Bayesian optimisation for fast approximate inference in state-space models with intractable likelihoods

Johan Dahlin, Mattias Villani and Thomas B. Schönb

December 21, 2016

Abstract

We consider the problem of approximate Bayesian parameter inference in non-linear state-space models with intractable likelihoods. Sequential Monte Carlo with approximate Bayesian computations (SMC-ABC) is one approach to approximate the likelihood in this type of models. However, such approximations can be noisy and computationally costly which hinders efficient implementations using standard methods based on optimisation and Monte Carlo methods. We propose a computationally efficient novel method based on the combination of Gaussian process optimisation and SMC-ABC to create a Laplace approximation of the intractable posterior. We exemplify the proposed algorithm for inference in stochastic volatility models with both synthetic and real-world data as well as for estimating the Value-at-Risk for two portfolios using a copula model. We document speed-ups of between one and two orders of magnitude compared to state-of-the-art algorithms for posterior inference.

Keywords: α-stable distributions, approximate Bayesian computations, Bayesian inference, Gaussian process optimisation, sequential Monte Carlo
1 Introduction

Dynamical modelling of time series data is an essential part of modern econometrics [McNeil et al., 2010]. A popular model is the state-space model (SSM) used extensively in both macroeconomics and financial economics to capture e.g. the volatility of log-returns using stochastic volatility (SV) models. Recently, it has been proposed by Stoyanov et al. [2010] and others to model the jumps frequently observed in log-returns using an $\alpha$-stable distribution [Nolan, 2003], which allows heavy-tailed and possibly skewed log-returns.

However, inference in models with $\alpha$-stable distributions is challenging and often takes significantly longer time to carry out even on modern computers. This is a hindrance to wide-spread adoption of such models in applied work. One such example application is Value-at-Risk (VaR) modelling for a portfolio of assets, where the volatility of each asset is captured using a SV model with $\alpha$-stable log-returns. Another example is multi-fractal learning models [Calvet and Czellar, 2015] to model asset prices.

We aim to develop an efficient Bayesian approach for parameter inference in SSMs,

$$
\begin{align*}
x_0 &\sim \mu_\theta(x_0), & x_t|x_{t-1} &\sim f_\theta(x_t|x_{t-1}), & y_t|x_t &\sim g_\theta(y_t|x_t),
\end{align*}
$$

where $\theta \in \Theta \subseteq \mathbb{R}^p$ denotes unknown static parameters. Here, $x_t \in \mathcal{X} \subseteq \mathbb{R}^n_x$ and $y_t \in \mathcal{Y} \subseteq \mathbb{R}^n_y$ denotes the latent state and the observations at time $t \in \{0, 1, \ldots, T\}$. It is also possible to include discrete states required e.g. by the multi-fractal learning model.

An important object in Bayesian inference is the parameter posterior distribution,

$$
p(\theta|y_{1:T}) = \frac{p(\theta)p_\theta(y_{1:T})}{\int_\Theta p(\theta')p_\theta(y_{1:T}) \, d\theta'},
$$

where $p(\theta)$ denotes the parameter prior distribution and we denote the likelihood (with slight abuse of notation) as $p_\theta(y_{1:T}) \triangleq p(y_1, y_2, \ldots, y_T|\theta)$. For an SSM, we say that the likelihood $p(y_1, y_2, \ldots, y_T|\theta)$ (and thereby the posterior) is intractable, when $g_\theta(y_t|x_t)$ cannot be evaluated point-wise. This intractability can be the result of that $g_\theta(y_t|x_t)$ lacks an analytical closed-form expression, is defined recursively or is computationally prohibitive to evaluate.

One approach for inference in models with intractable likelihoods is approximate Bayesian computations (ABC; Marin et al. 2012). In our setting, we can make use of an ABC version of sequential Monte Carlo (SMC-ABC; Jasra et al. 2012, Calvet and Czellar 2015) to estimate the likelihood of a perturbed version of (2). It is then straight-forward to make use of SMC-ABC in many inference algorithms, e.g., particle Metropolis-Hastings (PMH; Andrieu et al. 2010, Dahlin and Schön 2015), gradient-based optimisation [Yıldırım et al., 2014] and the SPSA algorithm [Spall 1998, Ehrlich et al., 2015].

However, a problem with the estimates obtained from SMC-ABC is that they often have a large...
variance and are computationally expensive. This usually results in long run-times (days) of inference algorithms, which makes it difficult to apply ABC efficiently for inference in many important settings. To mitigate this problem, we propose an approach based on Gaussian process optimisation (GPO; Brochu et al., 2010) for extracting a Laplace approximation of the posterior. This is an iterative optimisation method that often requires substantially less posterior evaluations and is more robust to noise compared with many alternatives. GPO operates by constructing a surrogate function that mimics the parameter posterior in analogue with Wood (2010). The resulting surrogate is smooth and computationally cheap to evaluate, which enables the use of standard optimisation methods to extract a Laplace approximation of the surrogate mimicking the true posterior.

The main contribution of this paper is to introduce, develop and numerically study the proposed GPO-SMC-ABC algorithm, which is a combination of SMC-ABC and GPO. The proposed algorithm can be a competitive alternative for inference in any model where the likelihood is computationally costly to evaluate/estimate. We compare the proposed algorithm to PMH and SPSA for inference in SV models using both synthetic and real-world data. We demonstrate that GPO-SMC-ABC: (i) provides good posterior approximations using only a small number of evaluations of the posterior, (ii) reduces the computational time by between one and two orders of magnitude compared with PMH, (iii) exhibits good robustness to the ABC approximation and noise in the estimates. Furthermore, we demonstrate how to make use of the proposed algorithm to estimate the VaR in two different portfolios of assets. In these cases, the proposed algorithm can estimate the VaR in about an hour compared with a day when using PMH. Hence, this can enable routine inference in SSMs with intractable likelihoods, which opens up for investigating their use in real-world applications.

Work related to the proposed algorithm are presented in e.g. Dahlin and Lindsten (2014), Gutmann and Corander (2016) and Meeds and Welling (2014). In the first two works, the authors aim to obtain a maximum likelihood estimate and MAP estimate using GPO, respectively. In the present work, we would like to approximate the entire posterior and not only the parameter that maximises the value of the likelihood or posterior. Moreover, compared with Meeds and Welling (2014), the uncertainty encoded into the surrogate function is utilized to determine the next point in which to sample the log-posterior. Finally, we consider SSMs with intractable likelihoods and therefore employ SMC-ABC algorithms within the GPO algorithm to estimate the log-posterior. SMC-ABC often give estimates with a larger variance, which makes inference more challenging compared with tractable SSMs considered by Dahlin and Lindsten (2014).

