Particle filter-based Gaussian process optimisation for parameter inference*

Johan Dahlin* Fredrik Lindsten**

* Division of Automatic Control, Linköping University, Sweden.
E-mail: johan.dahlin@liu.se
** Dept. of Engineering, University of Cambridge, United Kingdom.
E-mail: fredrik.lindsten@eng.cam.ac.uk

Abstract: We propose a novel method for maximum-likelihood-based parameter inference in nonlinear and/or non-Gaussian state space models. The method is an iterative procedure with three steps. At each iteration a particle filter is used to estimate the value of the log-likelihood function at the current parameter iterate. Using these log-likelihood estimates, a surrogate objective function is created by utilizing a Gaussian process model. Finally, we use a heuristic procedure to obtain a revised parameter iterate, providing an automatic trade-off between exploration and exploitation of the surrogate model. The method is profiled on two state space models with good performance both considering accuracy and computational cost.

Keywords: Maximum Likelihood, Parameter Estimation, Gaussian Processes, Sequential Monte Carlo, Machine Learning, Bayesian Optimisation.

1. INTRODUCTION

We are interested in maximum likelihood-based (ML) parameter inference in nonlinear and/or non-Gaussian state space models (SSM). An SSM with latent states $x_{0:T} \triangleq \{x_t\}_{t=0}^T$ and measurements $y_{1:T} \triangleq \{y_t\}_{t=1}^T$ is defined as

$$x_t | x_{t-1} \sim f_\theta(x_t | x_{t-1}),$$

$$y_t | x_t \sim g_\theta(y_t | x_t),$$

where $f_\theta(\cdot)$ and $g_\theta(\cdot)$ denote known distributions parameterised by the unknown static parameter vector $\theta \in \Theta \subseteq \mathbb{R}^d$. For simplicity, we assume that the initial state $x_0$ is known. Let $\mathcal{L}(\theta) \triangleq p_\theta(y_{1:T})$ denote the likelihood of $y_{1:T}$ for a given value of $\theta$. In ML estimation, we wish to estimate $\theta$ by solving the optimisation problem,

$$\hat{\theta}_{\text{ML}} = \arg\max_{\theta \in \Theta} \mathcal{L}(\theta) = \arg\max_{\theta \in \Theta} \ell(\theta),$$

where $\ell(\theta) \triangleq \log \mathcal{L}(\theta)$ denotes the log-likelihood function. Extensive treatments on ML inference are found in e.g. Ljung [1999] and Lehmann and Casella [1998].

The likelihood for a general SSM can be expressed as

$$\mathcal{L}(\theta) = \prod_{t=1}^T p_\theta(y_t | y_{1:t-1}),$$

where $p_\theta(y_t | y_{1:t-1})$ denotes the one-step predictive density. For a linear Gaussian models, these densities can be computed exactly by using the Kalman filter. However, for a nonlinear model the one-step predictive densities are in general intractable. It is therefore also intractable to evaluate the objective function in (2), which poses an obvious difficulty in addressing the ML problem.

Recently, ML estimation has been carried out in nonlinear SSMs by the aid of Sequential Monte Carlo [Doucet and Johansen, 2011]. This includes e.g. using gradient-based search [Poyiadjis et al., 2011] and the Expectation Maximisation (EM) algorithm [Schön et al., 2011, Lindsten, 2013]. However, some of these methods require computationally costly particle smoothing to estimate the necessary quantities, which can be a problem in some situations.

An alternative is to make use of the simultaneous perturbation stochastic approximation (SPSA) algorithm [Spall, 1987], which uses a steepest ascent algorithm with a stochastic approximation scheme to estimate the solution to (2). The gradients are estimated using finite differences with random perturbations. This results in that the algorithm only needs to sample the likelihood function twice at each iteration, independent of the dimension of the problem. SPSA is used in combination with SMC in e.g. Singh et al. [2011] and Ehrlich et al. [2012].

Another approach for maximum likelihood estimation is based on approximate inference based on Laplace approximations and moment matching. We do not consider these methods any further in this paper and refer interested readers to e.g. Bishop [2006], Khan et al. [2012] and Bell [2000] for more information.

