not be an easy task. However, we can reduce the computational complexity of such approximations by exploiting locality properties. For instance, if $f_\theta(\cdot) = h_{\theta_1,\ldots,\theta_{p-1}}(\cdot) + g_{\theta_p}(\cdot)$, where h is separable and g is non-separable, we only need to approximate g.

In section 3.4, we discuss the validity of the approximation in terms of the KL-divergence between the true and approximate densities. In section 3.4.1, we analyze the distance between an arbitrary density and its approximate form with respect to the order of the polynomial. We show that the distance goes to

Algorithm 5: Extended Parameter Filter

Result: Approximate the Gibbs density $p(\theta \mid x_{0:t}, y_{0:t})$ with the log-polynomial density $\hat{p}(\theta \mid x_{0:t}, y_{0:t})$

Output: $\tilde{x}^1 \dots \tilde{x}^N$

initialize $\{x_0^i\}$ and $S_0^i \leftarrow 0$;

for $t = 1, \dots, T$ **do**

  for $i = 1, \dots, N$ **do**

    $S_t^i = \text{update}(S_{t-1}^i, x_{t-1})$; `// update statistics for polynomial approximation` $\log(\hat{p}(\theta|\bar{x}_{0:t-1}, y_{0:t-1}))$

    sample $\theta^i \sim \hat{p}(\theta \mid \bar{x}_{0:t-1}^i, y_{0:t-1}) = \hat{p}(\theta \mid S_t^i)$;

    sample $x_t^i \sim p(x_t \mid \bar{x}_{t-1}^i, \theta^i)$;

    $w^i \leftarrow p(y_t \mid x_t^i, \theta^i)$;

  sample $\{\frac{1}{N}, \bar{x}_t^i, \bar{S}_t^i\} \leftarrow$Multinomial$\{w_t^i, x_t^i, S_t^i\}$;

  $\{x_t^i, S_t^i\} \leftarrow \{\bar{x}_t^i, \bar{S}_t^i\}$;

zero *super-exponentially*. Section 3.5 analyzes the error for the static parameter estimation problem and introduces the form of the log-polynomial approximation.

3.4 Approximating the conditional distribution of parameters

In this section, we construct approximate sufficient statistics for arbitrary one–dimensional state space models. We do so by exploiting log-polynomial approximations to arbitrary probability densities. We prove that such approximations can be made arbitrarily accurate. Then, we analyze the error introduced by log-polynomial approximation for the arbitrary one–dimensional model.

3.4.1 Taylor approximation to an arbitrary density

Let us assume a distribution p (known only up to a normalization constant) expressed in the form $p(x) \propto \exp(S(x))$, where $S(x)$ is an analytic function on the support of the distribution. In general we need a Monte Carlo method to sample from this arbitrary density. In this section, we describe an alternative, simpler sampling method. We propose that with a polynomial approximation $P(x)$ (Taylor, Chebyshev etc.) of sufficient order to the function $S(x)$, we may sample from a distribution $\widehat{p} \propto \exp(P(x))$ with a simpler (i.e. log-polynomial) structure. We show that the distance between the distributions p and $\widehat{p}$ reduces to 0 as the order of the approximation increases.

The following theorem is based on Taylor approximations; however, the theorem can be generalized to handle any polynomial approximation scheme. The proof is given in Section 3.8.

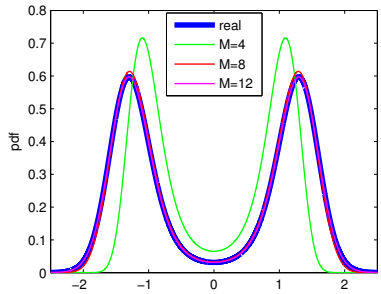


Figure 2: Approximated PDFs to the order M.

Theorem 2. *Let $S(x)$ be a $M+1$ times differentiable function with bounded derivatives, and let $P(x)$ be its M-th order Taylor approximation. Then the KL-divergence between distributions p and $\hat{p}$ converges to 0,* ***super-exponentially*** *as the order of approximation $M \to \infty$.*

We validate the Taylor approximation approach for the log-density $S(x) = -x^2 + 5\sin^2(x)$. Figure 2 shows the result for this case.

3.5 Online approximation of the Gibbs density of the parameter

In our analysis, we will assume the following model.

$$x_t = f_\theta(x_{t-1}) + v_t,\ v_t \sim N(0, \sigma^2)$$
$$y_t = g(x_t) + w_t,\ w_t \sim N(0, \sigma_o^2)$$

The posterior distribution for the static parameter is

$$p(\theta|x_{0:T}) \propto p(\theta) \prod_{t=1}^{T} p(x_t|x_{t-1}, \theta).$$

The product term, which requires linear time, is the bottleneck for this computation. A polynomial approximation to the transition function $f_\theta(\cdot)$ (the Taylor approximation around $\theta = 0$) is:

$$f_\theta(x_{t-1}) = h(x_{t-1}, \theta) = \sum_{i=0}^{M} \underbrace{\frac{1}{i!} \frac{d^i h(x_{t-1}, \theta^i)}{d\theta}\Big|_{\theta=0}}_{H^i(x_{t-1})} \theta^i + R_M(\theta)$$
$$= \sum_{i=0}^{M} H^i(x_{t-1})\theta^i + R_M(\theta) = \hat{f}(\theta) + R_M(\theta)$$

where R_M is the error for the M-dimensional Taylor approximation. We define coefficients $J^i_{x_{t-1}}$ to satisfy $\left(\sum_{i=0}^{M} H^i(x_{t-1})\theta^i\right)^2 = J^{2M}_{x_{t-1}}\theta^{2M} + \cdots + J^0_{x_{t-1}}\theta^0$.

Let $\hat{p}(\theta \mid x_{0:T})$ denote the approximation to $p(\theta \mid x_{0:T})$ obtained by using the polynomial approximation to f_θ introduced above.

Theorem 3. *$\hat{p}(\theta \mid x_{0:T})$ is in the exponential family with the log-polynomial density*

$$\log p(\theta) + \underbrace{\begin{pmatrix} \theta^1 \\ \vdots \\ \theta^M \\ \theta^{M+1} \\ \vdots \\ \theta^{2M} \end{pmatrix}^T}_{T(\theta)^T} \cdot \underbrace{\begin{pmatrix} \frac{1}{\sigma^2}\sum_{k=1}^T x_k H^1(x_{k-1}) - \frac{1}{2\sigma^2}\sum_{k=1}^T J^1_{x_{k-1}} \\ \vdots \\ \frac{1}{\sigma^2}\sum_{k=1}^T x_k H^M(x_{k-1}) - \frac{1}{2\sigma^2}\sum_{k=1}^T J^M_{x_{k-1}} \\ -\frac{1}{2\sigma^2}\sum_{k=1}^T J^{M+1}_{x_{k-1}} \\ \vdots \\ -\frac{1}{2\sigma^2}\sum_{k=1}^T J^{2M}_{x_{k-1}} \end{pmatrix}}_{\eta(x_0,\ldots,x_t)} \tag{18}$$

The proof is given in the supplementary material.

This form has finite dimensional sufficient statistics. Standard sampling from $p(\theta \mid x_{0:t})$ requires $O(t)$ time, whereas with the polynomial approximation we can sample from this structured density of fixed dimension in constant time (given that sufficient statistics were tracked). We can furthermore prove that sampling from this exponential form approximation is asymptotically correct.

Theorem 4. *Let $p_T(\theta \mid x_{0:T})$ denote the Gibbs distribution and $\widehat{p}_T(\theta \mid x_{0:T})$ its order M exponential family approximation. Assume that parameter θ has support $\mathcal{S}_\theta$ and finite variance. Then as $M \to \infty, T \to \infty$, the KL divergence between p_T and $\widehat{p}_T$ goes to zero.*

$$\lim_{M,T\to\infty} D_{KL}(p_T \mid\mid \hat{p}_T) = 0$$

The proof is given in Section 3.8. Note that the analysis above can be generalized to higher dimensional parameters. The one dimensional case is discussed for ease of exposition.

In the general case, an order M Taylor expansion for a p dimensional parameter vector θ will have M^p terms. Then each update of the sufficient statistics will cost $O(M^p)$ per particle, per time step, yielding the total complexity $O(NTM^p)$. However, as noted before, we can often exploit the local structure of f_θ to speed up the update step. Notice that in either case, the update cost per time step is fixed (independent of T).

3.6 Experiments

The algorithm is implemented for three specific cases. Note that the models discussed do not satisfy the Gaussian process model assumption of Storvik [2002].

3.6.1 Single parameter nonlinear model

Consider the following model with sinusoid transition dynamics (SIN):

$$\begin{aligned} x_t &= \sin(\theta x_{t-1}) + v_t,\ v_t \sim N(0, \sigma^2) \\ y_t &= x_t + w_t,\ w_t \sim N(0, \sigma^2_{\text{obs}}) \end{aligned} \tag{19}$$

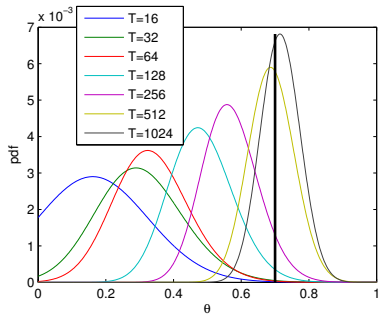
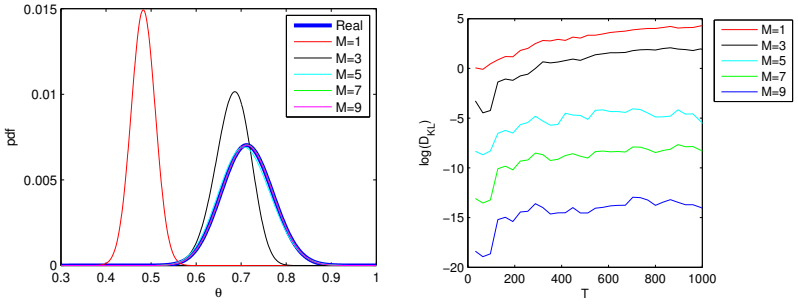


Figure 3: Sinusoidal dynamical model (SIN). Shrinkage of the Gibbs density $p(\theta \mid x_{0:T})$ with respect to time duration T. Note that as T grows, the Gibbs density converges to the true parameter value.



(a) Approximating Gibbs (b) KL-divergence

Figure 4: Sinusoidal dynamical model (SIN). (a) Convergence of the approximate densities to the Gibbs density $p(\theta \mid x_{0:1024})$ with respect to the approximation order M; (b) KL-divergence $D_{KL}(p \mid \hat{p})$ with respect to duration T and approximation order M.

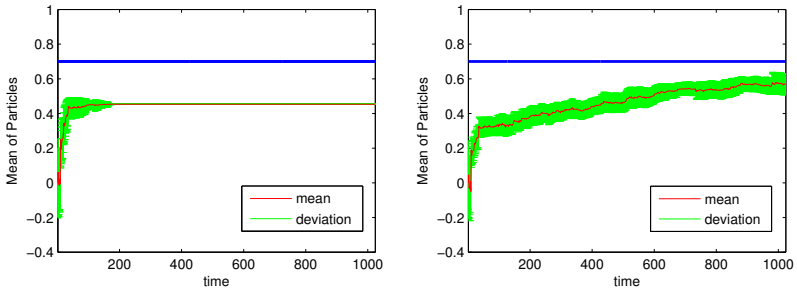
where $\sigma = 1$, $\sigma_{\text{obs}} = 0.1$ and the Gaussian prior for parameter θ is $N(0, 0.2^2)$. The observation sequence is generated by sampling from SIN with true parameter value $\theta = 0.7$.

Figure 3 shows how the Gibbs density $p(\theta \mid x_{0:t})$ shrinks with respect to time, hence verifying identifiability for this model. Notice that as T grows, the densities concentrate around the true parameter value.

A Taylor approximation around $\theta = 0$ has been applied to the transition function $\sin(\theta x_t)$. Figure 4(a) shows the approximate densities for different polynomial orders for $T = 1024$. Notice that as the polynomial order increases, the approximate densities converge to the true density $p(\theta \mid x_{0:1024})$.

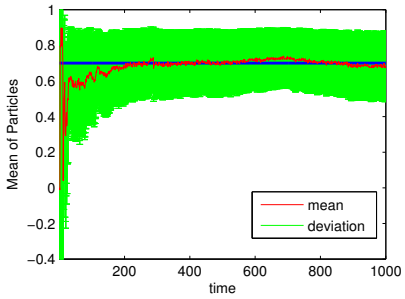
The KL-divergence $D_{KL}(p \mid\mid \hat{p})$ for different polynomial orders (N) and different data lengths (T) is illustrated in Figure 4(b). The results are consistent with the theory developed in Section 3.4.1.

The degeneracy of a bootstrap filter with $N = 50000$ particles can be seen from figure 5(a). The Liu–West approach with $N = 50000$ particles is shown in



(a) Particle filter (SIR)

(b) Liu–West filter



(c) EPF

Figure 5: Sinusoidal dynamical model (SIN). (a): Particle filter (SIR) with $N = 50000$ particles. Note the failure to converge to the true value of parameter θ (0.7, shown as the blue line). (b): Liu–West filter with $N = 50000$ particles. (c): EPF with $N = 1000$ particles and 7-th order approximation. Note both SIR and Liu–West do not converge, while the EPF converges quickly even with orders of magnitude fewer particles.

5(b). The perturbation is $\theta_t = \rho\theta_{t-1} + (1-\rho)\bar{\theta}_{t-1} + \sqrt{1-\rho^2}\,\mathrm{std}(\theta_{t-1})N(0,1)$, where $\rho = 0.9$. Notice that even with $N = 50000$ particles and large perturbations, the Liu–West approach converges slowly compared to our method. Furthermore, for high-dimensional spaces, tuning the perturbation parameter ρ for Liu–West becomes difficult.

The EPF has been implemented on this model with $N = 1000$ particles with a 7-th order Taylor approximation to the posterior. The time complexity is $O(NT)$. The mean and the standard deviation of the particles are shown in figure 5(c).

3.6.2 Cauchy dynamical system

We consider the following model.

$$x_t = ax_{t-1} + \mathrm{Cauchy}(0, \gamma) \tag{20}$$

$$y_t = x_t + N(0, \sigma_{\mathrm{obs}}) \tag{21}$$

Here Cauchy is the Cauchy distribution centered at 0 and with shape parameter $\gamma = 1$. We use $a = 0.7$, $\sigma_{\text{obs}} = 10$, where the prior for the AR(1) parameter is $N(0, 0.2^2)$. This model represents autoregressive time evolution with heavy-tailed noise. Such heavy-tailed noises are observed in network traffic data and click-stream data. The standard Cauchy distribution we use is

$$f_v(v; 0, 1) = \frac{1}{\pi(1 + v^2)} = \exp\left(-\log(\pi) - \log(1 + v^2)\right).$$

We approximate $\log(1 + v^2)$ by $v^2 - v^4/2 + v^6/3 - v^8/4 + \ldots$ (the Taylor approximation at 0).

Figure 6(a) shows the simulated hidden state and the observations ($\sigma_{obs} = 10$). Notice that the simulated process differs substantially from a standard AR(1) process due to the heavy-tailed noise. Storvik's filter cannot handle this model since the necessary sufficient statistics do not exist.

Figure 6(b) displays the mean value estimated by a bootstrap filter with $N = 50000$ particles. As before the bootstrap filter is unable to perform meaningful inference. Figure 6(c) shows the performance of the Liu–West filter with both $N = 100$ and $N = 10000$ particles. The Liu–West filter does not converge for $N = 100$ particles and converges slowly for $N = 10000$ particles. Figure 6(d) demonstrates the rapid convergence of the EPF for only $N = 100$ particles with 10th order approximation. The time complexity is $O(NT)$.

Our empirical results confirm that the EPF proves useful for models with heavy-tailed stochastic perturbations.