We continue with Section 2 where an overview of the proposed algorithm and its components are presented. Sections 3 and 4 discuss the details of these components and the resulting algorithm is presented in Section 5. We conclude the paper with an extensive numerical evaluation in Section 6 on simulated and real-world data, and some final remarks and future work in Section 7.
Figure 1: Samples from the log-posterior (left) of (16) with respect to $\mu$. The distribution (center) and QQ-plot (right) of 1,000 log-posterior estimates of (16). The purple lines indicate the best Gaussian approximation.

2 An intuitive overview of GPO-SMC-ABC

Our aim is to construct a Laplace approximation of the parameter posterior (2),

$$\hat{p}(\theta|y_{1:T}) = N\left(\theta; \hat{\theta}_{\text{MAP}}, \left[-\nabla^2 \log p(\theta|y_{1:T})|_{\theta=\hat{\theta}_{\text{MAP}}} \right]^{-1}\right),$$

where $N(\theta; \mu, \Sigma)$ denotes the Gaussian distribution with mean $\mu$ and covariance matrix $\Sigma$. $\hat{J}(\hat{\theta}_{\text{MAP}})$ denotes the estimate of the Hessian of the log-posterior at the posterior mode,

$$\hat{\theta}_{\text{MAP}} = \arg\max_{\theta \in \Theta} \log p(\theta|y_{1:T}),$$

where $y_{1:T}$ denotes the recorded observations. This can be seen as a Gaussian approximation around the mode of the posterior motivated by the Bernstein-von Mises theorem, which states that the posterior concentrates to a Gaussian distribution centred at the true parameters with the inverse expected information matrix as its covariance as $T \to \infty$. Note that, even if this is an asymptotic result it can provide reasonable approximations using a finite number of samples as discussed by Panov and Spokoiny [2015].

We encounter two main problems when constructing the Laplace approximation: (i) the optimisation problem in (4) is difficult to solve efficiently and (ii) $\hat{J}(\hat{\theta}_{\text{MAP}})$ is typically difficult to estimate with good accuracy. The first problem is due to the high variance in and computational cost of the posterior estimates. An example of this problem is presented in the left part of Figure 1. Here, we present the log-posterior estimates obtained by SMC-ABC over a grid of $\mu$ in (16). The high variance typically results in a slow convergence of parameter inference algorithms such as SPSA and PMH. It is also difficult to speed up the convergence by using gradient information as this requires running computationally expensive particle smoothing algorithms.
Instead, we propose to circumvent these problems by optimising a smooth surrogate function that mimics the log-posterior distribution similar to [Wood 2010]. We can then obtain a Laplace approximation using standard optimisation methods as the surrogate function is smooth and cheap to evaluate. The surrogate function is obtained by GPO algorithm, which is a specific instance of Bayesian optimisation [Brochu et al. 2010]. The surrogate function sequentially is updated using samples from the log-posterior. In this paper, we make use of the predictive distribution of a Gaussian process (GP) as the surrogate function. Hence, we can encode certain prior knowledge regarding the log-posterior into the GP prior, which reduces the number of samples required to explore the log-posterior.

The resulting algorithm iterates three steps. At the $k$th iteration:

(i) compute an approximation of the log-posterior at the parameter $\theta_k$ denoted $\xi_k = \log \bar{p}(\theta_k | y_{1:T})$ using SMC-ABC.

(ii) construct a surrogate function by a GP predictive posterior using $\{\theta_k, \xi_k\} = \{\theta_j, \xi_j\}_{j=1}^k$.

(iii) evaluate the acquisition rule to determine $\theta_{k+1}$.

The GPO algorithm then returns an estimate of the posterior and its uncertainty. There are two major advantages of GPs for estimating the posterior density. First, the uncertainty quantification can be used to develop so-called acquisition rules that explores areas with large uncertainty and exploits the information about the possible locations of the mode of the posterior. This typically results in a rapid convergence of the algorithm, which limits the number of posterior samples required and hence also decreases the computational cost. This cannot be done using a spline model for the posterior. Secondly, the GP can handle noisy function evaluations in a natural manner.

3 Estimating the log-posterior

In this section, we discuss how to estimate the log-posterior that is required for carrying out Step (i) of GPO-SMC-ABC. Remember that the main difficulty is that the log-posterior is intractable but is can be estimated using a family of methods known as particle filters. We focus on a specific version of this procedure which is known as the bootstrap particle filter (bPF).

We obtain an estimate of the marginal filtering distribution $p_\theta(x_t | y_{1:t})$ from the bPF as

$$p_\theta^N(dx_t | y_{1:t}) = \sum_{i=1}^N w_t^{(i)} \delta_{x_t^{(i)}}(dx_t),$$

where $x_t^{(i)}$ and $w_t^{(i)}$ denotes the particle $i$ at time $t$ and its normalised weight, respectively. Here, $\delta_x$ denotes the Dirac measure placed at $x$. As we shall see, the particle system $\{w_t^{(i)}, x_t^{(i)}\}_{i=1}^N$ can also be used to obtain estimates of the log-posterior.
### 3.1 Particle filtering with approximate Bayesian computations

The main problem with applying the bPF is that we assume that $g_\theta(y_t|x_t)$ cannot be evaluated point-wise. To circumvent this, we reformulate the problem using an ABC approach. This is done by augmenting the posterior with an auxiliary variable $\tilde{y}_{1:T}$, which is data simulated from $g_\theta(y_t|x_t)$ for $t = 1, \ldots, T$. The fundamental assumption of ABC is therefore that data $\tilde{y}_t$ generated from $g_\theta(y_t|x_t)$ should be similar to the observed data $y_t$ if $\theta$ is properly selected.

The augmented posterior can be expressed as

$$
p_{\epsilon}(\theta, \tilde{y}_{1:T} | y_{1:T}) = \int_\theta p_{\epsilon}(\theta) p_{\theta}(\tilde{y}_{1:T}) \rho(\tilde{y}_{1:T}; y_{1:T}, \epsilon) d\theta',
$$

We then assume that $p_{\epsilon}(\theta, \tilde{y}_{1:T} | y_{1:T})$ can be marginalised with respect to $\tilde{y}_{1:T}$ to obtain

$$
p_{\epsilon}(\theta | y_{1:T}) = \int p_{\epsilon}(\theta, \tilde{y}_{1:T} | y_{1:T}) d\tilde{y}_{1:T},
$$

when the tolerance parameter is small enough and where $p_{\epsilon}(\theta | y_{1:T})$ denotes the posterior of the perturbed model. Here, we make use of some density $\rho(\mu, \epsilon)$ with mean $\mu$ and tolerance parameter $\epsilon$ to compute the distance between the simulated observations $\tilde{y}_t$ and the true observations $y_t$. A common choice is the Gaussian density $\rho(\mu, \epsilon) = N(\mu, \epsilon^2)$.