In this paper, we propose a novel algorithm for ML estimation of static parameters in a nonlinear SSM. The method combines particle filtering (PF) with Gaussian process optimisation (GPO) [Jones, 2001, Boyle, 2007, Lizotte, 2008]. The latter is a method well-suited for optimisation when it is costly to evaluate the objective function. The resulting algorithm is efficient in the sense that it provides accurate parameter estimates while making use of only a small number of (costly) log-likelihood evaluations.

* Supported by the project Probabilistic modelling of dynamical systems (Contract number: 621-2013-5524) funded by the Swedish Research Council.
2. MAXIMUM LIKELIHOOD ESTIMATION WITH A SURROGATE COST FUNCTION

We now turn to our new procedure for ML estimation of general nonlinear SSMs (1). We start by outlining the main ideas of the procedure on a high level. The individual steps of the algorithm are discussed in detail in the consecutive sections. The algorithm is an iterative procedure, which thus generates a sequence of iterates \( \{ \theta_k \}_{k \geq 0} \) for the model parameters. Each iteration consists of three main steps:

(i) Given the current iterate \( \theta_k \), compute an estimate of the objective function (i.e. the log-likelihood) for this parameter value, denoted as \( \hat{\ell}_k \approx \ell(\theta_k) \).

(ii) Given the collection of tuples \( \{ \theta_j, \hat{\ell}_j \}_{j=0}^k \) generated up to the current iterate, create a model of the (intractable) objective function \( \ell(\theta) \).

(iii) Use the model as a surrogate for the objective function to generate a new iterate \( \theta_{k+1} \).

Note that the method requires only one estimation of the log-likelihood function at each iteration. This is promising, since it is typically computationally costly to estimate the log-likelihood value and we therefore wish to keep the number of such evaluations as low as possible.

For step (i), i.e. evaluating the log-likelihood function for a given value of \( \theta \), we use a PF, resulting in a (noisy) estimate of the objective function. This step is discussed in Section 3. For steps (ii) and (iii), we apply the GPO framework. First, we construct a surrogate for the objective function by modelling it as a Gaussian process, taking the information available in the previous iterates \( \{ \theta_j, \hat{\ell}_j \}_{j=0}^k \) into account. This is discussed in Section 4.

Then, we make use of a heuristic, referred to as an acquisition rule, to find the next iterate \( \theta_{k+1} \) based on the GP model. The acquisition rule is such that it favours values of \( \theta \) for which the model predicts a large value of the objective function and/or where there is a high uncertainty in the model. This is useful since it automatically results in a trade-off between exploration and exploitation of the model.

In this paper, we consider a simple numerical example to illustrate the different steps of the algorithm during the derivation. For this, the linear Gaussian state space (LGSS) model,

\[
\begin{align*}
x_{t+1} &\sim N(x_{t+1}; \theta, x_t, 1), \\
y_t &\sim N(y_t; x_t, 0.1^2),
\end{align*}
\]

with \( \Theta = [-1, 1] \) and parameter \( \theta^* = 0.5 \) is simulated for \( T = 250 \) time steps. The complete algorithm is evaluated in Section 6 on this model, as well as on a nonlinear SSM.

3. ESTIMATING THE LOG-LIKELIHOOD

We begin this section with a brief description of a PF. For more general introductions, see e.g. Doucet and Johansen [2011]. We then continue with discussing the specific problem of likelihood estimation using the PF.

3.1 The particle filter

The PF is a sequential Monte Carlo method used to approximate e.g. the intractable filtering distribution \( p_\theta(x_t|y_{1:t}) \) for a general SSM (1). This is done by representing it by a set of \( N \) weighted particles \( \{ x^{(i)}_t, w^{(i)}_t \}_{i=1}^N \) according to

\[
\hat{p}_\theta(dx_t|y_{1:t}) \approx \frac{1}{N} \sum_{i=1}^N w^{(i)}_t \delta_{x^{(i)}_t}(dx_t),
\]

where \( w^{(i)}_t \) and \( x^{(i)}_t \) denote the (unnormalised) weight and state of particle \( i \) at time \( t \), respectively. Here, \( \delta_{x^{(i)}_t} \) denotes the Dirac measure located at the point \( x^{(i)}_t \). These approximations are generated sequentially in time \( t \). Given the particles at time \( t-1 \), the PF proceeds to time \( t \) by:

(a) resampling,
(b) propagation and
(c) weighting.