3.6.3 Smooth Transition AR model

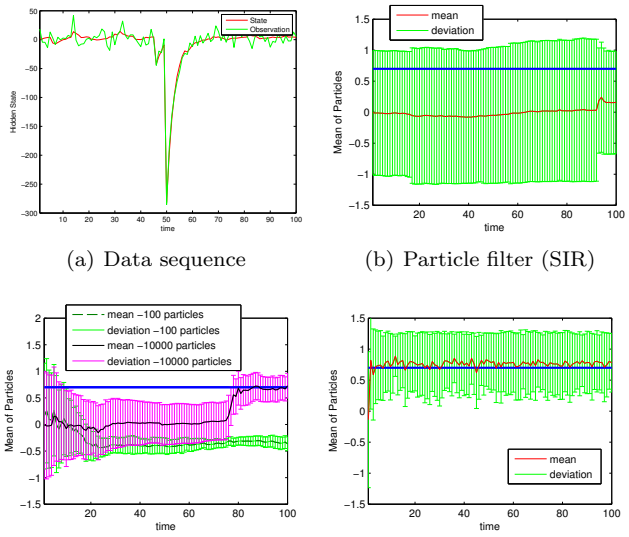
The smooth transition AR (STAR) model is a smooth generalization of the self-exciting threshold autoregressive (SETAR) model, [van Dijk et al., 2002]. It is generally expressed in the following form.

$$\begin{aligned} x_t = &\left(a_1 x_{t-1} + a_2 x_{t-2} + \cdots + a_p x_{t-p}\right)\left[1 - G(x_{t-d}; \gamma, c)\right] \\ &+ \left(b_1 x_{t-1} + b_2 x_{t-2} + \cdots + b_p x_{t-p}\right)\left[G(x_{t-d}; \gamma, c)\right] + \epsilon_t \end{aligned}$$

where ϵ_t is i.i.d. Gaussian with mean zero and variance σ^2 and $G(\cdot)$ is a nonlinear function of x_{t-d}, where $d > 0$. We will use the logistic function

$$G(y_{t-d}; \gamma, c) = \frac{1}{1 + \exp\left(-\gamma(x_{t-d} - c)\right)} \tag{22}$$

For high γ values, the logistic function converges to the indicator function, $\mathbb{I}(x_{t-d} > c)$, forcing STAR to converge to SETAR (SETAR corresponds to a switching linear–Gaussian system). We will use $p = 1 = d$, where $a_1 = 0.9$ and $b_1 = 0.1$ and $\sigma = 1$ (corresponding to two different AR(1) processes with high and low memory). We attempt to estimate parameters γ, c of the logistic function, which have true values $\gamma = 1$ and $c = 3$. Data (of length $T = 1000$) is generated from the model under fixed parameter values and with observation



(a) Data sequence (b) Particle filter (SIR)

(c) Liu–West filter (d) EPF

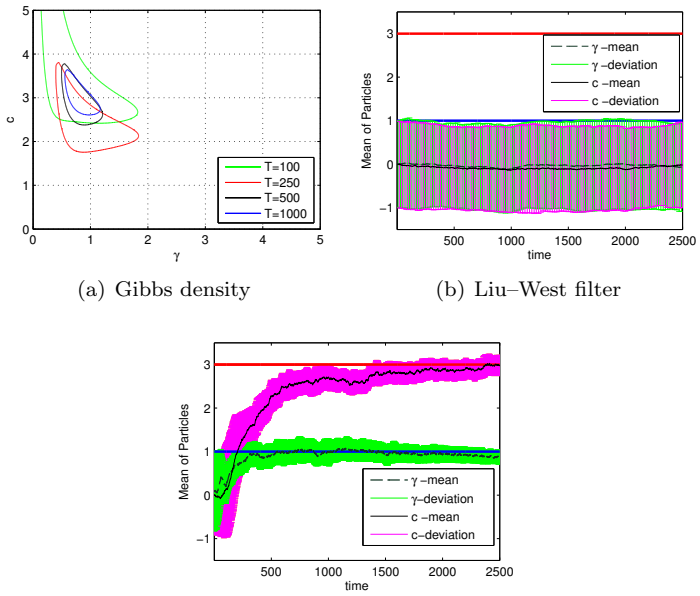
Figure 6: Cauchy dynamical system. (a): Example sequences for hidden states and observations. (b): Particle filter estimate with 50000 particles. (c): Liu–West filter with 100 and 10000 particles. (d): EPF using only 100 particles and 10th order approximation. Note EPF converges to the actual value of parameter a (=0.7, in blue line) while SIR does not even with orders of magnitude more particles, neither does Liu–West with the same number of particles.

model $y_t = x_t + w_t$, where w_t is additive Gaussian noise with mean zero and standard deviation $\sigma_{\text{obs}} = 0.1$. Figure 7(a) shows the shrinkage of the Gibbs density $p(\gamma, c \mid x_{0:T})$, verifying identifiability.

The non-separable logistic term is approximated as

$$\frac{1}{1+\exp\left(-\gamma(x_{t-1}-c)\right)} \approx \frac{1}{2} - \frac{1}{4}\gamma(c - x_{t-1}) + \frac{1}{48}\gamma^3(c - x_{t-1})^3 + \dots$$

Figure 7(b) displays the failure of the Liu–West filter for $N = 50000$ particles. Figure 7(c) shows the mean values for γ, c from EPF for only $N = 100$ particles with 9th order Taylor approximation. Sampling from the log-polynomial approximate density is done through the random-walk Metropolis–Hastings algorithm. For each particle path, at each time step t, the Metropolis–Hastings sampler is initialized from the parameter values at $t-1$. The burn-in period is set to be 0, so only one MH step is taken per time step (i.e., if a proposed sample is more likely it is accepted, else it is rejected with a specific probability). The whole filter has time complexity $O(NT)$.



(a) Gibbs density

(b) Liu–West filter

(c) EPF

Figure 7: STAR model. (a): Shrinkage of the Gibbs density $p(\gamma, c \mid x_{0:t})$ with respect to time. (b): Liu–West filter using 50000 particles. (c): EPF using 100 particles and 9th order approximation. Note the EPF's estimates for both parameters converge to the actual values quickly even with only 100 particles, while Liu–West does not converge at all.

3.7 Concluding Remarks

Learning the parameters of temporal probability models remains a significant open problem for practical applications. We have proposed the extended parameter filter (EPF), a novel approximate inference algorithm that combines Gibbs sampling of parameters with computation of approximate sufficient statistics. The update time for EPF is independent of the length of the observation sequence. Moreover, the algorithm has provable error bounds and handles a wide variety of models. Our experiments confirm these properties and illustrate difficult cases on which EPF works well.

One limitation of our algorithm is the complexity of Taylor approximation for high-dimensional parameter vectors. We noted that, in some cases, the process can be decomposed into lower-dimensional subproblems. Automating this step would be beneficial.

3.8 Supplementary Material

The Storvik filter as a Kalman filter in the parameter space
Let us consider the following model.

$$\begin{aligned} x_t &= \mathbf{A}x_{t-1} + v_t,\ v_t \sim N(0, \mathbf{Q}) \\ y_t &= \mathbf{H}x_t + w_t,\ w_t \sim N(0, \mathbf{R}) \end{aligned} \tag{23}$$

We will call the MMSE estimate Kalman filter returns as $x_{t|t} = \mathbb{E}[x_t \mid y_{0:t}]$ and the variance $\mathbf{P}_{t|t} = cov(x_t \mid y_{0:t})$. Then the update for the conditional mean estimate is as follows.

$$\begin{aligned} x_{t|t} &= \mathbf{A}x_{t-1|t-1} \\ &+ \underbrace{\mathbf{P}_{t|t-1}\mathbf{H}^T(\mathbf{H}\mathbf{P}_{t|t-1}\mathbf{H}^T + \mathbf{R})^{-1}}_{\mathbf{K}_t}(y_t - \mathbf{H}\mathbf{A}x_{t-1|t-1}) \end{aligned}$$

where as for the estimation covariance

$$\begin{aligned} \mathbf{P}_{t|t-1} &= \mathbf{A}\mathbf{P}_{t-1|t-1}\mathbf{A}^T + \mathbf{Q} \\ \mathbf{P}_{t|t} &= (\mathbf{I} - \mathbf{K}_t\mathbf{H})\mathbf{P}_{t|t-1} \end{aligned} \tag{24}$$

Matching the terms above to the updates in equation 12, one will obtain a linear model for which the transition matrix is $\mathbf{A} = \mathbf{I}$, the observation matrix is $\mathbf{H} = \mathbf{F}_t$, the state noise covariance matrix is $\mathbf{Q} = \mathbf{0}$, and the observation noise covariance matrix is $\mathbf{R} = \mathbf{Q}$.

Proof of Theorem 1

Theorem. *For a separable system, there exist fixed-dimensional sufficient statistics for the Gibbs density, $p(\theta \mid x_{0:T})$.*

Let us assume that $x \in \mathbb{R}^d$,$\theta \in \mathbb{R}^p$ and $f_\theta(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ is a vector valued function parameterized by θ. Moreover, due to the assumption of separability $f_\theta(x_{t-1}) = l(x_{t-1})^T h(\theta)$, where we assume that $l(\cdot) : \mathbb{R}^d \to \mathbb{R}^{m\times d}$ and $h(\cdot) : \mathbb{R}^p \to \mathbb{R}^m$ and m is an arbitrary constant. The stochastic perturbance will have the log-polynomial density $p(v_t) \propto \exp(\mathbf{\Lambda}_1 v_t + v_t^T \mathbf{\Lambda}_2 v_t + \dots)$ Let us analyze the case of $p(v_t) \propto \exp(\mathbf{\Lambda}_1 v_t + v_t^T \mathbf{\Lambda}_2 v_t)$, for mathematical simplicity.

Proof.

$$\log p(\theta \mid x_{0:T}) \propto \log p(\theta) + \sum_{t=1}^{T} \log p(x_t \mid x_{t-1}, \theta)$$

$$\propto \log p(\theta) + \sum_{t=1}^{T} \mathbf{\Lambda}_1 \left(x_t - l(x_{t-1})^T h(\theta)\right) +$$

$$\left(x_t - l(x_{t-1})^T h(\theta)\right)^T \mathbf{\Lambda}_2 \left(x_t - l(x_{t-1})^T h(\theta)\right)$$

$$\propto \log p(\theta) + \underbrace{\left(\sum_{t=1}^{T} -(\mathbf{\Lambda}_1 + 2x_t^T \mathbf{\Lambda}_2) l(x_{t-1})^T\right)}_{\mathbf{S}_1} h(\theta)$$

$$+ h^T(\theta) \underbrace{\left(\sum_{t=1}^{T} l(x_{t-1}) \mathbf{\Lambda}_2 l^T(x_{t-1})\right)}_{\mathbf{S}_2} h(\theta) + \text{constants}$$

Therefore, sufficient statistics ($\mathbf{S}_1 \in \mathbb{R}^{1\times m}$ and $\mathbf{S}_2 \in \mathbb{R}^{m\times m}$) exist. The analysis can be generalized for higher-order terms in v_t in similar fashion. □

Proof of Theorem 2

Theorem. *Let $S(x)$ be a $M+1$ times differentiable function with bounded derivatives, and let $P(x)$ be its M-th order Taylor approximation. Then the KL-divergence between distributions p and $\hat{p}$ converges to 0,* ***super-exponentially*** *as the order of approximation $M \to \infty$.*

Proposition 1. *Let $S(x)$ be a $M+1$ times differentiable function and $P(x)$ its order M Taylor approximation. Let $I = (x-a, x+a)$ be an open interval around x. Let $R(x)$ be the remainder function, so that $S(x) = P(x) + R(x)$. Suppose there exists constant U such that*

$$\forall y \in I, \quad \left|f^{(k+1)}(y)\right| \leq U$$

We may then bound

$$\forall y \in I, \quad |R(y)| \leq U \frac{a^{M+1}}{(M+1)!}$$

We define the following terms

$$\epsilon = U \frac{a^{M+1}}{(M+1)!}$$

$$Z = \int_I \exp(S(x))dx$$

$$\hat{Z} = \int_I \exp(P(x))dx$$

Since $\exp(\cdot)$ is monotone and increasing and $|S(x) - P(x)| \leq \epsilon$, we can derive tight bounds relating Z and $\widehat{Z}$.

$$\begin{aligned} Z &= \int_I \exp(S(x))dx \leq \int_I \exp(P(x) + \epsilon)dx \\ &= \hat{Z}\exp(\epsilon) \\ Z &= \int_I \exp(S(x))dx \geq \int_I \exp(P(x) - \epsilon)dx \\ &= \hat{Z}\exp(-\epsilon) \end{aligned}$$

Proof.

$$\begin{aligned} &D_{KL}(p||\hat{p}) = \int_I \ln\left(\frac{p(x)}{\hat{p}(x)}\right) p(x)dx \\ &= \int_I \Big(S(x) - P(x) + \ln(\hat{Z}) - \ln(Z)\Big) p(x)dx \\ &\leq \int_I |S(x) - P(x)|\, p(x)dx \\ &+ \int_I \Big|\ln(\hat{Z}) - \ln(Z)\Big|\, p(x)dx \\ &\leq 2\epsilon \propto \frac{a^{M+1}}{(M+1)!} \approx \frac{1}{\sqrt{2\pi(M+1)!}} \left(\frac{ae}{M+1}\right)^{M+1} \end{aligned}$$

where the last approximation follows from Stirling's approximation. Therefore, $D_{KL}(p||\hat{p}) \to 0$ as $M \to \infty$. □

Proof of Theorem 3

Theorem. *$\hat{p}(\theta \mid x_{0:T})$ is in the exponential family with the log-polynomial density*

$$\log p(\theta) + \underbrace{\begin{pmatrix} \theta^1 \\ \vdots \\ \theta^M \\ \theta^{M+1} \\ \vdots \\ \theta^{2M} \end{pmatrix}^T}_{T(\theta)^T} \cdot \underbrace{\begin{pmatrix} \frac{1}{\sigma^2}\sum_{k=1}^T x_k H^1(x_{k-1}) - \frac{1}{2\sigma^2}\sum_{k=1}^T J^1_{x_{k-1}} \\ \vdots \\ \frac{1}{\sigma^2}\sum_{k=1}^T x_k H^M(x_{k-1}) - \frac{1}{2\sigma^2}\sum_{k=1}^T J^M_{x_{k-1}} \\ -\frac{1}{2\sigma^2}\sum_{k=1}^T J^{M+1}_{x_{k-1}} \\ \vdots \\ -\frac{1}{2\sigma^2}\sum_{k=1}^T J^{2M}_{x_{k-1}} \end{pmatrix}}_{\eta(x_0,\ldots,x_t)} \tag{25}$$

Proof.

$$\log \hat{p}(\theta \mid x_{0:T}) = \log \left(p(\theta) \prod_{k=0}^{T} \hat{p}(x_k | x_{k-1}, \theta) \right)$$
$$= \log p(\theta) + \sum_{k=0}^{T} \log \hat{p}(x_k \mid x_{k-1}, \theta)$$

We can calculate the form of $\log \hat{p}(x_k \mid x_{k-1}, \theta)$ explicitly.

$$\log \hat{p}(x_k \mid x_{k-1}, \theta) = \log \mathcal{N}(\hat{f}(x_{k-1}, \theta), \sigma^2)$$
$$= -\log(\sigma\sqrt{2\pi}) - \frac{(x_k - \hat{f}(x_{k-1}, \theta))^2}{2\sigma^2}$$
$$= -\log(\sigma\sqrt{2\pi}) - \frac{x_k^2 - 2x_k\hat{f}(x_{k-1}, \theta) + \hat{f}(x_{k-1}, \theta)^2}{2\sigma^2}$$
$$= -\log(\sigma\sqrt{2\pi}) - \frac{x_k^2}{2\sigma^2} - \frac{\sum_{i=0}^{M} x_k H^i(x_{k-1})\theta^i}{\sigma^2}$$
$$+ \frac{\sum_{i=0}^{2M} J^i_{x_{k-1}} \theta^i}{2\sigma^2}$$

Using this expansion, we calculate

$$\log \hat{p}(\theta \mid x_{0:T}) = \log p(\theta) + \sum_{k=0}^{T} \log \hat{p}(x_k \mid x_{k-1}, \theta)$$
$$= \log p(\theta) - (T+1)\log(\sigma\sqrt{2\pi})$$
$$- \frac{1}{2\sigma^2}\left(\sum_{k=0}^{T} x_k^2\right) - T(\theta)^T \eta(x_0, \ldots, x_T)$$

where we expand $T(\theta)^T \eta(x_0, \ldots, x_T)$ as in 3. The form for $\log \hat{p}(\theta \mid x_{0:T})$ is in the exponential family. □

Proof of Theorem 4

Theorem. *Let $p_T(\theta \mid x_{0:T})$ denote the Gibbs distribution and $\widehat{p}_T(\theta \mid x_{0:T})$ its order M exponential family approximation. Assume that parameter θ has support $\mathcal{S}_\theta$ and finite variance. Then as $M \to \infty, T \to \infty$, the KL divergence between p_T and $\widehat{p}_T$ goes to zero.*

$$\lim_{M,T\to\infty} D_{KL}(p_T \,||\, \hat{p}_T) = 0$$

Proof. Assume that function f has bounded derivatives and bounded support I. Then the maximum error satisfies $\left| f_\theta(x_{k-1}) - \hat{f}_\theta(x_{k-1}) \right| \leq \epsilon_k$. It follows that $\hat{f}_\theta(x_{k-1})^2 - f_\theta(x_{k-1})^2 = -\epsilon_k^2 - 2\hat{f}_\theta(x_{k-1})\epsilon_k \approx -2\hat{f}_\theta(x_{k-1})\epsilon_k$.