To make use of this in the bPF, we reformulate the model following Jasra et al. [2012]. The observed data $y_{1:T}$ is perturbed by

$$
\tilde{y}_t = \psi(y_t) + z_t, \quad z_t \sim \rho(0, \epsilon),
$$

where $\psi$ denotes some suitable one-to-one transformation. Moreover, we assume that it is possible to simulate from the model using a transformation of random variables. This corresponds to that we can write $\tilde{y}_t = \tau_\theta(\tilde{x}_t)$, where $\tau_\theta$ denotes a function of $\tilde{x}_t^\top = (x_t^\top, v_t^\top)$ and $v_t \sim \nu_\theta(v_t | x_t)$. This is an useful construction as it is often possible to generate samples from complicated distributions but not evaluate them point-wise. See Appendix A for how to select $\tau_\theta$ and $\nu_\theta$ to generate $\alpha$-stable random variables.

By using these two steps, we can rewrite the SSM (1) as

$$
\begin{align*}
\tilde{x}_t | \tilde{x}_{t-1} & \sim \Xi_\theta(\tilde{x}_t | \tilde{x}_{t-1}) = \nu_\theta(v_t | x_t) f_\theta(x_t | x_{t-1}), \\
\tilde{y}_t | \tilde{x}_t & \sim h_{\theta, \epsilon}(\tilde{y}_t | \tilde{x}_t) = \rho(\tilde{y}_t; \psi(\tau_\theta(\tilde{x}_t)), \epsilon).
\end{align*}
$$

We can now apply a bPF described in Algorithm 1 for this new model, which does not require us to evaluate the $g_\theta(y_t | x_t)$ point-wise but only that we can simulate $\tilde{y}_t$. In this paper, we leave the choice of
Algorithm 1 SMC-ABC

Inputs: \( y_{1:T} \) (perturbed data), SSM \( \{ \} \), \( N \in \mathbb{N} \) (no. particles), \( \rho_c \) (density) with \( \epsilon \in \mathbb{R}_+ \) (tolerance).

Outputs: \( \log \hat{p}^N(\theta|y_{1:T}) \) (est. of log-posterior).

Note: all operations are carried out over \( i, j = 1, \ldots, N \).

1: Sample \( \tilde{x}^{(i)}_0 \sim \mu_0(x_0)\nu_0(v_0|x_0) \) and set \( w^{(i)}_0 = 1/N \).
2: for \( t = 1 \) to \( T \) do
3:  Resample the particles using systematic resampling to obtain the ancestor index \( a^{(i)}_t \) from a multinomial distribution given by \( P(a^{(i)}_t = j) = w^{(j)}_{t-1} \).
4:  Propagate the particles by \( \tilde{x}^{(i)}_t \sim \Xi_\theta(\tilde{x}^{(i)}_{t-1}|a^{(i)}_t) \) and set \( \hat{x}^{(i)}_t = \{ \tilde{x}^{(i)}_{0:t-1}, \tilde{x}^{(i)}_t \} \).
5:  Compute the weights by \( W^{(i)}_t = h_{\theta,\epsilon}(\hat{y}_t|\hat{x}^{(i)}_t) \) and normalise by \( w^{(i)}_t = W^{(i)}_t \frac{1}{\sum_{j=1}^N W^{(j)}_t} \).
6: end for
7: Compute \( \log \hat{p}^N(\theta|y_{1:T}) \) by (8).

\( \epsilon \) to the user, which can be done using e.g. pilot runs on simulated data. However, it is possible to adapt \( \epsilon \) on-the-fly using the approaches discussed by e.g. [Del Moral et al. 2012] or [Calvet and Czellar 2015]. However in our experience, these methods require a larger value of \( N \) than un-adapted SMC-ABC to provide log-posterior estimates with a reasonable bias. Therefore, we decided to fix \( \epsilon \) in this paper to obtain computationally efficient algorithms. See Section 7 for some more discussion.

3.2 The estimator and its statistical properties

An estimator for the log-posterior of the perturbed model \( \hat{p}^N(\theta|y_{1:T}) \) is given by

\[
\log \hat{p}^N(\theta|y_{1:T}) = \log \hat{p}^N_\theta(y_{1:T}) + \log p(\theta) = T \log N + \log p(\theta),
\]

which use the unnormalised weights from the particle system generated by Algorithm 1.

It is known from [Pitt et al. 2012] that the estimator (8) is biased for a finite number of particles but it is consistent and asymptotically Gaussian. Specifically, we have that the error in the log-posterior estimate fulfills a CLT given by

\[
\sqrt{N} \left[ \log p(\theta|y_{1:T}) - \log \hat{p}^N(\theta|y_{1:T}) + \frac{\gamma^2(\theta)}{2N} \right] \overset{d}{\to} \mathcal{N}(0, \gamma^2(\theta)), \quad N \to \infty,
\]

for some unknown variance \( \gamma(\theta) \). As a result, we have an expression for the bias of the estimator given by \( -\gamma^2(\theta)/2N \) for a finite number of particles. However, we see that the error is approximately Gaussian for this type of model in the finite sample case by the experimental data presented in the center and right parts of Figure 1.

The log-posterior estimator in (8) is consistent with respect to the perturbed model (7) but not the true model. [Dean and Singh 2011] show that the perturbation results in a bias in the parameter estimates (compared with the unperturbed model) that decrease proportional to \( O(\epsilon^2) \) under some
regularity assumptions. Furthermore, the asymptotic Gaussianity of the estimator and a Bernstein-von Mises-type theorem holds for a small enough $\epsilon$. This is an important fact to motivate the Laplace approximation as discussed in Section 2. We return to a numerical investigation of these properties in Section 6.1.

4 Constructing the surrogate of the log-posterior

In this section, we briefly discuss Steps (ii) and (iii) of GPO-SMC-ABC, where we construct a surrogate function to mimic the log-posterior. See Brochu et al. [2010] for more details.

4.1 Gaussian process prior

GPs [Rasmussen and Williams, 2006] are an instance of Bayesian non-parametric models and can be seen as a generalisation of the multivariate Gaussian distribution to an infinite dimensional setting. A realisation drawn from a GP can therefore be seen as an infinite vector of values or as a function over the real space $\mathbb{R}^p$. To construct the surrogate function, we assume a priori that the log-posterior is distributed according to

$$\log p(\theta|y_{1:T}) \sim \mathcal{GP}(m(\theta), \kappa(\theta, \theta')).$$  \hspace{1cm} (10)

Note that this does not correspond to an assumption that the log-posterior of the parameters in the SSM is Gaussian. Here, $\mathcal{GP}(m, \kappa)$ denotes a GP with mean function $m$ and covariance function $\kappa$ defined by

$$m(\theta) = \mathbb{E} \left[ \log p(\theta|y_{1:T}) \right], \quad \kappa(\theta, \theta') = \mathbb{E} \left[ \left( \log p(\theta|y_{1:T}) - m(\theta) \right) \left( \log p(\theta'|y_{1:T}) - m(\theta') \right) \right].$$

The mean function specifies the expected value of the process and the covariance function specifies the dependence between any pair of points on the log-posterior function. The covariance function depends on a set of hyperparameters, such as the length scale that controls the dependence, see Rasmussen and Williams [2006] for details. From the CLT in (9) and Figure 1, we know that the error in the log-posterior is approximately Gaussian,

$$\xi_k = \log \hat{p}^N(\theta_k|y_{1:T}) \approx \log p(\theta_k|y_{1:T}) + \sigma_k z_k, \quad z_k \sim \mathcal{N}(0, 1).$$  \hspace{1cm} (11)
where \( \sigma^2_\xi \) denotes some unknown variance estimated in a later stage of the algorithm. Consequently, we have that the **predictive posterior** for any test point \( \theta_* \in \Theta \) is given by

\[
\log p(\theta_*|y_{1:T})|D_k \sim \mathcal{GP}\left(\mu(\theta_*|D_k), \sigma^2(\theta_*|D_k) + \sigma^2_\xi\right),
\]

(12a)

\[
\mu(\theta_*|D_k) = m(\theta_*) + \kappa(\theta_*, \theta_k)\left[\kappa(\theta_k, \theta_k) + \sigma^2_\xi I_{k \times k}\right]^{-1}\left\{\xi_k - m(\theta_*)\right\},
\]

(12b)

\[
\sigma^2(\theta_*|D_k) = \kappa(\theta_*, \theta_*) - \kappa(\theta_*, \theta_k)\left[\kappa(\theta_k, \theta_k) + \sigma^2_\xi I_{k \times k}\right]^{-1}\kappa(\theta_k, \theta_*),
\]

(12c)

where we have introduced the notation \( D_k = \{\theta_k, \xi_k\} \) for the information available at iteration \( k \). The surrogate function of the log-posterior is then given by (12).

The major cost in computing the predictive posterior is incurred by the matrix inversion which is proportional to \( O(k^3) \). Hence, sparse formulations of the GP can be useful to decrease the computation cost for large \( K \).

### 4.2 Acquisition function

The surrogate function given by the GP predictive distribution gives us the estimate of the log-posterior and its uncertainty. As previously mentioned, this is useful information for creating an acquisition rule \( \text{AQ}(\theta_*|D_k) \) to balance exploration and exploitation. We can then determine the next point in which to sample the log-posterior by

\[
\theta_{k+1} = \arg\max_{\theta_* \in \Theta_{\text{GPO}}} \text{AQ}(\theta_*|D_k),
\]

where \( \Theta_{\text{GPO}} \) denotes a search space defined by the user. In this paper, we make use of the expected improvement (EI) due to its general good performance in numerical evaluations \cite{Lizotte2008}. To derive the EI rule, consider the predicted improvement defined as

\[
\text{PI}(\theta_*) = \max \left\{0, \log p(\theta_*|y_{1:T}) - \mu_{\text{max}} - \zeta\right\}, \quad \forall \theta_* \in \Theta_{\text{GPO}},
\]

(13)

where \( \mu_{\text{max}} \) denotes the maximum value of \( \mu(\theta) \) for the sampled points \( \theta \in \Theta_k \). Here, we follow \cite{Lizotte2008} and introduce \( \zeta \) as a parameter that controls the exploitation/exploration behaviour. Hence for \( \zeta = 0 \), we have that \( \text{PI}(\theta_*) \) is the difference between the posterior and the maximum value it assumes in the sampled points. Therefore, it is positive for points where the log-posterior is larger than the current peak and zero for all other points. The EI rule is obtained by computing the expected value of (13) with
This results in the acquisition rule given by

\[
\theta_{k+1} = \left\{ \arg \max_{\theta^\star \in \Theta_{GPO}} \sigma(\theta^\star | D_k) \left[ Z(\theta^\star) \Phi(Z(\theta^\star)) + \phi(Z(\theta^\star)) \right] \right\} + \tilde{z}_k, \tag{14}
\]

where \( \phi \) and \( \Phi \) denotes the density and distribution function of the standard Gaussian distribution, respectively. Here, we jitter to the solution of the optimisation problem by adding Gaussian noise \( \tilde{z}_k \sim N(0, \Sigma) \) with covariance \( \Sigma \). In practice, this improves the exploration and increases the accuracy of the obtained parameter estimates. Jittering is also advocated by Bull [2011] and Gutmann and Corander [2016] to increase the convergence rate of GPO.

The optimisation in (14) is possibly non-convex but it is cheap to evaluate the objective function as it only amounts to evaluating the GP predictive posterior in one point. We make use of the the gradient-free dividing rectangles (DIRECT; Jones et al., 1993) to solve (14) over \( \Theta_{GPO} \), which is determined from the support of the prior distribution \( p(\theta) \).

5 The GPO-SMC-ABC algorithm

GPO-SMC-ABC (Algorithm 2) is obtained by combining SMC-ABC (Algorithm 1) to approximate the log-posterior point-wise and GPO to create a surrogate function that mimics the log-posterior around its mode. In this section, we discuss some user choices and convergence results for the algorithm. See Appendix A for the choices made in this paper.

5.1 Initialisation and convergence criteria

We initialise Algorithm 2 at Line 1 to find some suitable hyperparameters for the GP prior. The hyperparameters are estimated using \( L \) initial samples from the log-posterior obtained by Latin hypercube sampling. We then execute Algorithm 1 for each of the sampled parameters \( \{\theta^\star_1, \theta^\star_2, \ldots, \theta^\star_L\} \) to obtain \( D^\star_L \) by the analogue of Line 4 in Algorithm 2. After the initialisation of the algorithm, we can update the hyperparameters with Line 5 at every iteration or at some pre-defined interval. Estimating the hyperparameters is computationally costly and it is therefore recommended to re-estimate them only at some fixed interval. The algorithm is usually executed for some pre-defined number of iterations \( K \) or until the EI is smaller than some \( \Delta EI > 0 \), i.e. until \( k \) satisfies \( EI(\theta_k | D) < \Delta EI \).

5.2 Extracting the Laplace approximation

From Algorithm 2 we obtain the predictive posterior mean function \( \mu(\theta^\star | D) \), which hopefully is an accurate surrogate for the log-posterior. We can then proceed to extract the MAP estimate \( \hat{\theta}_{MAP} \).
Algorithm 2 GPO-SMC-ABC

**Inputs:** Algorithm 1, \( p(\theta) \) (parameter prior), \( m(\theta) \) (mean function), \( \kappa(\theta,\theta') \) (covariance function), \( \theta_1 \) (initial parameter), \( \Sigma \) (jittering covariance matrix) and \( \Theta_{\text{GPO}} \) (optimisation bounds).

**Output:** \( \hat{\theta}_{\text{MAP}} \) (est. of the parameter) and \( \hat{J}(\hat{\theta}_{\text{MAP}}) \) (est. of posterior covariance).