Then the KL-divergence between the real posterior and the approximated posterior satisfies the following formula.

$$D_{KL}(p_T||\hat{p}_T) \tag{26}$$
$$= \int_{\mathcal{S}_\theta} \left(\frac{1}{\sigma^2} \sum_{k=1}^{T} \epsilon_k (x_k - \hat{f}_\theta(x_{k-1})) \right) p_T(\theta|x_{0:T}) d\theta$$

Moreover, recall that as $T \to \infty$ the posterior shrinks to $\delta(\theta - \theta^*)$ by the assumption of identifiability. Then we can rewrite the KL-divergence as (assuming Taylor approximation centered around θ_c)

$$\lim_{T\to\infty} D_{KL}(p_T||\hat{p}_T) \tag{27}$$
$$= \frac{1}{\sigma^2} \lim_{T\to\infty} \sum_{k=1}^{T} \epsilon_k \int_{\mathcal{S}_\theta} (x_k - \hat{f}_\theta(x_{k-1})) p_T(\theta|x_{0:T}) d\theta$$
$$= \frac{1}{\sigma^2} \lim_{T\to\infty} \sum_{k=1}^{T} \epsilon_k \cdot \tag{28}$$
$$\left(x_k - \sum_{i=0}^{M} H^i(x_{k-1}) \int_{\mathcal{S}_\theta} (\theta - \theta_c)^i p(\theta|x_{0:T}) d\theta \right)$$
$$= \frac{1}{\sigma^2} \lim_{T\to\infty} \sum_{k=1}^{T} \epsilon_k \left(x_k - \sum_{i=0}^{M} H^i(x_{k-1}) (\theta^* - \theta_c)^i \right)$$

If the center of the Taylor approximation θ_c is the true parameter value θ^*, we can show that

$$\lim_{T\to\infty} D_{KL}(p_T||\hat{p}_T) = \frac{1}{\sigma^2} \lim_{T\to\infty} \sum_{k=1}^{T} \epsilon_k \left(x_k - f_{\theta^*}(x_{k-1}) \right))$$
$$= \frac{1}{\sigma^2} \lim_{T\to\infty} \sum_{k=1}^{T} \epsilon_k v_k = 0 \tag{29}$$

where the final statement follows from law of large numbers. Thus, as $T \to \infty$, the Taylor approximation of any order will converge to the true posterior given that $\theta_c = \theta^*$. For an arbitrary center value θ_c,

$$D_{KL}(p_T||\hat{p}_T) = \frac{1}{\sigma^2} \sum_{k=1}^{T} \epsilon_k \left(x_k - \sum_{i=0}^{M} H^i(x_{k-1}) (\theta^* - \theta_c)^i \right) \tag{30}$$

Notice that $\epsilon_k \propto \frac{1}{(M+1)!}$ (by our assumptions that f has bounded derivative and is supported on interval I) and $H^i(\cdot) \propto \frac{1}{M!}$. The inner summation will be bounded since $M! > a^M, \forall a \in \mathbb{R}$ as $M \to \infty$. Therefore, as $M \to \infty$, $D_{KL}(p||\hat{p}) \to 0$. □

4 The Assumed Parameter Filter

In this section, we propose a novel algorithm for the general combined parameter and state estimation problem in SSMs. Our algorithm, called the Assumed Parameter Filter (APF) [Erol et al., 2017], is a hybrid of particle filtering for state variables and assumed density filtering for parameter variables. It projects the posterior distribution for parameters, $p(\theta \mid x_{0:t}, y_{0:t})$, into an approximation distribution and generates samples of parameters in constant-time per update. APF streams data and is a nearly-black-box algorithm: when an appropriate approximate distribution for the parameter is chosen, APF can be applied to any SSM that one can sample from and compute evidence likelihood. We emphasize the nearly-black-box property of APF by developing it as an automatic inference engine for a probabilistic programming language. Experiment results show that APF converges much faster than existing algorithms on a variety of models.

The design principles of APF are to inherit the appealing properties of the classical particle filter, which is applicable to arbitrary transitions and suitable for streaming data, while better overcoming the path degeneracy problem for parameter estimation without an expensive MCMC kernel.

We propose a nearly-black-box algorithm, called the Assumed Parameter Filter (APF), for online parameter and state estimation. In APF, the posterior of both states and parameters are jointly represented by K particles. The key point is that, unlike the bootstrap filter (which keeps a static parameter value in each particle), APF maintains an extra approximate distribution and samples from that distribution for the parameter at each time step.

In order to fight against sample impoverishment, for parameter θ_t^k at time t in particle k, we sample from a distribution $q_t^k(\theta)$ in some parametric family $\mathcal{Q}$ where $q_t^k(\theta)$ is the approximate representation of the true particle posterior $p(\theta \mid x_{0:t}^k, y_{0:t})$. In the special case where q is a delta function, APF recovers the bootstrap particle filter (proven in Section 4.11). In order to obtain the approximating distribution q_t^k from q_{t-1}^k, M additional Monte Carlo samples are utilized for each particle to perform the moment-matching operations required for assumed density approximation. The proposed method is illustrated in Alg. 6.

Following the assumed density filtering framework, we are approximating $p(\theta \mid x_{0:t}, y_{0:t})$ in a parametric distribution family $\mathcal{Q}$. In our algorithm this is expressed through the Update function. The Update function generates the approximating density q via minimizing the KL-divergence between the target p and the approximating distribution q.

4.1 Approximating $p(\theta \mid x_{0:t}, y_{0:t})$

At each time step with each new incoming data point we approximate the posterior distribution by a tractable and compact distribution from $\mathcal{Q}$. Our approach is inspired by assumed density filtering (ADF) for state estimation [Maybeck, 1982, Lauritzen, 1992, Boyen and Koller, 1998, Opper and Winther, 1998].

For our application, we are interested in approximately representing $p(\theta \mid$

Algorithm 6: Assumed Parameter Filter

Input: $y_{0:T}$, $\mathcal{Q}$, K, and M, the model (f_1, f_2, g, h)
Output: Samples $\left\{x_{0:T}^k, \theta_T^k\right\}_{k=1}^K$
Initialize $\left\{x_0^k, q_0^k(\theta)\right\}_{k=1}^K$ according to f_1, f_2 and $\mathcal{Q}$;
for $t = 1, \ldots, T$ **do**
  for $k = 1, \ldots, K$ **do**
    sample $\theta_t^k \sim q_{t-1}^k(\theta) \approx p(\theta \mid x_{0:t-1}^k, y_{0:t-1})$;
    sample $x_t^k \sim g(x_t \mid x_{t-1}^k, \theta_t^k)$;
    $w_t^k \leftarrow h(y_t \mid x_t^k, \theta_t^k)$;
    $q_t^k(\theta) \leftarrow \text{Update}(M, \mathcal{Q}; q_{t-1}^k(\theta), x_t^k, x_{t-1}^k, y_t)$;
  sample $\left\{\frac{1}{N}, \bar{x}_t^k, \bar{q}_t^k\right\} \sim \text{Multinomial}\left\{w_t^k, x_t^k, q_t^k\right\}$;
  $\left\{x_t^k, q_t^k\right\} \leftarrow \left\{\bar{x}_t^k, \bar{q}_t^k\right\}$;

$x_{0:t}, y_{0:t}$) in a compact form that belongs to a family of distributions. Due to the Markovian structure of the SSM, the posterior can be factorized as

$$p(\theta \mid x_{0:t}, y_{0:t}) \propto \prod_{i=0}^t s_i(\theta),$$
$$s_i(\theta) = \begin{cases} p(\theta)p(y_0 \mid x_0, \theta), i = 0 \\ p(x_i \mid x_{i-1}, \theta)p(y_i \mid x_i, \theta), i \geq 1 \end{cases}.$$

Let us assume that at time step $i-1$ the posterior was approximated by $q_{i-1} \in \mathcal{Q}$. Then with incorporation of (x_i, y_i), the posterior $\hat{p}$ will be

$$\hat{p}(\theta \mid x_{0:i}, y_{0:i}) = \frac{s_i(\theta)q_{i-1}(\theta)}{\int_\theta s_i(\theta)q_{i-1}(\theta)d\theta}. \tag{31}$$

For most models, $\hat{p}$ will not belong to $\mathcal{Q}$. ADF projects $\hat{p}$ into $\mathcal{Q}$ via minimizing KL-divergence:

$$q_i(\theta) = \arg\min_{q \in \mathcal{Q}} D\left(\hat{p}(\theta \mid x_{0:i}, y_{0:i}) \mid\mid q(\theta)\right) \tag{32}$$

For $\mathcal{Q}$ in the exponential family, minimizing the KL-divergence reduces to moment matching [Seeger, 2005]. For $q_i(\theta) \propto \exp\left\{\gamma_i^T m(\theta)\right\}$, where γ_i is the canonical parameter and $m(\theta)$ is the sufficient statistic, we compute moments of the one-step ahead posterior $\hat{p}$ and update γ_i to match.

$$\begin{aligned} g(\gamma_i) &= \int m(\theta)q_i(\theta)d\theta = \int m(\theta)\hat{p}(\theta)d\theta \\ &\propto \int m(\theta)s_i(\theta)q_{i-1}(\theta)d\theta \end{aligned}$$

where $g(\cdot)$ is the unique and invertible link function. Thus, for the exponential family, the Update function computes the moment matching integrals to update the canonical parameters of $q_i(\theta)$. For the general case, where these integrals

may not be tractable, we propose approximating them by a Monte Carlo sum with M samples, sampled from $q_{i-1}(\theta)$:

$$Z \approx \frac{1}{M}\sum_{j=1}^{M} s_i(\theta^j), \quad g(\gamma_i) \approx \frac{1}{MZ}\sum_{j=1}^{M} m(\theta^j)s_i(\theta^j)$$

where $\theta^j \sim q_{i-1}(\theta)$. In our framework, this approximation is done for all particle paths $x_{0:i}^k$ and the corresponding q_{i-1}^k, hence leading to $O(KM)$ sampling operations per time step.

4.2 Asymptotic performance for APF

In a similar spirit to Opper and Winther [1998], we prove that assumed density filtering framework can successfully converge to the target posterior with increasing amount of data. For simplicity, we only consider continuous parameters and use Gaussian as the approximation distribution. We assume an *identifiable* model (posterior is asmptotically Gaussian around the true parameter) and also assume that in the model, only the transition is parametrized by the parameter θ while the observation model is known.

Theorem 5. *Let $(f_1, f_2, g_\theta, h_\theta)$ be an identifiable Markovian SSM, and let $\mathcal{Q}$ be multivariate Gaussian. The KL divergence between $p(\theta \mid x_{0:t}, y_{0:t})$ and the assumed density $q_t(\theta)$ computed as explained in the previous subsection will converge to zero as $t \to \infty$.*

$$\lim_{t\to\infty} D_{\mathrm{KL}}\left(p(\theta \mid x_{0:t}, y_{0:t})||q_t(\theta)\right) = 0. \tag{33}$$

The proof is given in Section 4.11. The theorem states that the error due to the projection diminishes in the long-sequence limit. Therefore, with $K, M \to \infty$, APF would produce samples from the true posterior distribution $p(\theta, x_t \mid y_{0:t})$. For finite K, however, the method is susceptible to path degeneracy.

Similar to Storvik [2002], Lopes et al. [2010], we are sampling from $p(\theta \mid x_{0:t}^i, y_{0:t})$ at each time step to fight against sample impoverishment. It has been discussed before that these methods suffer from *ancestral path degeneracy* [Chopin et al., 2010, Lopes et al., 2010, Poyiadjis et al., 2011]. For any number of particles and for a large enough n, there exists some $m < n$ such that $p(x_{0:m} \mid y_{0:n})$ is represented by a single unique particle. For dynamic models with long memory, this will lead to a poor approximation of sufficient statistics, which in turn will affect the posterior of the parameters. Poyiadjis et al. [2011] have shown that even under favorable mixing assumptions, the variance of an additive path functional computed via a particle approximation grows quadratically with time. To fight against path degeneracy, one may have to resort to *fixed-lag smoothing* or *smoothing*. Olsson et al. [2008] used fixed-lag smoothing to control the variance of the estimates . Del Moral et al. [2010] proposed an $O(K^2)$ per time step forward smoothing algorithm which leads to variances growing linearly with t instead of quadratically. Poyiadjis et al. [2011] similarly proposed an $O(K^2)$ algorithm that leads to linearly growing variances . These techniques can be augmented into the APF framework to overcome the path degeneracy problem for models with long memory.

4.3 Special cases: Gaussians, mixtures of Gaussians and discrete distributions

4.3.1 Gaussian case:

For a multivariate Gaussian $\mathcal{Q}$, explicit recursions can be derived for $\hat{p}(\theta) \propto s_i(\theta)q_{i-1}(\theta)$ where $q_{i-1}(\theta) = \mathcal{N}(\theta; \mu_{i-1}, \Sigma_{i-1})$. The moment matching recursions are

$$\begin{array}{l} \mu_i = \frac{1}{Z}\int \theta s_i(\theta)q_{i-1}(\theta)d\theta, \\ \Sigma_i = \frac{1}{Z}\int \theta\theta^T s_i(\theta)q_{i-1}(\theta)d\theta - \mu_i\mu_i^T. \end{array} \tag{34}$$

where $Z = \int \hat{p}(\theta)d\theta = \int s_i(\theta)q_{i-1}(\theta)d\theta$. These integrals can be approximated via Monte Carlo summation as previously described. One alternative is *deterministic sampling*. Since q is multivariate Gaussian, Gaussian quadrature rules can be utilized. In the context of expectation-propagation this has been proposed by Zoeter and Heskes [2005]. In the context of Gaussian filtering, similar quadrature ideas have been applied as well [Huber and Hanebeck, 2008].

For an arbitrary polynomial $f(x)$ of order up to $2M-1$, $\int f(x)e^{-x^2}dx$ can be calculated exactly via Gauss-Hermite quadrature with M quadrature points. Hence, the required moment matching integrals in Eq.(34) can be approximated arbitrarily well by using more quadrature points. The unscented transform [Julier and Uhlmann, 2004] is one specific Gaussian quadrature rule that uses $M = 2d$ *deterministic* samples to approximate an integral involving a d-dimensional multivariate Gaussian. In our case these samples are: $\forall 1 \leq j \leq d$,

$$\theta_j = \mu_{i-1} + \left(\sqrt{d\Sigma_{i-1}}\right)_j, \quad \theta_{d+j} = \mu_{i-1} - \left(\sqrt{d\Sigma_{i-1}}\right)_j.$$

where $(\cdot)_j$ means the jth column of the corresponding matrix. Then, one can approximately evaluate the moment matching integrals as follows:

$$\begin{array}{l} Z \approx \frac{1}{2d}\sum_{j=1}^{2d} s_i(\theta^j), \quad \mu_i \approx \frac{1}{2dZ}\sum_{j=1}^{2d} \theta^j s_i(\theta^j), \\ \Sigma_i \approx \frac{1}{2dZ}\sum_{j=1}^{2d} \theta^j(\theta^j)^T s_i(\theta^j) - \mu_i\mu_i^T. \end{array}$$

4.3.2 Mixtures of Gaussians:

Weighted sums of Gaussian probability density functions can be used to approximate another density function arbitrarily closely. Gaussian mixtures have been used for state estimation since the 1970s [Alspach and Sorenson, 1972].

Let us assume that at time step $i-1$ it was possible to represent $p(\theta \mid x_{0:i-1}, y_{0:i-1})$ as a mixture of Gaussians with L components.

$$\begin{array}{rcl} p(\theta \mid x_{0:i-1}, y_{0:i-1}) & = & \sum_{m=1}^{L} \alpha_{i-1}^m \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m) \\ & = & q_{i-1}(\theta) \end{array}$$

Given x_i and y_i;

$$\hat{p}(\theta \mid x_{0:i}, y_{0:i}) \propto \sum_{m=1}^{L} \alpha_{i-1}^m s_i(\theta) \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m)$$

The above form will not be a Gaussian mixture for arbitrary s_i. We can rewrite it as:

$$\hat{p} \propto \sum_{m=1}^{L} \alpha_{i-1}^m \beta^m \frac{s_i(\theta) \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m)}{\beta^m} \tag{35}$$

where the fraction is to be approximated by a Gaussian via moment matching and the weights are to be normalized. Here, each $\beta^m = \int s_i(\theta) \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m) d\theta$ describes how well the m-th mixture component explains the new data. That is, a mixture component that explains the new data well will get up-weighted and vice versa. It is important to note that the proposed approximation is not exactly an ADF update. The problem of finding a mixture of fixed number of components to minimize the KL divergence to a target distribution is intractable [Hershey and Olsen, 2007]. The proposed update here is the one that matches the mean and covariance.