1. Estimate the hyperparameters of the GP prior by using some initial data \( D^*_L \).
2. Initialise the parameter to \( \theta_1 \) and set \( k = 1 \).
3. **while** convergence criteria is not satisfied **do**
   4. Estimate \( \xi_k = \log \hat{p}(\theta_k\mid y_{1:T}) \) by Algorithm 1 and set \( D_k = \{D^*_L, \theta_k, \xi_k\} \).
   5. (if required) Update the hyperparameters of the GP prior using \( D_k \).
   6. Construct the GP surrogate \( \log p(\theta^*_\star\mid y_{1:T})\mid D_k \) using (12).
   7. Compute \( \mu_{\text{max}} = \arg\max_{\theta \in \theta_k} \mu(\theta\mid D_k) \).
   8. Compute \( \theta_{k+1} \) by (14) using optimisation over \( \Theta_{\text{GPO}} \).
   9. Set \( k = k + 1 \).
10. **end while**
11. Compute the MAP estimate \( \hat{\theta} \) by optimising \( \mu(\theta\mid D_k) \) using optimisation over \( \Theta_{\text{GPO}} \).
12. Extract the Hessian estimate \( J(\hat{\theta}_{\text{MAP}}) \) using e.g. finite-differences on \( \mu(\theta\mid D_k) \).

defined by (4). As the surrogate is smooth and cheap to evaluate, we can carry out the optimisation using standard methods such as the DIRECT algorithm to find the mode. The estimate of the Hessian of the log-posterior \( J(\hat{\theta}_{\text{MAP}}) \) can be computed by a finite-difference scheme or by analytically computing the Hessian of \( \mu(\theta^*_\star\mid D) \) when possible. Moreover, it can be obtained using a quasi-Newton algorithm to solve (4).

5.3 Convergence properties

There are only a limited number of results regarding the convergence properties of the GPO algorithm in the literature. Most of the properties have been studied numerically by benchmarking the GPO algorithm against alternatives on a large number of optimisation problems. However, some theoretical results are discussed by [Bull 2011] and [Vazquez and Bect 2010]. They conclude that GPO using the EI rule samples the log-posterior densely if it is continuous with respect to the GP prior. Also, GPO achieves an optimal convergence rate of the order \( O\left( (K \log K)^{-5/p} (\log K)^{1/2} \right) \) for the Matérn 5/2 covariance function, where \( K \) and \( p \) denote the number of samples and parameters to infer, respectively.

6 Numerical illustrations and real-world applications

In this section, we provide four illustrations of the properties and advantages of the proposed algorithm. The implementation details are collected in Appendix A and the source code, data and results from runs are available from [https://github.com/compops/gpo-abc2015/](https://github.com/compops/gpo-abc2015/)
6.1 Stochastic volatility model with Gaussian log-returns

Consider the stochastic volatility model with Gaussian log-returns (GSV) given by

\[
\begin{align*}
x_0 &\sim N\left(x_0; \mu, \sigma^2_v \frac{1}{1 - \phi^2}\right), \\
x_{t+1} &\sim N\left(x_{t+1}; \mu + \phi(x_t - \mu), \sigma^2_v\right), \\
y_t &\sim N\left(y_t; 0, \exp(x_t)\right),
\end{align*}
\]

with parameters \( \theta = \{\mu, \phi, \sigma_v\} \). Here, the latent log-volatility is assumed to follow a mean-reverting random walk with mean \( \mu \in \mathbb{R} \), persistence \( \phi \in (-1, 1) \) and standard deviation of the increments \( \sigma_v \in \mathbb{R}_+ \). We generate a single synthetic data set from this model with \( T = 500 \) observations, parameters \( \theta^* = \{0.20, 0.96, 0.15\} \) and initial state \( x_0 = 0 \).

This model is interesting since it allows us to compare the GPO-SMC-ABC algorithm to an algorithm that uses standard SMC to estimate the log-posterior as we can evaluate \( g_\theta(y_t|x_t) \) in closed-form. We refer to this version of the algorithm as GPO-SMC and it corresponds to the GPO-SMC-ABC algorithm with tolerance parameter \( \epsilon \approx 0 \).

In the left part of Figure 2, we present the posterior estimates from GPO-SMC (solid curve) and PMH (histogram). We begin by observing a good fit of the Laplace approximation from GPO-SMC to the histogram approximation obtained by PMH; both the location and the spread of the posterior approximations are similar. Using GPO-SMC results in a speed-up of about 30 times \( (15,000/500 \approx 30) \) compared with PMH. The main computational cost for both algorithms is incurred by running the particle filter (one run per iteration). The overhead for the proposed algorithm (estimating hyperparameters in the GP and computing the predictive posterior) is negligible compared with the computational cost of running a particle filter.

We now continue with analysing the Laplace approximations obtained by GPO-SMC-ABC. In the right side of Figure 2 we present the posterior approximations obtained by varying the tolerance parameter \( \epsilon \) between 0.1 and 0.5 to study the bias and robustness of the approximation. Darker shades of grey indicate a larger tolerance parameter. For \( \epsilon = 0.10 \) and 0.2, we see that the approximations are rather poor with a significant bias and bad fit to the spread. However for the other choices of \( \epsilon \), the approximations from GPO-SMC-ABC converges quickly to be similar to GPO-SMC (solid curve). From these results, we conclude that we have a bias in the posterior approximation when \( \epsilon \) is too small and the approximation tends to grow wider as \( \epsilon \) increases. Hence, a bias-variance trade-off is introduced into the posterior approximation depending on the value of \( \epsilon \).

We continue by comparing GPO-SMC to SPSA, where the latter is a gradient-free alternative with good convergence properties in many applications. SPSA operates by constructing a finite-difference approximation of the gradient at each iteration after which it takes a step in the gradient direction. Note that SPSA requires two log-posterior estimates at each iteration compared with only one sample in the GPO-SMC. Another possible drawback with SPSA is that it only provides the MAP estimate and no
Figure 2: Marginal parameter posteriors for the synthetic data in the SVG model. Left: Solid curves indicate the Laplace approximations of the posterior using GPO-SMC for $\mu$ (green), $\phi$ (orange) and $\sigma_v$ (purple). The histograms represent the exact posteriors estimated using PMH and the dark grey (left) curves indicate the prior distributions. Right: Laplace approximations (shaded areas) from GPO-SMC for the three parameters. The grey curves (right) indicate the Laplace approximations obtained by GPO-SMC-ABC using five different values of the tolerance parameter $\epsilon$ in the ABC approximation.
Figure 3: The trace of the MAP estimate for $\mu$ (green), $\phi$ (orange) and $\sigma_v$ (purple) from GPO-SMC (solid) and SPSA (solid-circle) as a function of the number of log-posterior samples. The first $L = 50$ samples of GPO-SMC are used to estimate the hyperparameters. Both algorithms are run for a total of 700 log-posterior samples. Dashed lines indicate the posterior means from PMH.

quantification of the posterior uncertainty.

In Figure 3, we compare the MAP parameter estimates of two algorithms as a function of the number of log-posterior estimates. The first $L = 50$ samples of the log-posterior are used to estimate the hyperparameters of the GP prior. After this initial phase, GPO-SMC converges quickly to reasonable values of the parameters using about half the number of posterior samples. This results in a speed-up of factor 2 when using the proposed algorithm compared with SPSA.

6.2 Stochastic volatility model with $\alpha$-stable log-returns

In the upper part of Figure 4, we present the log-returns of future contracts on coffee during the period between June, 2013 and December, 2014. The data seem to indicate the presence of jumps around the first half of 2014. This is common in financial data and can be modelled in a number of different ways. Here, we consider the stochastic volatility model with $\alpha$-stable log-returns ($\alpha$SV) given by

$$x_0 \sim \mathcal{N}\left(x_0; \mu, \frac{\sigma_v^2}{1 - \phi^2}\right), \quad x_{t+1} \sim \mathcal{N}\left(x_{t+1}; \mu + \phi(x_t - \mu), \sigma_v^2\right), \quad y_t \sim \mathcal{A}\left(y_t; \alpha, \exp(x_t)\right),$$

with parameters $\theta = \{\mu, \phi, \sigma_v, \alpha\}$ and $\mathcal{A}(\alpha, \gamma)$ denoting a zero-mean symmetric $\alpha$-stable distribution with stability parameter $\alpha \in (0, 2)$ and scale $\gamma \in \mathbb{R}_+$. The stability parameter determines the tail behaviour of the distribution, see Nolan [2003] for a discussion of the $\alpha$-stable distribution and its properties. The likelihood is in general intractable for this model and therefore only PMH and GPO-SMC-ABC can be applied for this model.

In the middle and lower part of Figure 4, we compare the posterior approximations obtained with GPO-SMC-ABC (solid curves) with 10 independent runs and PMH (histograms). We see that the mixing
Figure 4: Upper: log-returns (green) of coffee futures and the estimate of the log-volatility (dark grey). The shaded area indicates the 95% confidence region for the log-returns according to the $\alpha$SV model. Middle and lower: marginal parameter posteriors in the $\alpha$SV model estimated by GPO-SMC-ABC (solid curves) using 10 independent runs and PMH (histogram) for $\mu$ (green), $\phi$ (orange), $\sigma_v$ (purple) and $\alpha$ (magenta). Dark grey curves indicate the prior distributions.
Algorithm 3 Copula modelling using αSV as the marginal models

Stage 1 (repeated for each asset $i$)

1: Run Algorithm 2 to obtain the log-volatility estimate $\hat{x}_{1:T,i}$ and the parameter estimate $\hat{\theta}_{\text{MAP},i}$ of (16). Compute the filtered log-returns $\hat{e}_{t,i}$ by

$$\hat{e}_{t,i} = \exp \left( -\frac{1}{2} \hat{x}_{t,i} \right) y_t, \quad \text{for } t = 1, \ldots, T.$$  

(17)

2: Estimate the cumulative distribution function (CDF) denoted by $\hat{G}_i$ from $\{\hat{e}_{t,i}\}_{t=1}^T$. Compute the probability transformation of the residuals by

$$\hat{u}_{t,i} = \hat{G}_i(\hat{e}_{t,i}), \quad \text{for } t = 1, \ldots, T.$$  

(18)

Stage 2

4: Infer the parameters of the copula to model the dependency between $\{\{\hat{u}_{t,i}\}_{t=1}^T\}_{i=1}^{30}$.

6.3 Computing VaR for a portfolio of oil futures

We follow Charpentier [2015] to construct a copula model to capture the dependency structure between prices of oil future contracts. The data that we consider is presented in Figure 5 and consists of weekly log-returns between January 10, 1997 and June 4, 2010 of Brent (produced in the North Sea), Dubai (produced in the Persian Gulf) and Maya (produced in the Gulf of Mexico) oil. We partition the data set into two parts and make use of the first 465 data points for estimating the αSV and copula models. The remaining 233 data points are kept for validating the model by back-testing.

We adopt the commonly used two-stage approach to copula modelling outlined in Algorithm 3, where marginal models are first fitted separately to each of the log-return series, and then combined using a copula to model the dependency structure [Joe, 2005]. We make use of the procedure in Section 6.2 to estimate the parameters of an αSV model for each type of oil. The filtered residuals (17) are assumed to be independent and identically distributed as $A(\hat{\alpha}_i, 1)$ by (16) if $\hat{x}_{1:T}$ is known. Stage 1 is carried out independently for each asset and is therefore straight-forward to implemented in a parallel manner. The use of the proposed algorithm decreases the computational time for this stage from hours or days for each asset to about half an hour compared with PMH.

We follow McNeil et al. [2010, p. 231-231] to model the dependency in the residuals. This amounts to applying a probability transform via the empirical CDF on the residuals into the 3-dimensional hypercube, see right part of Figure 5. The transformed residuals $\{\hat{u}_{1:T,1}, \ldots, \hat{u}_{1:T,d}\}$ are then combined
Figure 5: Left: weekly log-returns of Brent (green), Dubai (orange) and Maya (purple) oil between January 10, 1997 and June 4, 2010. The shaded area indicates the 95% confidence region for the log-returns according to the $\alpha$SV model. The dark grey curves indicate estimates of the log-volatility. Right: the corresponding transformed filtered residuals $\hat{u}_{t,i}$.
by a Student’s t-copula function to find a model for the joint distribution. The degrees of freedom and the correlation matrix in the Student’s t-copula are estimated using MAP and matching of moments via Kendall’s $\tau$, respectively.

Finally, we make use of the copula models and their margins to estimate the VaR for each type of oil. The VaR at confidence $\bar{\alpha} \in (0, 1)$ is defined by

$$\text{VaR}_{\bar{\alpha}}(e_t) = \inf \{-e_t \in \mathbb{R} : G(-e_t) \geq \bar{\alpha}\},$$

i.e. the smallest loss $-e_t$ such that probability that the loss (the negative log-return) exceeds $-e_t$ is no larger then $(1 - \bar{\alpha})$. We adopt a Monte Carlo approach to estimate $\text{VaR}_{\alpha}(e_t)$ by: (i) simulating from the copula, (ii) obtain simulated filtered residuals by applying a quantile transform based on the empirical CDF, (iii) computing the resulting log-returns by the estimated volatility and the inverse of Equation 17. We repeat this 100,000 times for each asset and then compute the empirical quantile corresponding to $\text{VaR}_{0.99}(e_t)$ for an equally weighted portfolio. However, more advanced portfolio weighting schemes can be implemented.