The resulting approximated density would be $q_i(\theta) = \sum_{m=1}^{K} \alpha_i^m \mathcal{N}(\theta; \mu_i^m, \Sigma_i^m)$ where the recursion for updating each term is as follows:

$$\begin{aligned}
\beta^m &= \int s_i(\theta) \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m) d\theta \\
\alpha_i^m &= \frac{\alpha_{i-1}^m \beta^m}{\sum_\ell \alpha_{i-1}^\ell \beta^\ell} \\
\mu_i^m &= \frac{1}{\beta^m} \int \theta s_i(\theta) \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m) d\theta \\
\Sigma_i^m &= \frac{1}{\beta^m} \int \theta\theta^T s_i(\theta) \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m) d\theta - \mu_i^m (\mu_i^m)^T
\end{aligned}$$

Similar to the Gaussian case, the above integrals are generally intractable. Either a Monte Carlo sum or a Gaussian quadrature rule can be utilized to approximately update the means and covariances.

4.3.3 Discrete parameter spaces:

Let us consider a d-dimensional parameter space where each parameter can take at most N_θ values. For discrete parameter spaces, one can always track $p(\theta \mid x_{0:t}, y_{0:t})$ exactly with a constant-time update; the constant, however, is exponential in d [Boyen and Koller, 1998]. Hence, tracking the sufficient statistics becomes computationally intractable with increasing dimensionality. For discrete parameter spaces we propose projection onto a fully factorized distribution, i.e., $q_i(\theta) = \prod_j q_{j,i}(\theta_j)$. For this choice, minimizing the KL-divergence reduces to matching marginals:

$$\begin{aligned}
Z &= \textstyle\sum_\theta s_i(\theta) q_{i-1}(\theta) \\
q_{j,i}(\theta_j) &= \tfrac{1}{Z} \textstyle\sum_{\theta \setminus \theta_j} s_i(\theta) q_{i-1}(\theta).
\end{aligned}$$

Computing these summations is intractable for high-dimensional models, hence we propose using Monte Carlo summation. In the experiments, we consider a simultaneous localization and mapping problem with a discrete map.

4.4 Discussions

4.4.1 Applicability:

APF follows the framework of particle filtering and the only requirement for updating the approximate distribution is being able to compute the likelihood of the states conditioned on the sampled parameter. Thus, APF can be applied to the same category of models as the particle filter. The critical issue for APF is the choice of the family of approximation distributions $\mathcal{Q}$. Although Gaussian mixtures can arbitrarily approximate any density, different forms of $\mathcal{Q}$ can significantly improve the practical performance. For example, when the dimensionality of the parameter space is large, one may want to use a diagonal Gaussian distribution for fast convergence; for non-negative parameters, the gamma distribution may be favored. Note that different $\mathcal{Q}$s yield different updating formulae. To this perspective, APF is nearly-black-box: a user can apply APF to arbitrary SSMs just by choosing an appropriate approximating distribution. In the next section, we further explore the nearly-black-box property of APF by adapting it to the backend inference engine of a probabilistic programming language.

4.4.2 Modularity:

One can utilize a Storvik filter for a subset of parameters with fixed-dimensional sufficient statistics, and for the rest of the parameters, approximate sufficient statistics can be generated via the APF framework. This is similar to the extended Liu-West filter [Rios and Lopes, 2013] where a Storvik filter is used in conjunction with the artificial dynamics approach.

4.4.3 Complexity:

The time complexity of APF is $O(MKT)$ over T time steps for K particles with M extra samples to update the sufficient statistics through the moment matching integrals. Setting K and M adequately is crucial for performance. Small K prevents APF exploring the state space sufficiently whereas small M leads to inaccurate sufficient statistics updates which will in turn result in inaccurate parameter estimation.

Note that the typical complexity of PMCMC algorithms is $O(NKT)$ where N denotes the number of MCMC samples. Although in the same order of APF for time complexity, we find in practice that M is often orders of magnitude smaller than N for achieving a given level of accuracy. PMCMC algorithms often requires a large amount of MCMC iterations for mixing properly while very small M is sufficient for APF to produce accurate parameter estimation, especially for the Gaussian case as discussed above.

Moreover, the actual running time for APF is often much smaller than its theoretical upper bound $O(MKT)$. Notice that the approximation computation in APF only requires the local data in a single particle and does not influence the weight of that particle. Hence, one important optimization specialized for APF is to resample all the particles prior to the update step and only update the approximate distribution for those particles that do not disappear after resampling. It is often the case that a small fraction of particles have significantly large weights. Consequently, in our experiment, the actual running time of APF is several times faster than the theoretically required time $O(MKT)$ (see practical performances of APF in the Experiment section).

Lastly, the space complexity for APF is in the same order as the bootstrap particle filter, namely $O(K)$. Overall, APF is constant time and memory per update and hence fits into online/streaming applications.

4.5 Using APF in probabilistic programming

This section shows APF can be integrated into a probabilistic programming language (PPL), BLOG [Milch et al., 2005], from which the general research community can benefit. PPLs aim to allow users to express an arbitrary Bayesian model via a probabilistic program while the backend engine of PPL automatically performs black-box inference over the model. PPLs largely simplify the development process of AI applications involving rich domain knowledge and have led to many successes, such as human-level concept learning [Lake et al., 2015], global seismic monitoring [Arora, 2012], and 3D scene perception [Kulkarni et al., 2015].

We developed a compiled inference engine, the State and Parameter Estimation Compiler (SPEC), utilizing APF under BLOG [Milch et al., 2005] thanks to its concise syntax [Li and Russell, 2013]: in the BLOG language, state variables are those indexed by timestep, while all other variables are effectively parameters; thus, by identifying the static and dynamic variables, the SPEC compiler can automatically work out how to apply APF to the filtering problem. Since APF is based on the BLOG language, the general compilation process is based on the Swift compiler for BLOG [Wu et al., 2016]. There are also other compiled PPL systems, such as Church [Goodman et al., 2008] and Anglican [Wood et al., 2014]. However, these systems do not have any language primitives distinguishing state and parameter. Potentially APF can be also applied to these systems by adding new syntax for declaring the parameter.

In order to demonstrate the online property of APF, SPEC also includes some extended syntax to allow streaming applications. The following BLOG program describes a simple SSM, the `SIN` model:

$$\begin{array}{cc} X_0 \sim \mathcal{N}(0,1) & \Theta \sim \mathcal{N}(0,1) \\ X_t \sim \mathcal{N}(\sin(\theta x_{t-1}),1) & Y_t \sim \mathcal{N}(x_t, 0.5^2) \end{array}$$

```
1  // parameter
2  random Real theta ~ Gaussian(0,1);
3  // state X(t)
4  random Real X(Timestep t) ~
5    // initial value, X(0)
6    if t == @0 then Gaussian(0, 1)
7    // transition
8    else Gaussian(sin(theta * X(t - @1)), 1);
9  // observed variable Y(t)
10 random Real Y(Timestep t)~Gaussian(X(t),0.25);
11 // user declared C++ function
12 extern Real loadData(Timestep t);
13 // observations
14 obs Y(t) = loadData(t) for Timestep t;
15 // query states and the parameter
16 query X(t) for Timestep t;
17 query theta;
```

The keyword `random` declares random variables in the model: those with an argument of type `Timestep` are states (dynamic variables, i.e., `X(t)` and `Y(t)`) while others are parameters (static variables, i.e., `theta`). Line 14 states that `Y(t)` is observed while line 16 and 17 query the posterior distribution of the state `X(t)` at each time step and the parameter `theta`.

In SPEC, we extend the original syntax of BLOG by (1) introducing a new keyword `extern` (Line 12) to import arbitrary customized C++ functions (e.g., input functions for streaming data at each time step) and (2) the for-loop style observation statement (Line 14) and query statement (Line 16, 17). These changes allow APF to be applied in a completely online fashion.

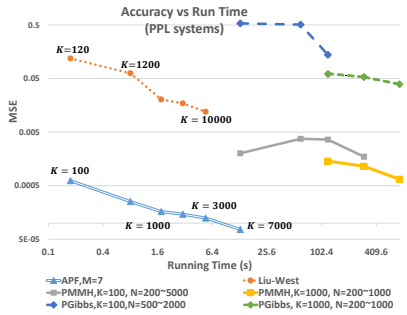
A user can utilize SPEC to perform inference with APF for both $\{X_t\}$ and Θ by simply coding this tiny program without knowing algorithm details. SPEC automatically analyzes the parameters, selects approximate distributions and applies APF to this model. By default, we use Gaussian distributions with Gauss-Hermite quadratures for continuous parameters and factored categorical distributions for discrete parameters. SPEC is extensible for more approximate distributions for further development. Moreover, due to the memory efficiency of APF, many other optimizations from the programming language community can be applied to further accelerate the practical performance of APF.[1]

4.6 Experiments

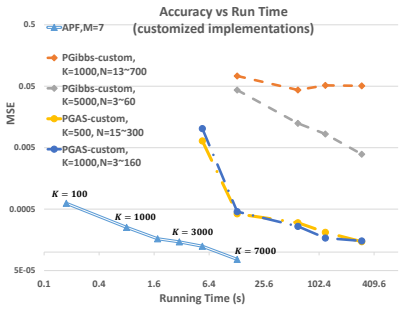
We evaluated APF on three benchmark models: 1. `SIN`: a nonlinear dynamical model with a single continuous parameter; 2. `SLAM`: a simultaneous localization and Bayesian map learning problem with 20 discrete parameters; 3. `BIRD`: a 4-parameter model to track migrating birds with real-world data. We compare the estimation accuracy of APF, as a function of run time, against the Liu-West filter (LW) and PMCMC algorithms including particle marginal Metropolis-Hastings (PMMH), particle Gibbs (PGibbs) and particle Gibbs with ancestor sampling (PGAS). For implementation, APF and LW are natively supported by our automatic engine SPEC. PMMH is manually adapted from the code

[1] The details of the compilation optimizations in SPEC can be found in the supplementary materials.

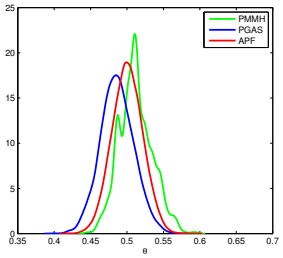
compiled by SPEC. For PGibbs and PGAS, we compare both the code generated the Anglican compiler [Wood et al., 2014], and our customized implementations.



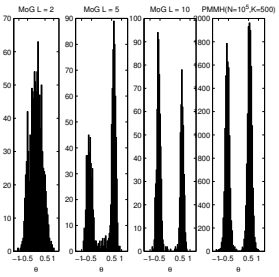
(a) Accuracy on `SIN` using black-box algorithms in PPLs



(b) Accuracy on `SIN` via customized implementations



(c) Density estimation on `SIN`



(d) Histograms for multimodal `SIN`

Figure 8: Performance plots in `SIN`, `SLAM`, `BIRD` experiments: (a) accuracy on `SIN` comparing APF with other black-box algorithms for PPLs (Liu-West, PMMH from SPEC and PGibbs by Anglican); (b) accuracy on `SIN` comparing APF with our customized implementations of PGibbs and PGAS in C++; (c) density estimation of the posterior distribution of θ by APF, PMMH and PGAS; (d) the histograms of samples for θ in the multimodal `SIN` model by APF using $L = 2, 5, 10$ mixtures of Gaussians and the almost-true posterior by PMMH;

4.7 Toy nonlinear model (`SIN`)

We consider the `SIN` model in the previous section with the true parameter $\theta^\star = 0.5$. 5000 data points are generated to ensure a sharp posterior. Notice that it is not possible to use the Storvik filter [Storvik, 2002] or the particle learning algorithm [Lopes et al., 2010] for this model as sufficient statistics do not exist for Θ.

We evaluate the mean squared error over 10 trials between the estimation results and $\theta^\star$ within a fixed amount of time. For APF and LW, we consider the

mean of the samples for Θ at the last time step for parameter estimation while for PMCMC algorithms, we take the average of the last half of the samples and leave the first half as burn-in. We omit PGAS results by Anglican here since Anglican takes more than 10 minutes to produce a sample with 100 particles.

For APF, we choose Gaussian as the approximate distribution with $M = 7$. For PMMH, we use a local truncated Gaussian as the proposal distribution for Θ.

Note that for PGAS and PGibbs, we need to sample from $\Pr[\Theta|X_{1:T}, Y_{1:T}]$ while this cannot be efficiently computed in `SIN`. As a black-box inference system, the Anglican compiler avoids this by treating every variable as state variable. In our customized implementations, we use a piecewise linear function to approximate $\Pr[\Theta|X_{1:T}, Y_{1:T}]$[2].

The results for black-box algorithms, including APF, Liu-West filter (supported by SPEC), PMMH (adapted from the code by SPEC) and PMCMC algorithms (PGibbs and PGAS in Anglican) are shown in Fig. 8(a). We also compare APF against out customized implementation of PGibbs and PGAS in Fig. 8(b). APF produced a result of orders of magnitude smaller error within a much smaller amount of run time: an estimation for θ with $1.6 * 10^{-4}$ square error with only 1000 particles in 1.5 seconds.

Note that, as mentioned in the discussion section, in theory APF with $M = 7$ should be 7 times slower than the plain particle filter. But in practice, thanks to the trick – only updating the approximate distributions for the retained particles, APF is just 2 times slower.

4.7.1 Density Estimation:

We also show the kernel density estimates of the posterior of θ in Fig. 8(c), where we ran APF with 10^4 particles and $M = 7$ as well as our customized version of PGibbs and PGAS with 5000 particles for 6 hours (around 5000 MCMC iterations). For PGibbs and PGAS, we left the first 1000 samples as burn-in. APF produced a reasonable mode centered exactly at the true parameter value 0.5 using merely 40 seconds while PGibbs and PMMH mixed slowly.

4.7.2 Bimodal Variant:

Consider a multimodal variant of `SIN` as follows: $X_t \sim \mathcal{N}(\sin(\theta^2 x_{t-1}), 1)$, $Y_t \sim \mathcal{N}(x_t, 0.5^2)$. Due to the θ^2 term, $p(\theta \mid y_{0:t})$ will be bimodal. We generate 200 data points in this case and execute APF with $K = 10^3$ particles and $M = 7$ using mixtures of $L = 2, 5, 10$ Gaussians as the approximate distribution. To illustrate the true posterior, we ran PMMH with $K = 500$ for 20 minutes (much longer than APF) to ensure it mixes properly.

The histograms of the samples for θ are demonstrated in Fig. 8(d). APF successfully approximates the multimodal posterior when $L = 5, 10$ and the

[2]We discretize $[-1, 1]$ uniformly into 500 intervals. 500 is smaller than the number of particles used by the PMCMC algorithms, so this process does not influence the total running time significantly.

weights are more accurate for $L = 10$. For $L = 2$, APF only found a single mode with a large bias. This suggests that increasing the number of mixture components used for approximation can help find different modes in the true posterior in practice.

4.8 Simultaneous localization and mapping (SLAM)

We consider a simultaneous localization and mapping example (SLAM) modified from [Murphy et al., 1999]. The map is defined as a 1-dimensional grid, where each cell has a static label (parameter to be estimated) which will be (noisily) observed by the robot. More formally, the map is a vector of boolean random variables $M(i) \in \{1, \ldots, N_O\}$, where $1 \leq i \leq N_L$. Neither the map nor the robot's location $L_t \in \{1, \ldots, N_L\}$ is observed.

Given the action, move right or left, the robot moves in the direction of action with a probability of p_a and stays at its current location with a probability of $1-p_a$ (i.e., robot's wheels slip). The prior for the map is a product of individual cell priors, which are all uniform. The robot observes the label of its current cell correctly with probability p_o and incorrectly with probability of $1 - p_o$. In the original example, $N_L = 8$, $p_a = 0.8$, $p_o = 0.9$, and 16 actions were taken. In our experiment, we make the problem more difficult by setting $N_L = 20$ and deriving a sequence of 41 actions to ensure the posterior converge to a sharp mode.

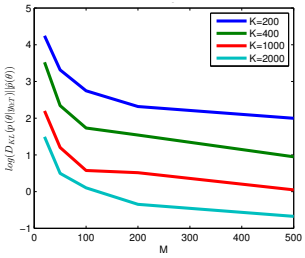
We evaluate the KL-divergence between the prediction posterior and the exact true posterior within various time limits. We omit Liu-West filter since is not applicable for discrete parameters. In APF, we use a Bernoulli distribution as the approximate distribution for every grid cell. For PMMH, we use a coordinate MCMC kernel: we only sample a single grid at each MCMC iteration. For PGibbs and PGAS, since it is hard to efficiently sample from $\Pr[\theta|X_{1:T}, Y_{1:T}]$, we only show results by Anglican.

Fig. 9(b) shows that APF approximates the posterior distribution much more accurately than other methods within a shorter run time. For PGAS by Anglican, due to its system overhead, the overall run time is significantly longer.

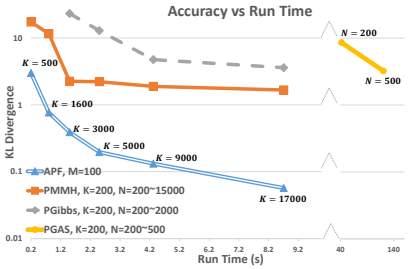
Similar to SIN, although APF performs $20M = 2000$ extra samples per time step in SLAM, in practice, APF is merely 60x slower than the plain particle filter due to the resampling trick.