The resulting VaR-estimate is presented Figure 6 where we compared the $\alpha$SV model with a GSV model estimated using GPO-SMC in a similar manner. We note that the estimates from the two models are quite different especially at the end of the data series. Back-testing on the validation data gives 0 violations for both models and the expected number of violations is 2.3. The GPO-SMC-ABC algorithm requires around 30 – 60 minutes to infer the model for each of the three assets, where the inference is straight-forward to run on parallel architecture. The corresponding computational time required by the PMH algorithm would be in the order of 18 – 24 hours.
6.4 Computing VaR for a portfolio of stocks

We offer a final numerical example to illustrate that the proposed method can be applied to large portfolios as well. The data that we consider is the 30 industrial portfolio provided by Kenneth French. The portfolio consists of monthly log-returns of 30 different industrial sectors during the period September, 1926 to December, 2015. Again, we partition the data into an estimation set and a validation set with 716 and 358 data points, respectively.

We adopt the same procedure as in Section 6.3 and the results are presented in Figure 7. The conclusions are similar with both VaR estimates being quite similar. Back-testing gives 0 and 1 violations for the αSV and GSV models, respectively. The expected number of violations is 3.5. The computational speed-up with GPO-SMC-ABC compared with PMH is similar to in Section 6.3 as the number of observations per asset are similar.

7 Conclusions

We have proposed GPO-SMC-ABC, an algorithm to approximate the posterior distribution in SSMs with intractable likelihoods. The illustrations provided in Section 6 indicate that the proposed algorithm is quite accurate and exhibits a substantial decrease in the computational cost. We obtain similar posterior estimates to PMH with a speed-up of one or two orders of magnitude, which reduces computational time from tens of hours to tens of minutes. Moreover, GPO-SMC-ABC is more robust to noise in the log-posterior estimates, which typically results in that the PMH algorithm gets stuck at times and that the SPSA algorithm converges slowly. Overall, this shows that the proposed algorithm is an efficient inference method that makes it possible for practitioners to use models with intractable likelihoods, such as copula models with α-stable margins, in applied work.
Future work includes: (i) adopting a sparse presentation of the GP, (ii) developing new acquisition functions, (iii) making use of better tailored covariance functions and (iv) incorporating adaptive approaches for choosing the tolerance level $\epsilon$. Some ideas for sequential and sparse representations of GPs are discussed by [Huber 2014] and [Bijl et al. 2015]. It would also be interesting to design new acquisition functions to obtain good estimates of the Hessian of the log-posterior or higher order moments at the same time as the MAP estimate. This could improve the Laplace approximation of the parameter posterior and open up for alternative posterior approximations such as using the skewed Student’s $t$-distribution. Moreover, an interesting approach for tailored GP priors would be to make use of non-stationary covariance functions [Paciorek and Schervish 2004]. This would capture the fact that the log-posterior often falls off rapidly in some parts of the parameter space and is almost flat in other parts.

Finally, it would be beneficial to include some adaptive approach to select $\epsilon$ in the SMC-ABC algorithm to decrease the number of choices for the user. As previously discussed, we initially implemented the adaptive algorithms proposed in [Del Moral et al. 2012] and [Calvet and Czellar 2015]. However, we ran into problems when using them to approximate the log-posterior values using moderate values of $N$. The estimates exhibited a large bias that resulted in biased estimates of $\theta$ using PMH. Therefore, we choose not to adapt $\epsilon$ to be able to keep $N$ much smaller. It is therefore interesting to explore other adaptive schemes that provide good estimates of the log-posterior using a moderate value of $N$.

Acknowledgements

This work was supported by: *Probabilistic modeling of dynamical systems* (Contract number: 621-2013-5524) and *CADICS*, a Linnaeus Center, both funded by the Swedish Research Council. The simulations were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at Linköping University, Sweden. JD would like to thank Fredrik Lindsten, Neil Lawrence and Carl-Henrik Ek for fruitful discussions. Thanks also to Joerg M. Gablonsky, Abraham Lee, Per A. Brodtkorb and the GPy team for making their Python implementations available.

A Implementation details

**GPO algorithm:** We make use of the GPy package [The GPy authors 2014] for calculating the GP predictive posterior and estimating the GP prior hyperparameters. In this paper, we assume a zero prior mean function $m(\theta) = 0$ and a combination of the bias and the Matern 5/2 covariance functions. This choice corresponds to a prior for the log-posterior with some non-zero mean and two continuous derivatives. These are reasonable assumptions as this kind of smoothness is assumed in the Laplace approximation. We estimate the hyperparameters using empirical Bayes (EB), i.e. by optimising the marginal likelihood with respect to $\lambda$. 
For the acquisition rule in (14), we use $\zeta = 0.01$ and $\Sigma = 0.01I_p$. We initialise the GPO algorithm using $L = 50$ samples obtained using Latin hypercube sampling with the implementation written by Abraham Lee available at https://pypi.python.org/pypi/pyDOE. The optimisation problems in Lines 8 and 11 in Algorithm 2 are solved using the DIRECT implementation written by Joerg M. Gablonsky, available from https://pypi.python.org/pypi/DIRECT/. Finally for Line 12, we make use of the Python implementation by Per A. Brodtkorb available at https://pypi.python.org/pypi/Numdifftools.

Section 6.1: We use $N = 2,000$ particles in GPO-SMC and $N = 2,000$ particles with the Gaussian density with tolerance level $\epsilon = 0.20$ and $\psi(x) = x$ in GPO-SMC-ABC to produce the results in Figure 2. We run the GPO algorithms for $K = 450$ iterations after the initialisation and re-estimate the hyperparameters of the GP prior every 25th iteration. The search space for the GPO algorithm $\Theta_{GPO}$ is given by $\mu \in (0, 1)$, $\phi \in (0, 1)$ and $\sigma_v \in (0, 0.01)$. We use the following prior densities

$$p(\mu) \sim \mathcal{N}(\mu; 0, 0.2^2), \quad p(\phi) \sim \mathcal{T}\mathcal{N}_{(-1,1)}(\phi; 0.9, 0.05^2), \quad p(\sigma_v) \sim \mathcal{G}(\sigma_v; 2, 20),$$

where $\mathcal{T}\mathcal{N}_{(a,b)}(\cdot)$ denotes a truncated Gaussian distribution on $[a,b]$, $\mathcal{G}(a,b)$ denotes the Gamma distribution with mean $a/b$.

For PMH, we make use of the SMC-ABC algorithm with $N = 2,000$ particles to estimate the log-posterior. We initialise PMH in $\theta_0 = \{0.10, 0.95, 0.12\}$ and run the algorithm for $M = 15,000$ iterations (discarding the first 5,000 as burn-in). The parameter proposal is selected as

$$q(\theta' | \theta_k) = \mathcal{N}(\theta'; \theta_k, \Sigma_q), \quad \Sigma_q = \frac{2.562^2}{3} \cdot 10^{-4} \cdot \text{diag}(137, 7, 38),$$

which results from an asymptotic rule-of-thumb, see Dahlin and Schön [2015], with an estimate of the posterior covariance from a pilot run.

For SPSA, we make use of $N = 2,000$ particles and follow Spall [1998] to select the hyperparameters as $a = 0.001$, $c = 0.30$, $A = 35$, $\alpha = 0.602$ and $\gamma = 0.101$ using pilot runs.