4.8.1 Choices of Parameters:

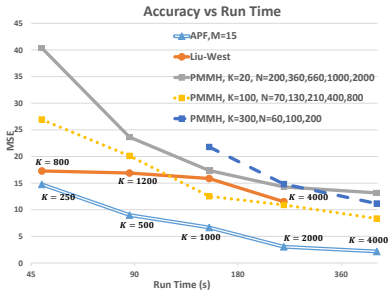
We experiment APF with various settings (number of particles K and number of samples M) and evaluate the average log KL-divergence over 100 trials. The results in Fig. 9(a) agree with the theory. As K increases the KL-divergence decreases whereas after a certain point, not much gain is obtained by increasing M. When M is large enough, the moment matching integrals are more or less exactly computed and the error is not due to the Monte Carlo sum but due to the error induced by the assumed-density projection step, which cannot be avoided.



(a) APF with different parameters on `SLAM`



(b) Accuracy on `SLAM`



(c) Accuracy on `BIRD`

Figure 9: Performance plots in `SLAM`, `BIRD` experiments: (a) APF with different configurations for K and M on `SLAM`; (b) accuracy on `SLAM`; (c) accuracy on `BIRD`.

4.9 Tracking bird migration (`BIRD`)

The bird migration problem (`BIRD`) is originally investigated in [Elmohamed et al., 2007], which proposes a hidden Markov model to infer bird migration paths from a large database of observations[3].

In the `BIRD` model, there are 4 continuous parameters with 60 dynamic states where each time step contains 100 observed variables and more than 10^4 hidden variables.

We again measure the mean squared estimation error over 10 trials between the average of the samples for the parameters and the ground truth within different time limits. For APF, we use a diagonal Gaussian approximation with $M = 15$. For PMMH we use a truncated Gaussian proposal with diagonal covariance and leave the first half of the samples as burn-in. We did not compare against PGAS and PGibbs since these algorithms require storing the full history, which consumes too much memory (60x larger) to run enough particles. The results illustrated in Fig. 9(c) again show that APF achieves much better convergence within a much tighter computational budget.

[3] http://ppaml.galois.com/wiki/wiki/CP2BirdMigration

4.10 Concluding Remarks

We proposed the assumed parameter filter (APF), an online algorithm for joint parameter and state estimation in general state-space models, which provably converges in the long-sequence limit under standard conditons.

It is a "nearly-black-box" algorithm in the sense that its default assumed-density models can handle a large range of cases, while the algorithm can be extended easily to new cases by supplying a well-defined set of functions. APF is not a drop-in replacement for unbiased algorithms with theoretical guarantees, e.g., PMCMC and SMC^2 [Chopin et al., 2013], but an efficient alternative in practice. Experiments exhibit that APF has better estimation performance using much less computation time compared to several standard algorithms on a variety of applications.

4.11 Supplementary Material

Bootstrap particle filter as a subcase of APF

Here we will show that when q is a delta function, APF recovers the bootstrap particle filter. The Dirac delta function can be considered as the limit of a Gaussian as the variance goes to zero, $\delta(\theta-\mu) = \lim_{\sigma^2\to 0}\mathcal{N}(\theta;\mu,\sigma^2)$. Therefore, we can view q as an exponential family distribution. Specifically we are dealing with a Gaussian distribution with unknown mean and known variance (zero-variance). Then the moment matching integral required for assumed density filtering reduces to matching the means. If $q_{i-1} = \delta(\theta - \mu_{i-1})$, then

$$\begin{aligned}\mu_i = \int \theta q_i(\theta)d\theta &= \int \theta\hat{p}(\theta)d\theta \\ &= \frac{\int \theta s_i(\theta)\delta(\theta-\mu_{i-1})d\theta}{\int s_i(\theta)\delta(\theta-\mu_{i-1})d\theta} \\ &= \frac{\mu_{i-1}s_i(\mu_{i-1})}{s_i(\mu_{i-1})} = \mu_{i-1} \end{aligned} \tag{36}$$

where the last equality follows from the *sifting property* of the Dirac delta function. The main result here is that for $\mathcal{Q}$ Dirac delta, $\mu_i = \mu_{i-1}$; that is, APF Update step does not propose new values. Therefore, our proposed algorithm recovers the standard bootstrap particle filter.

Proof for Theorem 5

Theorem. *Let $(f_1, f_2, g_\theta, h_\theta)$ be an identifiable Markovian SSM, and let $\mathcal{Q}$ be multivariate Gaussian. The KL divergence between $p(\theta \mid x_{0:t}, y_{0:t})$ and the assumed density $q_t(\theta)$ computed as explained in the previous subsection will converge to zero as $t \to \infty$.*

$$\lim_{t\to\infty} D_{\mathrm{KL}}\left(p(\theta \mid x_{0:t}, y_{0:t}) || q_t(\theta)\right) = 0. \tag{37}$$

In this section[4] we will assume an identifiable model where the posterior distribution approaches normality and concentrates in the neighborhood of the posterior mode. Suppose $\hat{\theta}$ is the posterior mode and hence the first-order partial derivatives of $\log p(\theta \mid x_{0:T})$ vanish at $\hat{\theta}_n$. Define

$$\hat{I} = -\frac{\partial^2 \log p(\theta \mid x_{0:T})}{\partial\theta\partial\theta^T}\bigg|_{\theta=\hat{\theta}} \tag{38}$$

Applying a second-order Taylor approximation around $\hat{\theta}$ to the posterior density results in

$$\log p(\theta \mid x_{0:T}) \approx \log p(\hat{\theta} \mid x_{0:T}) - \frac{1}{2}(\theta-\hat{\theta})^T\hat{I}(\theta-\hat{\theta}) \tag{39}$$

Hence;

$$p(\theta \mid x_{0:T}) \propto \exp\left\{-\frac{1}{2}(\theta-\hat{\theta})^T\hat{I}(\theta-\hat{\theta})\right\} \tag{40}$$

which is a p ($\theta \in \mathbb{R}^p$) dimensional Gaussian density with mean $\hat{\theta}$ and covariance $\hat{I}^{-1}$. As the posterior becomes highly concentrated in a neighborhood of the posterior mode, the effect of the prior on the posterior diminishes, which is the Bernstein-von Mises theorem. Then we can rewrite Equation 40 as

$$p(\theta \mid x_{0:T}) \propto \exp\left\{-\frac{T}{2}\sum_{ij}(\theta_i-\hat{\theta}_i)\hat{J}_{ij}(\theta_j-\hat{\theta}_j)\right\} \tag{41}$$

where $\hat{J}_{ij} = -\partial_i\partial_j\frac{1}{T}\sum_{t=1}^{T}\log p(x_t \mid x_{t-1},\hat{\theta})$.

The assumed density filter updates for the Gaussian case has been derived in earlier sections. We will reorganize them in a more convenient form.

$$\begin{aligned}
\mu_i(t) &= \frac{\int \theta_i p(x_t \mid x_{t-1},\theta)q_{t-1}(\theta)d\theta}{\int p(x_t \mid x_{t-1},\theta)q_{t-1}(\theta)d\theta} \\
\Sigma_{ij}(t) &= \frac{\int \theta_i\theta_j p(x_t \mid x_{t-1},\theta)q_{t-1}(\theta)d\theta}{\int p(x_t \mid x_{t-1},\theta)q_{t-1}(\theta)d\theta} - \mu_i(t)\mu_j(t)
\end{aligned}$$

We will use a simple property of centered Gaussian random variables z, $E[zf(z)] = E(f'(z)).E(z^2)$ which can be proven by applying integration by parts. Then the explicit updates can be written as follows:

$$\mu_i(t) = \mu_i(t-1) + \sum_j \Sigma_{ij}(t-1) \times \partial_j \log E_u[p(x_t \mid x_{t-1}, \mu(t-1)+u)] \tag{42}$$

$$\Sigma_{ij}(t) = \Sigma_{ij}(t-1) + \sum_{kl}\Sigma_{ik}(t-1)\Sigma_{lj}(t-1) \times \partial_k\partial_l \log E_u[p(x_t \mid x_{t-1}, \mu(t-1)+u)]$$

[4]Our discussion follows Opper and Winther [1998] which considers the asymptotic performance of assumed density filtering for independent identically distributed data. We are also assuming that parameters only affect the hidden states x_t for simplifying the analysis.

where u is a zero-mean Gaussian random vector with covariance $\Sigma(t-1)$. We define $V_{kl} = \partial_k \partial_l \log E_u[p(x_t \mid x_{t-1}, \theta + u)]$ and assume that for large times we can replace the difference equation for $\Sigma(t)$ with a differential equation. Then we can rewrite Equation 42 as

$$\frac{d\Sigma}{dt} = \Sigma V \Sigma \tag{43}$$

which is solved by

$$\frac{d\Sigma^{-1}}{dt} = -V \tag{44}$$

Integrating both sides

$$\Sigma^{-1}(t) - \Sigma^{-1}(t_0) = -\int_{t_0}^{t} V(\tau) d\tau \tag{45}$$

For large t ; we expect the covariance Σ to be small such that $\log E_u[p(x_t \mid x_{t-1}, \mu + u)] = \log p(x_t \mid x_{t-1}, \mu)$. Assuming that the online dynamics is close to θ^* and dividing both sides of Equation 45 by t and taking the limit $t \to \infty$, we get

$$\lim_{t\to\infty} \frac{(\Sigma^{-1}(t))_{ij}}{t} = \lim_{t\to\infty} \frac{-\int_{t_0}^{t} \partial_i \partial_j \log p(x' \mid x, \theta^*)}{t} \tag{46}$$

Further assuming ergodicity (i.e., markov process converging to some stationary distribution π), we can replace the time average with the probabilistic average.

$$\lim_{t\to\infty} \frac{(\Sigma^{-1}(t))_{ij}}{t} = -\int \pi(x) p(x' \mid x, \theta^*) \partial_i \partial_j \log p(x_2 \mid x_1, \theta^*) dx dx' \tag{47}$$

If we define the right hand side as $A_{ij} = -\int \pi(x) p(x' \mid x, \theta^*) \partial_i \partial_j \log p(x_2 \mid x_1, \theta^*) dx dx'$ we have;

$$\lim_{t\to\infty} \Sigma(t) = \frac{A^{-1}}{t} \tag{48}$$

We will also analyze the asymptotic scaling of the estimation error, defined as the deviation between θ^* and $\mu(t)$. Assuming that the estimate μ is close to θ^* and the posterior is sharply concentrated we can neglect the expectation with respect to u in Equation 42. Defining $\mu_i(t) = \theta_i^* + \epsilon_i(t)$, and applying a first order Taylor approximation around θ^* we get;

$$\epsilon_i(t+1) - \epsilon_i(t) = \sum_{\ell} \Sigma_{i\ell} \partial_\ell \log P + \sum_{k\ell} \Sigma_{i\ell} \epsilon_k(t) \partial_k \partial_l \log P \tag{49}$$

where $P \equiv p(x' \mid x, \theta^*)$. Taking the expectation with respect to the stationary distribution (denoted by an overbar) and using the relationship in Equation 48 and replacing the difference equation with a differential equation we get an equation of motion for the expected error $e_i = \bar{\epsilon_i}$.

$$\frac{de_i}{dt} + \frac{e_i}{t} = \sum_j \frac{(A^{-1})_{ij}}{t} \overline{\partial_j \log P} \tag{50}$$

As $t \to \infty$ right hand side vanishes and hence the error term decays like $e_i \propto \frac{1}{t}$.

Revisiting Equation 40, the true posterior covariance matrix is given by $C^{-1} = T\hat{J}$. Due to our ergodicity assumption, $\lim_t \hat{J} = A$. Hence the true posterior density covariance asymptotically converges to A^{-1}/T which is the same limit as the assumed density filter covariance $\Sigma(t)$.

KL divergence between two d-dimensional multivariate Gaussians, $\mathcal{N}(\mu_1, \Sigma_1)$ and $\mathcal{N}(\mu_2, \Sigma_2)$ is given by

$$\frac{1}{2}\left[\log\frac{|\Sigma_2|}{|\Sigma_1|} - d + \mathrm{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_2 - \mu_1)^T\Sigma_2^{-1}(\mu_2 - \mu_1)\right] \tag{51}$$

We have shown that $\lim_{t\to\infty} C, \Sigma = A^{-1}/t$ and $\mu(t) \to \theta^*$. Due to the identifiability assumption, the posterior mode is also assumed to converge to the parameter θ^*. Applying these findings to the earlier KL-divergence formula, we can see that;

$$\lim_{t\to\infty} D_{KL}(p(\theta \mid x_{0:t})||q_t(\theta)) = 0. \tag{52}$$

For the `SIN` model discussed in the experiments section the true posterior $p(\theta \mid x_{0:t})$ is computed for a grid of parameter values in $O(t)$ time per parameter value. Assumed density filtering is also applied with $\mathcal{Q}$ Gaussian and the true density (solid) vs. assumed density (dashed) is illustrated in Figure 10. Notice that, ADF is slightly off at earlier stages, however, does indeed catch up with the ground truth with more data.

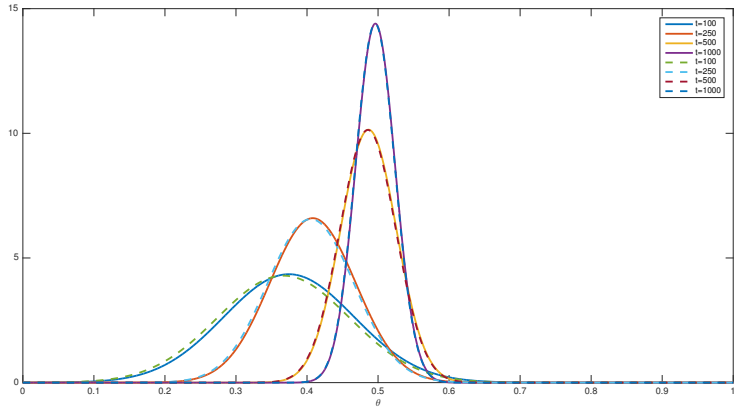
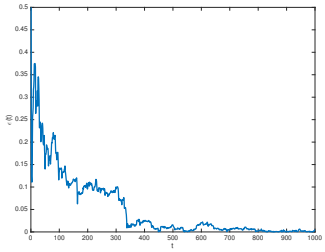
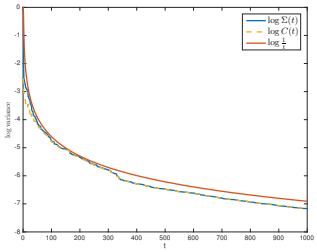


Figure 10: True posterior $p(\theta \mid x_{0:t})$ vs assumed density filter estimate $q_t(\theta)$ (solid vs dashed line respectively). for the `SIN` model.

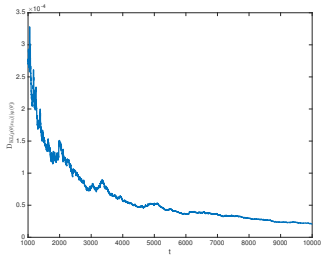
As predicted by Equation 50, the error term converges to zero as shown in Figure 11(a). Figure 11(b) illustrates the asymptotic behavior of the true posterior covariance $C(t)$ and assumed density covariance $\Sigma(t)$. Assumed density filter quickly approximates the covariance.



(a) Error term $\epsilon(t)$



(b) Covariance term $\Sigma(t)$



(c) KL divergence

Figure 11: `SIN` model with $\theta^* = 0.5$

Most importantly, as can be seen from the plot, $\log C(t)$ and $\log \Sigma(t)$ is $log(1/t)+$constant asymptotically, and this agrees with our derivations in Equation 48. Figure 11(c) confirms our theoretical result of KL divergence converging to zero in the long-sequence limit.

Mixture of Gaussians Derivation

Let us assume that at time step $i-1$ it was possible to represent $p(\theta \mid x_{0:i-1}, y_{0:i-1})$ as a mixture of Gaussians with L components.

$$\begin{aligned} p(\theta \mid x_{0:i-1}, y_{0:i-1}) &= \sum_{m=1}^{L} \alpha_{i-1}^m \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m) \\ &= q_{i-1}(\theta) \end{aligned}$$

Given x_i and y_i;

$$\hat{p}(\theta \mid x_{0:i}, y_{0:i}) \propto \sum_{m=1}^{L} \alpha_{i-1}^m s_i(\theta) \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m)$$

The above form will not be a Gaussian mixture for arbitrary s_i. We can rewrite it as:

$$\hat{p} \propto \sum_{m=1}^{L} \alpha_{i-1}^m \beta^m \frac{s_i(\theta) \mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m)}{\beta^m} \tag{53}$$

where the fraction is to be approximated by a Gaussian via moment matching and the weights are to be normalized. Here, each $\beta^m = \int s_i(\theta)\mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m)d\theta$ describes how well the m-th mixture component explains the new data. That is, a mixture component that explains the new data well will get up-weighted and vice versa. The resulting approximated density would be $q_i(\theta) = \sum_{m=1}^K \alpha_i^m \mathcal{N}(\theta; \mu_i^m, \Sigma_i^m)$ where the recursions for updating each term is as follows:

$$\begin{aligned}
\beta^m &= \int s_i(\theta)\mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m)d\theta \\
\alpha_i^m &= \frac{\alpha_{i-1}^m \beta^m}{\sum_\ell \alpha_{i-1}^\ell \beta^\ell} \\
\mu_i^m &= \frac{1}{\beta^m}\int \theta s_i(\theta)\mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m)d\theta \\
\Sigma_i^m &= \frac{1}{\beta^m}\int \theta\theta^T s_i(\theta)\mathcal{N}(\theta; \mu_{i-1}^m, \Sigma_{i-1}^m)d\theta - \mu_i^m(\mu_i^m)^T
\end{aligned}$$

Similar to the Gaussian case, the above integrals are generally intractable. Either a Monte Carlo sum or a Gaussian quadrature rule can be utilized to approximately update the means and covariances.

Compilation Optimizations in SPEC

In this section, we introduce some optimizations in the implementation of SPEC. These techniques are mostly standard techniques from programming languages and adopted by many modern PPL compilers.

Memory Efficiency

Memory Pre-allocation: Memory management is a critical issue for particle filter algorithms since it is often necessary to launch a large number of particles in practice. Systems that does not manage the memory well will repeatedly allocate memory at run time. For example, in BLOG, the inference engine will allocate memory for new particles and erase the memory belonging to the old ones after each iteration. This introduces tremendous overhead since memory allocation is slow.

By contrast, SPEC will analyze the input model and allocate the minimum static memory for computation: if the user specifies to run K particles, and the Markov order of the input model is D, SPEC will allocate static memory for $(D+1) * K$ particles in the target code. When the a new iteration starts, we utilize a rotational array to re-use the memory of previous particles.

Lightweight Memoization: Notice that adopting an expressive language interface (i.e. syntax of BLOG) might lead to time-varying dependencies between random variables. Similar to the compilation of lazy evaluation in programming language community, SPEC memoizes the value for each random variables already sampled. An example compiled code fragment for the `SIN` is shown below.

```
class TParticle { public:
// value of x and y at current timestep
  double val_x, val_y;
// flag of whether x/y is sampled
  int mark_x, mark_y;
```

```
} particles[K][DEP]; // particle objects
// getter function of y(t)
double get_y(int t) {
// get the corresponding particle
  TParticle &part = get_particle(t);
  // memoization for y(t)
  if(part.mark_y == cur_time) return part.val_y;
  part.mark_y = cur_time;
// sample y(t), call getter function of x(t)
  part.val_y = sample_normal(sin(get_theta() * get_x(t-1)), 0.5);
  return part.val_x;
}
```

This code fragment demonstrate the basic data structures as well as the memoization framework in the target code. `K` denotes the number of particles. `DEP` is equal to the Markov order of the model plus 1. In the memoization framework, SPEC allocates static memory for all the random variables within a particle data structure and generates a flag for each random variable denoting whether it has been sampled or not. Each random variable also has its *getter* function. Whenever accessing to a random variable, we just call its getter function. In practice, memoization causes negligible overhead.

Memoization also occurs in BLOG. However, every random variable in BLOG has a string "name", and a generic hashmap is used for value storage and retrieval, which brings a great deal of constant factor at run time.

Other Optimizations: SPEC also avoids dynamic memory allocation as much as possible for intermediate computation step. For example, consider a multinomial distribution. When its parameters change, a straightforward implementation to update the parameters is to re-allocate a chunk of memory storing all the new parameters and pass in to the object of the multinomial distribution. However, this dynamic memory allocation operation is also avoided in SPEC by pre-allocating static memory to store the new parameters.

Computational Efficiency

Pointer References: Resampling step is critical for particle filter algorithms since it requires a large number of data copying operations. Note that a single particle might occupy a large amount of memory in real applications. Directly copying data from the old particles to a new ones induce tremendous overhead.

In SPEC, every particle access to its data via an indirect pointer. As a result, redundant memory copying operations are avoided by only copying the pointers referring to the actual particle objects during resampling Note that each particle might need to store multiple pointers when dealing with models with Markov order larger than 1. An example compiled code is shown below.

```
// indirect pointers to the actual particle
TParticles* part_ptrs[K][DEP];
TParticle& get_particle(int t) {// rotational array
  return *part_ptr[cur_part][t % DEP]; }
```

Locality: Besides using pointer references to reduce the amount of moving data, SPEC also enhances program locality to speed up re-sampling. In the compiled code, the index of the arrays stores the indirect pointers are carefully designed to take advantage of memory locality when copying. The fragment of code used in resampling step is shown below.

```
void resample_ptr( int* target,
  TParticle* part_ptr[K][DEP], //storage of  pointers
  TParticle* backup_ptr[K][DEP]) // temporary storage
{ for (int i = 0; i < K; ++i) {
    // pos: index of particle to copy data from
    int& pos = target_ptr[i];
    // move continuous range of data
    std::memcpy(backup_ptr[i], part_ptr[pos], sizeof(TParticle*)* DEP);  }
  std::memcpy(ptr_temp_memo, backup_ptr, sizeof(TParticle*)* DEP * K);
}
```

In the preceding code fragment, the first dimension of the storage array, `part_ptr`, corresponds to the particle index. While the second dimension corresponds to the current iteration. In this case, when copying the pointers during resampling step, all the memory are continuously located in the memory space, which reduces the constant factor at run time.

5 Model Based Probabilistic Inference for Intensive Care Medicine

Due to advances in electronics, modern ICUs are capable of collecting and archiving ample amounts of clinical data. An ICU patient is continuously monitored by various sensors as asynchronous interventions, tests and additional measurements are carried out. While data are valuable, *understanding* and *acting* upon them is what provides benefits in terms of improved health outcomes and reduced costs.

A patient in an ICU is continuously monitored by various sensors, while many interventions, tests, and additional measurements are done asynchronously. The sensory data are usually displayed in real time, and can, if continuously observed, provide an expert physician with a great deal of insight into a patient's condition. In most ICU settings, however, data are reduced to an hourly paper chart generated by a nurse and reviewed daily by the physician. It seems plausible, therefore, that standards of care might be improved by automated data analysis and decision support systems.

Current monitoring technology is largely based on data display. The monitor displays signals from various sensors and provides statistics such as short-term averages. Some monitors also deploy automated alarms based on either simple threshold based rules or single-channel analysis. However, false alarm rates of these monitors often exceed 90%, leading to *alarm fatigue* Drew et al. [2014], Chopra and McMahon [2014], Tsien and Fackler [1997].

An ideal system should not only report all pertinent measurements to clinicians, but should also summarize information by *inferring* the states of various *latent* physiological and pathophysiological variables [Heldt et al., 2006, Heldt and Verghese, 2010, Heldt et al., 2013]. We propose a model-based probabilistic inference framework that can handle artifact-ridden data, variability in patient physiology and unknown disease states.

We describe the patient's physiology and the sensor dynamics as a probabilistic model using a dynamic Bayesian network [Erol et al., 2013b, Sivaganesan et al., 2012, Erol et al., 2015]. We use existing works on human physiology to describe how the state of the patient evolves over time and employ a nontrivial sensor model that is capable of explaining various artifactual readings in the ICU setting. Then, the main task of the ICU monitoring system is estimating the state of the patient accurately given a sequence of observations. Due to the nonlinear, non-Gaussian behavior of our model, exact inference is intractable. Hence, we resort to approximate inference via sequential Monte Carlo (SMC) methods.

In section 5.1, we describe the model-based probabilistic inference approach and list motivating reasons for such an approach. Our main interest is the intracranial hemodynamics of traumatic brain injury (TBI) patients. We briefly go over the physiology and sensor models and also present some inference results. We conclude by discussing the required extensions to the proposed models and the limitations of the current approach.

5.1 Model-Based Probabilistic Inference

Due to the uncertainty in our knowledge of patient physiology, and the partial, noisy/artifactual nature of the observations, we adopt a probabilistic, model-based approach. As mentioned earlier, the core task of the ICU monitoring system then becomes estimating the state of a patient given a sequence of observations by computing a posterior probability distribution. Recent trends in machine learning suggest adopting a purely data-driven, model-free approach. However, this may not be the best fit for the intensive care domain due to the following reasons: 1) The data is complex and artifact-ridden. Data may be missing from some sensors for extended periods and there are many artifactual readings some persisting over extended periods. 2) There is a disparity between the number of available signals and the dimensionality of the state space. We only have access to few measurements whereas the latent state-space is high-dimensional. 3) Models of the underlying physiology are available and sensor dynamics can be explained. By being model-free, the available knowledge and information is not exploited.

5.1.1 Probabilistic Modeling

Most physiological models are typically expressed as *determinisitic* differential equations. A classic example is the Guyton model which aims to describe the whole human circulation system in terms of various interconnected subsystems Guyton et al. [1972]. Deterministic modeling is not a good fit as patient-specific parametrization is unknown and many pathophysiological states manifest stochastic behavior.

We use two different models: The *transition (physiology) model*, $p(\text{states}_t \mid \text{states}_{t-1}, \theta)$, (where states_t stands for the physiological states at time step t and θ for the patient-specific parameters), describes how the state of the patient evolves with respect to time. The variable state_t may include physiological, pathophysiological states, sensor artifacts and failure states, drug administration and so on. Hence, this approach can be extended to handle various possible scenarios of interest in an ICU setting. The *sensor model*, $p(\text{observations}_t \mid \text{states}_t)$, describes how a patient and sensor state are related to the observations.

We represent the model using the dynamic Bayesian network (DBN) framework. DBNs are concise descriptions of stochastic differential equations and they can handle both discrete and continuous variables Murphy [2002].

The core task of ICU monitoring is calculating a posterior probability distribution over states given an observation history, $p(\text{states}_t \mid \text{observations}_{0:t})$. Our model is nonlinear, non-Gaussian, and hybrid (containing discrete and continuous state variables). Therefore to compute the required posterior densities, we resort to sequential Monte Carlo (SMC) algorithms discussed in this thesis.

Every human has a unique set of parameters. Learning these parameter values, $p(\theta \mid \text{observations}_{0:t})$, is a crucial part of the proposed framework; failing to do so will lead to inaccurate state estimates. Therefore, we do joint state and

parameter estimation in our temporal model as discussed in the earlier sections of this thesis.

5.2 Application: Intracranial Hemodynamics

Our particular focus is on neurocritical care for traumatic brain injury (TBI), which is the developed world's leading cause of morbidity and mortality for individuals under the age of 45 Werner and Engelhard [2007]. TBI patients may exhibit a multitude of primary and secondary brain injury pathways, and their obtunded state adds to the difficulty of clinical assessment. The physiological mechanism of interest for such patients is the intracranial hemodynamics system, for which the key variable is the intracranial pressure (ICP)—the pressure of the cerebrospinal fluid that surrounds the brain tissue. We are interested in inferring critical physiological events and states using from ICP data and other measurements.

For most purposes, available measurements include intracranial pressure (ICP), arterial blood pressure (ABP), blood oxygen level and in some cases cerebral blood flow velocity (CBFV). The problem is depicted in Figure 12. Our goal is calculating the posterior probability on the latent variables given the observed variables (shaded nodes). We have three main subsystems: 1)

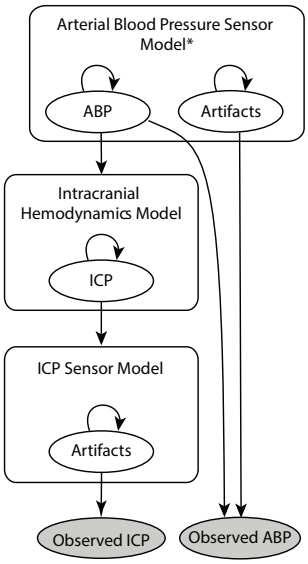


Figure 12: High-level DBN representation of intracranial hemodynamics

Intracranial hemodynamics model 2) ABP sensor model 3) ICP sensor model. Our sensor model is capable of explaining major clinical artifacts like blood draw, zeroing, line clog, and drainage.

Although not shown in Figure 12, transcranial Doppler measurements of CBFV Steiner and Andrews [2006] may be available as well. Under these

circumstances, noninvasive estimation of ICP via the proposed model-based probabilistic approach using noninvasive measurements like ABP and CBFV is possible. There are recent works in the literature that rely on a model-based approach to achieve the noninvasive reconstruction goal Kashif et al. [2012], Kashif [2011].

5.3 Probabilistic Model of Intracranial Hemodynamics

The brain is enclosed inside the rigid skull and bathed by the cerebrospinal fluid (CSF). The volume of the intracranial compartment consists of three components: the brain tissue, cerebral vasculature, and CSF compartments. Since the intracranial compartment is a closed system, an increase in one component must lead to a decrease in one or both of the other two. This is known as the Monro-Kellie doctrine [Mokri, 2001].

Cerebral blood flow is proportional to the so called cerebral perfusion pressure (CPP) which is the gradient between mean arterial pressure and ICP. The brain demands a continuous and substantial supply of blood flow. In order to deal with fluctuations in CPP our bodies utilize cerebral autoregulation, which is the mechanism that regulates the cerebral blood flow to meet the strict constraint of brains blood supply needs. For many TBI patients, autoregulation is no longer intact therefore the blood oxygen supply to the brain is highly dependent on maintaining CPP at reasonable levels. One of the main goals of our proposed framework is to estimate the unobserved autoregulation state of the patient via streaming continuous measurements.

Latent true mean ABP value is an exogenous input to the intracranial hemodynamics model. We are adopting the intracranial dynamics model developed by Ursino and Lodi [1997], Ursino et al. [1997]. This model has been extended to include various pathophysiological states such as autoregulation failure, hematoma, edema, internal bleeding, vasospasm etc. Some results on state and parameter estimation are presented in Erol et al. [2013b], Sivaganesan et al. [2012].

Ursino model consists of bunch of nonlinear differential equations that regulate pressure and flow in the intracranial compartment. We discretize these differential equations, introduce stochasticity to explain small random perturbations and randomly occurring pathophysiologies and represent the intracranial dynamics using a dynamic Bayesian network (DBN). A portion of the model is illustrated in 13.

We have developed differential equations for various sets of physiological variables. For instance, ICP can be expressed mathematically as follows.

$$\begin{aligned}\dot{P}_{ic} &= \left(\frac{2G_1}{C_{ic}}\right)P_a + \left(\frac{-2G_1}{C_{ic}}\right)P_1 + \left(\frac{-2G_2}{C_{ic}}\right)P_2 + \left(\frac{2G_2+G_{pv}}{C_{ic}}\right)P_c \\ &+ \left(\frac{-G_{pv}-G_{vs}}{C_{ic}}\right)P_v + \left(\frac{G_{vs}}{C_{ic}}\right)P_{vs} + \left(\frac{G_f}{C_{ic}}\right)(P_c - P_{ic})_+ - \left(\frac{G_0}{C_{ic}}\right)(P_{ic} - P_{vs})_+\end{aligned}$$

where C_1, C_2, G_f, and G_0 are static parameters and P_a, P_1, P_2, P_c, P_v and P_{vs}

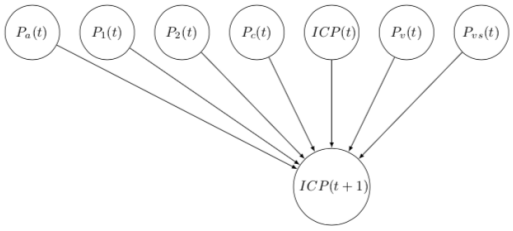


Figure 13: A portion of the DBN that illustrates the transition for the intracranial pressure $ICP(t)$

are various pressures inside the intracranial compartment.

$$\begin{aligned} C_{ic} &= \frac{1}{(K_E \times P_{ic})} \\ C_{vi} &= \frac{1}{K_v(P_v - P_{ic} - P_{v1})} \end{aligned}$$

K_E, K_v and P_{v1} are static parameters as well. The complete model has more than 10 dynamic state variables and more than 25 static parameters.

Figure 13 illustrates the aforementioned differential equation graphically. Due to the highly interconnected nature of the DBN structure, we will not illustrate the whole intracranial dynamics structure here.

Our model is capable of explaining various stochastic physiological and clinical scenarios. We have the capability to simulate pathophysiologies like hematoma, edema, hypo/hyper-tension, and hyperventilation. Furthermore, our model can simulate the dynamics of clinical interventions like jugular vein compression and cerebrospinal fluid (CSF) drainage that is performed frequently at neurocritical ICUs to assess the patient's physiological state.