Section 6.2: The real-world data is computed as $y_t = 100[\log(s_t) - \log(s_{t-1})]$, where $s_t$ denotes the price of a future contract on coffee obtained from https://www.quandl.com/CHRIS/ICE_KC2. We follow Yıldırım et al. [2014] and apply $\psi(x) = \arctan(x)$ to stabilise the variance of the likelihood (and gradient estimate). A two step approach is applied to sample from the zero-mean symmetric $\alpha$-stable distribution $\mathcal{A}(\alpha, \gamma)$. Firstly, we sample $v_t^{(1)} \sim \exp(1)$ and $v_t^{(2)} \sim \mathcal{U}(-\pi/2, \pi/2)$. Secondly, we obtain a sample (when $\alpha \neq 1$) by applying the transformation

$$\hat{y}_t = \gamma \frac{\sin(\alpha v_t^{(2)})}{[\cos(v_t^{(2)})]^{1/\alpha}} \left[\frac{\cos(\{(\alpha - 1)v_t^{(2)}\})}{v_t^{(1)}}\right]^{\frac{1-\alpha}{\alpha}},$$

see Nolan [2003] for more information on how to generate $\alpha$-stable random numbers.
We use $N = 2,000$ particles with the Gaussian density with tolerance level $\epsilon = 0.10$ in SMC-ABC to estimate the log-posterior. We run the GPO algorithm using the same settings as before but add $\alpha \in (1.2, 2)$ to the search space and $p(\alpha) \sim B(\alpha/2; 20, 2)$ to the prior distributions, where $B(a, b)$ denotes the Beta distribution. We initialise PMH in $\theta_0 = \{0.22, 0.93, 0.25, 1.55\}$ and the parameter proposal is selected using a pilot run as

$$q(\theta' | \theta_k) = N(\theta'; \theta_k, \Sigma_q), \quad \Sigma_q = \frac{2.562^2}{4} \cdot 10^{-3} \cdot \text{diag}(26, 1, 9, 11).$$

Section 6.3 and 6.4: The data sets are downloaded from http://freakonometrics.free.fr/oil.xls and http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html, respectively. We use $N = 5,000$ particles in the SMC and SMC-ABC algorithms and keep the remaining settings as Sections 6.1 and 6.2. For the stock portfolio example, we change the search space of $\mu$ to $(0, 4)$ as the weekly log-returns in this data set can be much larger than the daily log-returns in the other data sets. The MAP estimate of the degrees of freedom in the copula is obtained by a quasi-Newton solver using a uniform prior.

References

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

H. Bijl, J-W. van Wingerden, T. B. Schön, and M. Verhaegen. Online sparse Gaussian process regression using FITC and PITC approximations. In *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, pages 703–708, Beijing, China, October 2015.

E. Brochu, V. M. Cora, and N. De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Pre-print*, 2010. arXiv:1012.2599v1.

A. D. Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879–2904, 2011.

L. E. Calvet and V. Czellar. Accurate methods for approximate Bayesian computation filtering. *Journal of Financial Econometrics*, 13(4):798–838, 2015.

A. Charpentier. Prévision avec des copules en finance. Technical report, May 2015. URL https://hal.archives-ouvertes.fr/hal-01151233

J. Dahlin and F. Lindsten. Particle filter-based Gaussian process optimisation for parameter inference. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014.
J. Dahlin and T. B. Schön. Getting started with particle Metropolis-Hastings for inference in nonlinear models. *Pre-print*, 2015. arXiv:1511.01707v4.

T. A. Dean and S. S. Singh. Asymptotic behaviour of approximate Bayesian estimators. *Pre-print*, 2011. arXiv:1105.3655v1.

P. Del Moral, A. Doucet, and A. Jasra. An adaptive sequential Monte Carlo method for approximate Bayesian computation. *Statistics and Computing*, 22(5):1009–1020, 2012.

E. Ehrlich, A. Jasra, and N. Kantas. Gradient free parameter estimation for hidden Markov models with intractable likelihoods. *Methodology and Computing in Applied Probability*, 17(2):315–349, 2015.

M. U. Gutmann and J. Corander. Bayesian optimization for likelihood-free inference of simulator-based statistical models. *Journal of Machine Learning Research*, 17(125):1–47, 2016.

M. F. Huber. Recursive Gaussian process: On-line regression and learning. *Pattern Recognition Letters*, 45:85–91, 2014.

A. Jasra, S. S. Singh, J. S. Martin, and E. McCoy. Filtering via approximate Bayesian computation. *Statistics and Computing*, 22(6):1223–1237, 2012.

H. Joe. Asymptotic efficiency of the two-stage estimation method for copula-based models. *Journal of Multivariate Analysis*, 94(2):401–419, 2005.

D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.

D. J. Lizotte. *Practical Bayesian optimization*. PhD thesis, University of Alberta, 2008.

J-M. Marin, P. Pudlo, C. P. Robert, and R. J. Ryder. Approximate Bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012.

A. J. McNeil, R. Frey, and P. Embrechts. *Quantitative risk management: concepts, techniques, and tools*. Princeton University Press, 2010.

E. Meeds and M. Welling. GPS-ABC: Gaussian process surrogate approximate Bayesian computation. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI)*, Quebec City, Canada, July 2014.

J. Nolan. *Stable distributions: models for heavy-tailed data*. Birkhauser, 2003.

C. J. Paciorek and M. J. Schervish. Nonstationary covariance functions for Gaussian process regression. In *Proceedings of the 2004 Conference on Neural Information Processing Systems (NIPS)*, pages 273–280, Vancouver, Canada, December 2004.
M. Panov and V. Spokoiny. Finite sample Bernstein-von-Mises theorem for semiparametric problems. *Bayesian Analysis*, 10(3):665–710, 2015.

M. K. Pitt, R. S. Silva, P. Giordani, and R. Kohn. On some properties of Markov chain Monte Carlo simulation methods based on the particle filter. *Journal of Econometrics*, 171(2):134–151, 2012.

C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

J. C. Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):817–823, 1998.

S. V. Stoyanov, B. Racheva-Iotova, S. T. Rachev, and F. J. Fabozzi. Stochastic models for risk estimation in volatile markets: a survey. *Annals of Operations Research*, 176(1):293–309, 2010.

The GPy authors. GPy: A Gaussian process framework in Python. [http://github.com/SheffieldML/GPy](http://github.com/SheffieldML/GPy), 2014.

E. Vazquez and J. Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and inference*, 140(11):3088–3095, 2010.

S. N. Wood. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature Letters*, 466:1102–1104, 2010.

S. Yıldırım, S. S. Singh, T. Dean, and A. Jasra. Parameter estimation in hidden Markov models with intractable likelihoods using sequential Monte Carlo. *Journal of Computational and Graphical Statistics*, 24(3):846–865, 2014.