Figure 14, 15, and 16 illustrate pathophysiology simulations and show how our model can express events like hypotension, hyperventilation, and hematoma respectively. Simulations agree with medical knowledge and brief explanations to the underlying mechanism for each scenario are given in the caption.

Figure 17 and 18 depict clinical intervention simulations and show how our model can express events like CSF drainage and jugular vein compression respectively.

Pathophysiology and clinical intervention variables are described as Bernoulli processes in our DBN. These variables can switch from normal to abnormal state with some probability. Given observations, our inference algorithms have the capability to infer whether pathophysiologies are happening or the nature of interventions.

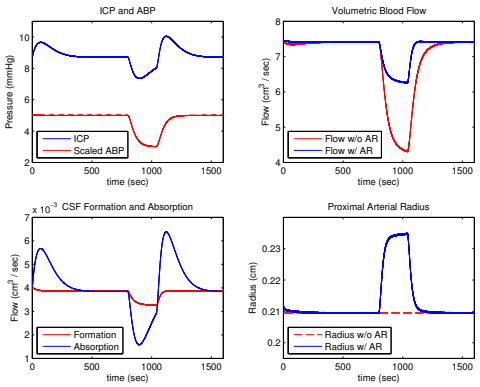


Figure 14: **Hypotension**: The simulated behavior of intracranial variables around a 4 min period in which the ABP drops from 100 to 60 mmHg. Top left: ICP falls and rises with ABP. Top right: CBF stays within a tight range when autoregulation is intact, but decreases markedly when it is impaired. Bottom left: CSF absorption drops below formation as the reduction in ICP decreases its pressure gradient. Bottom right: Proximal arterial radius increases during hypotension, but only if autoregulation is intact.

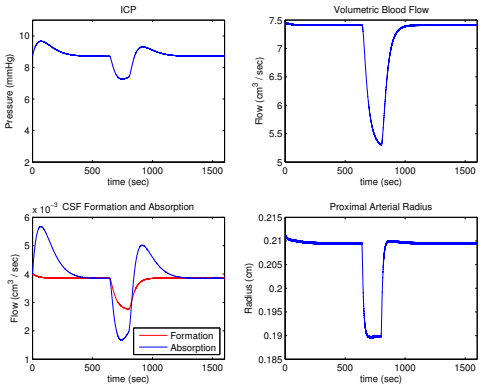


Figure 15: **Hyperventilation**: CO_2 levels decrease. Blood flow is restricted via autoregulation to keep oxygen levels in range. Top left: Less blood flows in and hence ICP drops. Top right: CBF decreases to keep O_2 and CO_2 levels in check. Bottom right: Proximal artery constricts to limit cerebral blood flow.

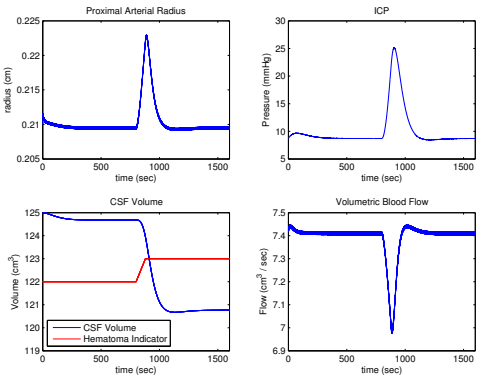


Figure 16: **Hematoma**: Extraneous volume starts growing hence replacing other volumes. Top left: Radius increases in order to maintain steady flow due to increased ICP. Top right: ICP rises as there is an additional volume source in the intracranial compartment. Bottom left: CSF gets absorbed to open up space to hematoma. Bottom right: CBF decreases then recovers due to autoregulation.

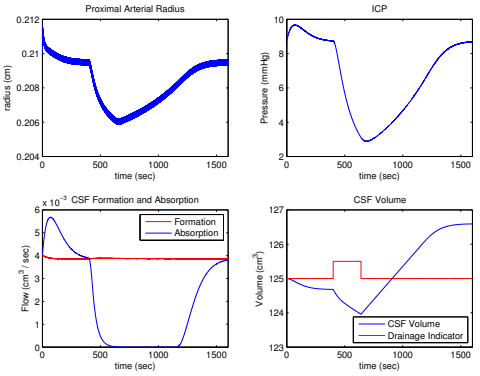


Figure 17: **CSF drainage**: CSF is drained at a constant rate for a few seconds. Top left: CPP increases and in order to maintain blood flow proximal artery constricts. Top right: ICP falls as there is less volume in the compartment. Bottom left: CSF absorption ceases.

5.3.1 Inference in Intracranial Hemodynamics Model

Minutes' worth of intracranial pressure data is simulated via our model at 125 Hz. We first investigate inferring autoregulation failure. Autoregulation state is one of the most important variables for physician to infer for TBI patients. Usually a physician might resort to an invasive clinical intervention to observe the reaction and assess accordingly. With particle filtering, we can infer via leveraging the information buried in high-frequency data. Figure 19 illustrates

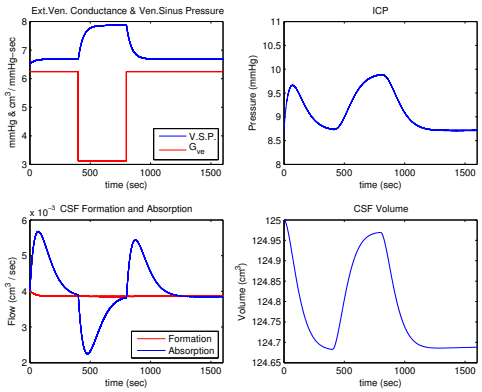


Figure 18: **Jugular vein compression**: Simulated behavior during jugular compression. Top left: Venous sinus pressure (VSP) rises as jugular vein is compressed. Top right: ICP rises as jugular compression prevents venous outflow. Bottom left: CSF absorption rate drops below formation rate as rise in VSP decreases its pressure gradient. Bottom right: CSF volume rises while CSF formation exceeds absorption.

the inference of autoregulation failure. Within several seconds of latency our bootstrap filter with 5000 particles infers that a failure has happened. For this specific experiment we assumed that all parameters were known and set to their demographic means provided by Ursino et al. [1997].

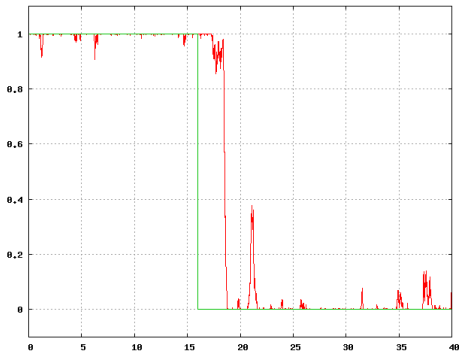
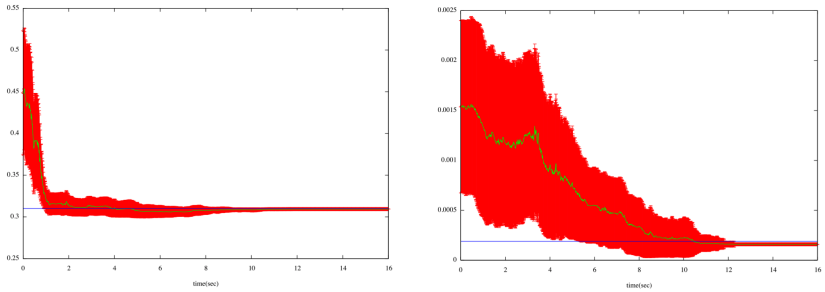


Figure 19: Green line shows the exact autoregulation behavior. Autoregulation failure happens around the 15th second whereas the particle filter infers the failure correctly around the 20th second. Red line is the mean of the particles, showing approximately the probability of whether autoregulation is intact.

For the following joint state and parameter estimation experiment, we assume all the parameters are known and set to their demographic means except the intracranial compartment elasticity constant k_E and CSF outflow conduc-

tance G_0. The Liu-West filter with 5000 particles lead to reasonable posteriors that converge to the true parameter values as depicted in Figure 20(a) and 20(b).



(a) k_E Inference (b) G_0 Inference

Figure 20: State and Parameter Estimation. Blue horizontal line depicts the true parameter setting. (a)-Particle mean (green) and std (red) for elasticity constant k_E. (b)-Particle mean and std for CSF outflow conductance G_0.

5.4 Sensor Model

5.4.1 ABP Sensor Model

As stated by Aleks et al. [2009], blood pressure informs much of medical thinking and is typically measured continuously in the ICU. The most common way of determining blood pressure in the ICU is to place a transducer on an arterial line, a catheter that is inside one of the patients small arteries; this data is then displayed on a bedside monitor. Due to the high variance of pressure during the cardiac cycle, we use three measurements: systolic (the maximum reached during the cardiac cycle), diastolic (the corresponding minimum), and mean blood pressures [Aleks et al., 2009].

In [Aleks et al., 2009], using the model-based probabilistic inference methodology, over 90% of false ABP alarms (threshold-based) were eliminated while only missing fewer than 1% of the true alarms. The aforementioned work used a minute-resolution DBN. We reproduced their results and then used this model to create a second-resolution blood pressure model. As we predicted, the second-resolution model is able to capture events that the minute-resolution model was unable to capture. For example, most events that lasted less than 10 seconds are not detected by the minute-model, but are picked up by the second-resolution model.

So far, we modeled three major ABP sensor artifacts: zero events, bag events, and line clogging. Zero events occur when an ICU staff calibrates the instrumentation device, effectively zeroing out all the signals. It is seen as a severe and sudden drop in all of the values due to the transducer reading atmospheric pressure. Bag events occur when the transducer reads the intravenous (IV) bag pressure instead of the patient's blood pressure and is seen as a dramatic spike

in the observed values. Finally, line clogs occur when there is a kink or an obstruction in the line to the transducer. This zeros the variation in pressure, causing the systolic and diastolic to converge to the mean value.

Figure 21 shows real data from an ICU patient over the span of 100 minutes. From top to bottom, one can see the observed systolic, mean, and diastolic blood pressures. Just by looking at data, it is obvious that handling the artifacts and the missing data is crucial for the success of a real-time decision support system.

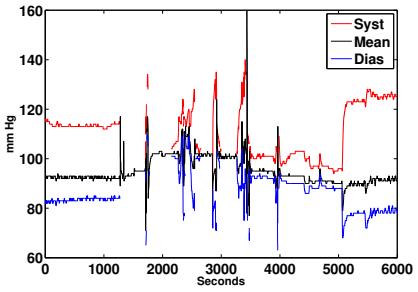


Figure 21: Sample artifactual ABP trace

5.4.2 Model

We developed a one second-resolution model that is capable of generating artifacts and physiological signals. The DBN representation of the developed generative model is depicted in Figure 22.

At the top of the DBN are the true pulse, true mean, and systolic fraction (the ratio of the difference of the diastolic and systolic pressures and the mean). These values give us the true systolic and diastolic values, the ground truth values that our algorithm attempts to infer. We simply used a random-walk type model to describe the evolution of these values. A better cardiovascular dynamics model will significantly improve the performance of the proposed approach.

The values near the bottom are the apparent systolic, diastolic, and mean blood pressure– values that include artifacts and are similar to what the transducer would be measuring. The observed values just below include some additional signal noise and represent what would typically be seen on the monitor.

At any given time, a patient can be in a normal, zero, bag, or clog state. Depending on which state the patient is in, *newPot[S,D,M]* will reflect the new potential that each apparent pressure will converge to. A probabilistic decision is then made on whether patients stays in their current state. Finally, we propagate to the next time step from the current values using our mathematical models. This can be seen in Figure 22 as the arrows point from time step t to $t+1$.

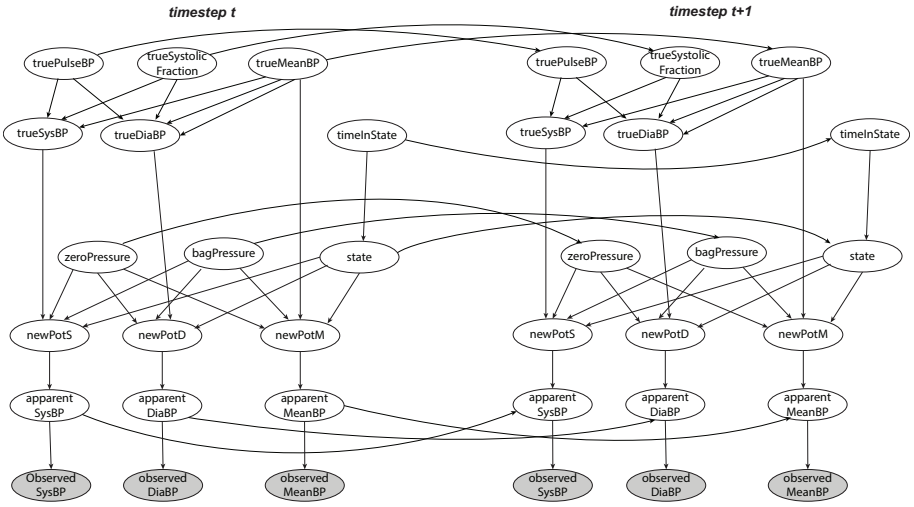
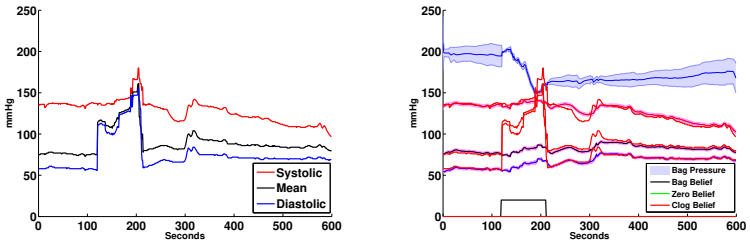


Figure 22: Dynamic Bayesian network representation of the arterial blood pressure sensor model

Bag Events

One artifact is the bag event, which occurs when the transducer reads the positive pressure from the IV bag elevated at a height. Figure 23(a) shows what the event would look like on an ICU monitor.



(a) Data (b) Inference

Figure 23: (a) Sample blood draw trace; (b) Probabilistic belief on systolic, mean, diastolic, and bag latent variables

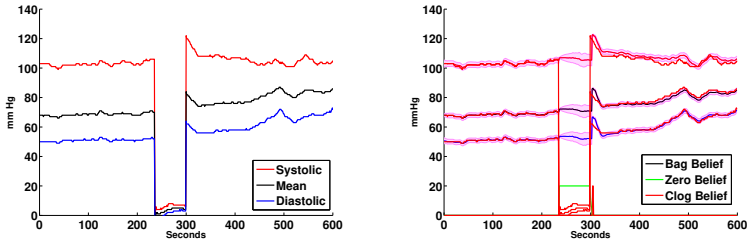
The inference results using a particle filter with $N = 20000$ particles is illustrated in Figure 23(b). The three blue lines represent the mean belief of the diastolic, mean, and systolic ABP values, which do not follow the spike of the bag event. The lines at the bottom report the beliefs for each event occurring; in this case, only the bag event registers. The purple shade around the blue lines describe the uncertainty in our estimates.

It is also interesting to see how the posterior density over the bag pressure

evolves. The posterior density starts at the prior and gets more and more ambiguous over time. Once the bag event is inferred, the bag value is quickly updated and the particle filter is almost certain of the bag pressure value. This behavior is also theoretically expected.

Zero Events

The second artifact incorporated into our DBN is a zeroing event. As mentioned, this occurs when the transducer is exposed to atmospheric pressure which is approximately zero pressure. Figure 24(a) shows an example of such an event.

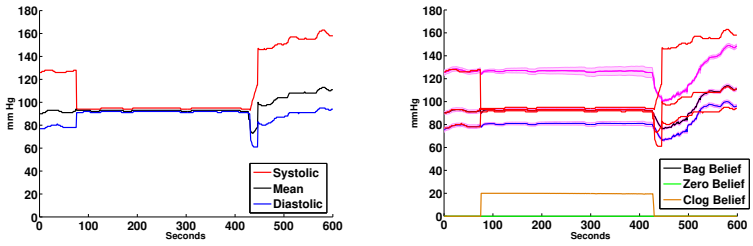


(a) Data (b) Inference

Figure 24: (a) Sample zeroing trace; (b) Probabilistic belief on systolic, mean, diastolic, and zeroing latent variables

This event is similar to a bag event in that the sensor no longer reads patient data. A working inference algorithm should detect this event and show that the true diastolic, mean, and systolic values remain in a safe range. Figure 24(b) shows how the inference performs on this event. The blue lines again represent mean belief values for systolic, diastolic, and mean blood pressures in our inference model. We can see that the zero event is detected and the values reflect that. Note that the purple shade gets wider during the event. This is due to the fact that the inference no longer trusts the current observations and the beliefs on the ground truth variables get more and more uncertain with time.

Clog Events



(a) Data (b) Inference

Figure 25: (a) Sample line clog trace; (b) Probabilistic belief on systolic, mean, diastolic, and line clog latent variables

The last ABP artifact modeled is a line clog, which causes loss of pulse-to-

pulse pressure. As seen in Figure 25(a), the observed values for systolic and diastolic blood pressure converge to the mean for the entirety of the event.

The inference results are depicted in Figure 25(b) . The particles are capable of tracking the clogging artifact as well as interpolating the missing systolic and diastolic values.

5.4.3 ICP Sensor Model

For the neurocritical care of TBI patients, intracranial pressure (ICP) is the most important measurement for diagnosis and treatment. Current treatment procedure keeps the cerebral perfusion pressure (CPP), which is defined as the gradient between mean arterial pressure (MAP) and ICP, in a safe range in order to keep the patients brain supplied with oxygen. The simplest treatment strategy to keep ICP below a certain level is cerebrospinal fluid (CSF) drainage. CSF is drained when ICP exceeds some set threshold (usually 20 or 25 mmHg). During the drainage, the ICP sensor reads a random pressure value. A sample drainage trace is illustrated in Figure 26.

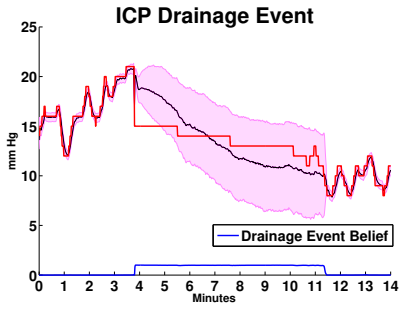


Figure 26: Probabilistic belief on intracranial pressure during CSF drainage

Unlike blood pressure artifacts, drainage also affects the physiology. During the drainage, the patient's true ICP slowly falls due to the volume loss. The particle filter, after noticing the sharp drop from 22 mmHg to 14 mmHg, immediately detects a drainage event since the intracranial physiology model is incapable of explaining such an instantaneous change. It is important to notice that the intracranial pressure belief is nicely estimated in a physiologically reasonable way during the drainage event.

5.5 Concluding Remarks

We described a model-based probabilistic framework capable of representing highly complex physiological phenomena. Using state-of-the-art statistical learning algorithms, we are able to do combined state and parameter estimation. The proposed approach can be used for estimating physiological and pathophysiological states, sensor artifacts and failure states, and drug administration.

The sensor models are still quite inadequate; various other artifacts still need to be added in order to handle real-life clinical data. Artifacts we are considering to add to our DBN in the immediate future are: line flushes, sensor detachment, patient coughing or thrashing, and a nurse rolling the patient. Although our preliminary results seem promising, we still need to validate the artifact cleaning approach on more real data.

The current physiology model we are using is also fairly restrictive as it doesn't describe various pathophysiological phenomena. We need to extend the model provided by Ursino to handle different disease states. Furthermore, we currently do not have a pharmacokinetics model that can explain the dynamics after drug administration. Drugs such as mannitol are frequently used in the clinical care of TBI patients Rangel-Castillo et al. [2008]. We would next want to extend our generative model to describe the effects of mannitol administration to infer the onset of drug administration as well as the dosage.

We also still need to validate the performance of the pathophysiological state estimation by comparing inferential results of our algorithms on real data against the physician's diagnosis after a provocative test or intervention.

6 Conclusion

Many sequential Monte Carlo algorithms have been introduced for joint state and parameter estimation in the literature, however, the task still remains a challenge for state space models with complex dependencies and nonlinear dynamics. Existing algorithms are either restricted to a particular class of models or computationally expensive. We proposed two new algorithms, namely, the extended parameter filter and the assumed parameter filter that try to close the trade-off gap between computational efficiency, accuracy, and applicability to arbitrary model classes.

Our first contribution, the extended parameter filter extends the Storvik filter in a natural way by introducing *separability*. EPF further utilizes polynomial approximation schemes to handle much broader model classes. Although theoretically very appealing, the need for manually deriving polynomial approximations constitute a substantial problem for a fully automatic algorithm that can be utilized in a probabilistic programming language.

Our second contribution, the assumed parameter filter is a nearly-black box algorithm applicable to arbitrary temporal models. We have developed it as an automatic inference engine for a probabilistic programming engine. Experiments show that APF converges much faster than existing algorithms on a variety of models. APF is a very promising direction for online joint state and parameter estimation. One avenue for future work is analyzing the theoretical conditions under which APF can approximate the posterior robustly for arbitrary models. Automating the choice of approximation densities will also be beneficial for making inference algorithms and probabilistic programming more accessible to the non-expert.

Finally, we discussed our work on model-based probabilistic inference for intensive care medicine. We have shown some results on simulated data for intracranial hemodynamics model. For sensor artifact clearing, several results have been shown on real clinical blood pressure data.

In order to make the approach work with real clinical data, one needs to model, infer, analyze/visualize results, and re-model continually. Probabilistic programming is invaluable for such a task as it decouples the task of modeling and inference. In future, by developing stronger and better black-box inference algorithms for joint state and parameter estimation we should be able to adequately handle complex models and speed up the research process.

References

Norm Aleks, Stuart J Russell, Michael G Madden, Diane Morabito, Kristan Staudenmayer, Mitchell Cohen, and Geoffrey T Manley. Probabilistic detection of short events, with application to critical care monitoring. In *Advances in Neural Information Processing Systems*, pages 49–56, 2009.

Daniel L Alspach and Harold W Sorenson. Nonlinear Bayesian estimation using Gaussian sum approximations. *Automatic Control, IEEE Transactions on*, 17(4):439–448, 1972.

Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B*, 72(3):269–342, 2010.

Nimar S Arora. *Model-based Bayesian Seismic Monitoring*. PhD thesis, EECS Department, University of California, Berkeley, May 2012. URL `http://www2.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-125.html`.

Nimar S. Arora, Stuart J. Russell, Paul Kidwell, and Erik B. Sudderth. Global seismic monitoring as probabilistic inference. In *NIPS*, pages 73–81, 2010.

Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

Xavier Boyen and Daphne Koller. Tractable inference for complex stochastic processes. In *UAI*, pages 33–42. Morgan Kaufmann Publishers Inc., 1998.

Olivier Cappé, Simon J Godsill, and Eric Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.

Carlos M. Carvalho, Michael S. Johannes, Hedibert F. Lopes, and Nicholas G. Polson. Particle Learning and Smoothing. *Statistical Science*, 25:88–106, 2010. doi: 10.1214/10-STS325.

N. Chopin, P. E. Jacob, and O. Papaspiliopoulos. SMC^2: An efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society: Series B*, 75(3):397–426, 2013. ISSN 1467-9868. doi: 10.1111/j.1467-9868.2012.01046.x. URL `http://dx.doi.org/10.1111/j.1467-9868.2012.01046.x`.

Nicolas Chopin, Alessandra Iacobucci, Jean-Michel Marin, Kerrie Mengersen, Christian P Robert, Robin Ryder, and Christian Schäfer. On particle learning. *arXiv preprint arXiv:1006.0554*, 2010.

Vineet Chopra and Laurence F McMahon. Redesigning hospital alarms for patient safety: Alarmed and potentially dangerous. *JAMA*, 311(12):1199–1200, 2014.

Dan Crisan and Arnaud Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on signal processing*, 50(3):736–746, 2002.

Pierre Del Moral, Arnaud Doucet, and Sumeetpal Singh. Forward smoothing using sequential Monte Carlo. *arXiv preprint arXiv:1012.5390*, 2010.

Arnaud Doucet and Adam M. Johansen. A tutorial on particle filtering and smoothing: fifteen years later. *The Oxford Handbook of Nonlinear Filtering*, pages 4–6, December 2011.

Barbara J Drew, Patricia Harris, Jessica K Zègre-Hemsey, Tina Mammone, Daniel Schindler, Rebeca Salas-Boni, Yong Bai, Adelita Tinoco, Quan Ding, and Xiao Hu. Insights into the problem of alarm fatigue with physiologic monitor devices: A comprehensive observational study of consecutive intensive care unit patients. *PloS one*, 9(10):e110274, 2014.

MaS Elmohamed, Dexter Kozen, and Daniel R Sheldon. Collective inference on Markov models for modeling bird migration. In *Advances in Neural Information Processing Systems*, pages 1321–1328, 2007.

Yusuf B Erol, Lei Li, Bharath Ramsundar, and Russell Stuart. The extended parameter filter. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1103–1111, 2013a.

Yusuf B Erol, Stuart J Russell, Ahilan Sivaganesan, and Geoffrey T Manley. Combined state and parameter estimation of human intracranial hemodynamics. *NIPS-13 Workshop on Machine Learning for Clinical Data Analysis and Healthcare*, 2013b.

Yusuf B Erol, Romi Phadte, Harsimran Sammy Sidhu, Claire Asselstine, David Phillips, Geoffrey Manley, and Stuart Russell. Model based probabilistic inference for intensive care medicine. *Meaningful use of complex medical data*, 2015.

Yusuf Bugra Erol, Yi Wu, Lei Li, and Stuart J Russell. A nearly-black-box online algorithm for joint parameter and state estimation in temporal models. In *AAAI*, pages 1861–1869, 2017.

Walter R. Gilks and Carlo Berzuini. Following a moving target – Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 63(1):127–146, 2001.

Noah D. Goodman, Vikash K. Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: A language for generative models. In *UAI*, pages 220–229, 2008.

N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings F Radar and Signal Processing*, 140(2):107–113, 1993. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=210672`.

Arthur C Guyton, Thomas G Coleman, and Harris J Granger. Circulation: overall regulation. *Annual review of physiology*, 34(1):13–44, 1972.

Thomas Heldt and George C Verghese. Model-based data integration in clinical environments. In *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, pages 5209–5212. IEEE, 2010.

Thomas Heldt, Bill Long, George C Verghese, Peter Szolovits, and Roger G Mark. Integrating data, models, and reasoning in critical care. In *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*, pages 350–353. IEEE, 2006.

Thomas Heldt, George C Verghese, and Roger G Mark. Mathematical modeling of physiological systems. In *Mathematical Modeling and Validation in Physiology*, pages 21–41. Springer, 2013.

John R Hershey and Peder A Olsen. Approximating the Kullback Leibler divergence between Gaussian mixture models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–317. IEEE, 2007.

Marco F Huber and Uwe D Hanebeck. Gaussian filter based on deterministic sampling for high quality nonlinear estimation. In *Proceedings of the 17th IFAC World Congress (IFAC 2008)*, volume 17, 2008.

Simon J. Julier and Jeffrey K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

Rudolf E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME – Journal of Basic Engineering*, 82 (Series D): 35–45, 1960.

Nicholas Kantas, Arnaud Doucet, Sumeetpal Sindhu Singh, and Jan Maciejowski. An overview of sequential Monte Carlo methods for parameter estimation in general state-space models. In *15th IFAC Symposium on System Identification*, volume 15, pages 774–785, 2009.

Nikolas Kantas, Arnaud Doucet, Sumeetpal S Singh, Jan Maciejowski, Nicolas Chopin, et al. On particle methods for parameter estimation in state-space models. *Statistical science*, 30(3):328–351, 2015.

Faisal M Kashif, George C Verghese, Vera Novak, Marek Czosnyka, and Thomas Heldt. Model-based noninvasive estimation of intracranial pressure from cerebral blood flow velocity and arterial pressure. *Science translational medicine*, 4(129):129ra44–129ra44, 2012.

Faisal Mahmood Kashif. *Modeling and estimation for non-invasive monitoring of intracranial pressure and cerebrovascular autoregulation.* PhD thesis, Massachusetts Institute of Technology, 2011.

Tejas D Kulkarni, Pushmeet Kohli, Joshua B Tenenbaum, and Vikash Mansinghka. Picture: A probabilistic programming language for scene perception. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4390–4399, 2015.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350 (6266):1332–1338, 2015.

Steffen L Lauritzen. Propagation of probabilities, means, and variances in mixed graphical association models. *Journal of the American Statistical Association*, 87(420):1098–1108, 1992.

Lei Li and Stuart J. Russell. The BLOG language reference. Technical Report UCB/EECS-2013-51, EECS Department, University of California, Berkeley, May 2013. URL `http://www.eecs.berkeley.edu/Pubs/TechRpts/2013/EECS-2013-51.html`.

Fredrik Lindsten, Michael I Jordan, and Thomas B Schön. Particle Gibbs with ancestor sampling. *The Journal of Machine Learning Research*, 15(1):2145–2184, 2014.

Jane Liu and Mike West. Combined parameter and state estimation in simulation-based filtering. In *Sequential Monte Carlo Methods in Practice*. 2001.

Hedibert F Lopes, Carlos M Carvalho, Michael Johannes, and Nicholas G Polson. Particle learning for sequential Bayesian computation. *Bayesian Statistics*, 9:175–96, 2010.

Peter S Maybeck. *Stochastic models, estimation, and control*, volume 3. Academic press, 1982.

Brian Milch, Bhaskara Marthi, Stuart Russell, David Sontag, Daniel L. Ong, and Andrey Kolobov. Blog: Probabilistic models with unknown objects. In *IJCAI*, pages 1352–1359, 2005.

Bahram Mokri. The monro–kellie hypothesis applications in csf volume depletion. *Neurology*, 56(12):1746–1748, 2001.

Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *AAAI*, pages 593–598, 2002.

Kevin P Murphy et al. Bayesian map learning in dynamic environments. In *NIPS*, pages 1015–1021, 1999.

Kevin Patrick Murphy. *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

Radford M. Neal. Slice sampling. *Annals of Statistics*, 31(3):705–767, 2003.

Jimmy Olsson, Olivier Cappé, Randal Douc, Eric Moulines, et al. Sequential Monte Carlo smoothing with application to parameter estimation in nonlinear state space models. *Bernoulli*, 14(1):155–179, 2008.

Manfred Opper and Ole Winther. A Bayesian approach to on-line learning. *On-line Learning in Neural Networks, ed. D. Saad*, pages 363–378, 1998.

Nicholas G. Polson, Jonathan R. Stroud, and Peter Müller. Practical filtering with sequential parameter learning. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(2):413–428, 2008.

George Poyiadjis, Arnaud Doucet, and Sumeetpal Sindhu Singh. Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):pp. 65–80, 2011. ISSN 00063444. URL `http://www.jstor.org/stable/29777165`.

Leonardo Rangel-Castillo, Shankar Gopinath, and Claudia S Robertson. Management of intracranial hypertension. *Neurologic clinics*, 26(2):521–541, 2008.

Maria Paula Rios and Hedibert Freitas Lopes. The extended Liu and West filter: Parameter learning in markov switching stochastic volatility models. In *State-Space Models*, pages 23–61. Springer, 2013.

Branko Ristic, Sanjeev Arulampalam, and Neil James Gordon. *Beyond the Kalman filter: Particle filters for tracking applications*. Artech House, 2004.

Daniel Ritchie, B. Mildenhall, N. D. Goodman, and P. Hanrahan. Controlling procedural modeling programs with stochastically-ordered sequential Monte Carlo. In *SIGGRAPH*, 2015.

Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

Matthias Seeger. Expectation propagation for exponential families. Technical report, 2005.

Ahilan Sivaganesan, Yusuf Erol, Geoffrey T Manley, and Stuart Russell. Modeling and machine learning of cerebrovascular dynamics: A framework for monitoring unmeasurable patient variables. *Neurosurgery*, 71(2):E559, 2012.

LA Steiner and PJD Andrews. Monitoring the injured brain: Icp and cbf. *British journal of anaesthesia*, 97(1):26–38, 2006.

Geir Storvik. Particle filters for state-space models with the presence of unknown static parameters. *IEEE Transactions on Signal Processing*, 50(2):281–289, 2002.

Christine L Tsien and James C Fackler. Poor prognosis for existing monitors in the intensive care unit. *Critical care medicine*, 25(4):614–619, 1997.

M Ursino, CA Lodi, S Rossi, and N Stocchetti. Intracranial pressure dynamics in patients with acute brain damage. *Journal of Applied Physiology*, 82(4): 1270–1282, 1997.

Mauro Ursino and Carlo Alberto Lodi. A simple mathematical model of the interaction between intracranial pressure and cerebral hemodynamics. *Journal of Applied Physiology*, 82(4):1256–1269, 1997.

Dick van Dijk, Timo Tersvirta, and Philip Hans Franses. Smooth transition autoregressive models – a survey of recent developments. *Econometric Reviews*, 21:1–47, 2002.

Greg Welch and Gary Bishop. An introduction to the Kalman filter, 1995.

C Werner and K Engelhard. Pathophysiology of traumatic brain injury. *British journal of anaesthesia*, 99(1):4–9, 2007.

Frank Wood, Jan Willem van de Meent, and Vikash Mansinghka. A new approach to probabilistic programming inference. In *AISTATS*, pages 1024–1032, 2014.

Yi Wu, Lei Li, Stuart Russell, and Rastislav Bodik. Swift: Compiled inference for probabilistic programming languages. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

Onno Zoeter and Tom Heskes. Gaussian quadrature based expectation propagation. In *Proceedings of AISTATS*, 2005.