In step (a), the particles are resampled with replacement, using the probabilities given by their (normalized) importance weights. This is done to rejuvenate the particle system and to put emphasis on the most probable particles. The result is an unweighted particle system \( \{ \hat{x}^{(i)}_{t-1:1}/N \}_{i=1}^N \) at time \( t-1 \).

In step (b), the particles are propagated to time \( t \) by sampling from a proposal kernel \( x^{(i)}_t \sim R_\theta(x_t|x^{(i)}_{t-1}, y_t) \) for \( i = 1 \) to \( N \). Finally in Step (c), the particles are assigned importance weights. This is done to account for the discrepancy between the proposal and the target densities. The importance weights are given by

\[
w^{(i)}_t = \frac{w^{(i)}_t x^{(i)}_t}{R_\theta(x^{(i)}_{t-1:1}/N, x^{(i)}_{t-1:1}/N)}.
\]

In the sequel, we use the bootstrap PF which means that new particles are proposed according to the state dynamics, i.e. \( R_\theta(\cdot) = f_\theta(\cdot) \) and \( w^{(i)}_t = g_\theta(y_t|x^{(i)}_t) \). Although more sophisticated alternatives exist, see e.g. the fully-adapted PF introduced in Pitt and Shephard [1999].

3.2 Estimation of the likelihood

In order to use the PF for estimating the likelihood, we start by writing the one-step predictive density as

\[
p_\theta(y_t|y_{1:t-1}) = \int p_\theta(y_t, x_t|x_{t-1}) p_\theta(x_{t-1}|y_{1:t-1}) \text{d}x_{t-1:t} = \int W_\theta(x_t, x_{t-1}) R_\theta(x_t|x_{t-1}, y_t) p_\theta(x_{t-1}|y_{1:t-1}) \text{d}x_{t-1:t},
\]

where we have multiplied and divided with the proposal kernel \( R_\theta(\cdot) \). To approximate the integral, we note that the (unweighted) particle pairs \( \{ x^{(i)}_{t-1:1}, x^{(i)}_{t-1:1} \}_{i=1}^N \) are approximately drawn from \( R_\theta(x_t|x_{t-1}, y_t) p_\theta(x_{t-1}|y_{1:t-1}) \). Consequently, we obtain the Monte Carlo approximation

\[
p_\theta(y_t|y_{1:t-1}) \approx \frac{1}{N} \sum_{i=1}^N w^{(i)}_t.
\]

By inserting this approximation into (3) we obtain the particle estimate of the likelihood,

\[
\hat{L}(\theta) = \prod_{t=1}^T \left( \frac{1}{N} \sum_{i=1}^N w^{(i)}_t \right).
\]

This likelihood estimator has been studied extensively in the SMC literature. The estimator is consistent and, in fact, also unbiased for any \( N \geq 1 \); see e.g. Pitt et al. [2012].
Algorithm 1: PF for log-likelihood estimation

Inputs: An SSM (1), \( y_{1:T} \) (obs.) and \( N \) (no. particles).
Output: \( \hat{\ell}(\theta) \) (est. of the log-likelihood).

1: Initialise particles \( x_0^{(i)} \) for \( i = 1 \) to \( N \).
2: for \( t = 1 \) to \( T \) do
3: Resample the particles with weights \( \{w_t^{(i)}\}_{i=1}^N \).
4: Propagate the particles using \( R_t(\cdot) \).
5: Compute (5) to obtain \( \{w_t^{(i)}\}_{i=1}^N \).
6: end for
7: Compute (6) to obtain \( \tilde{\ell}(\theta) \).

and Proposition 7.4.1 in Del Moral [2004]. Furthermore, a central limit theorem holds,

\[
\sqrt{N} \left[ \hat{\ell}(\theta) - \ell(\theta) \right] \xrightarrow{d} \mathcal{N}\left(0, \psi^2(\theta)\right),
\]

for some asymptotic variance \( \psi^2(\theta) \); see Proposition 9.4.1 in Del Moral [2004].

3.3 Estimation of the log-likelihood

However, working directly with the likelihood typically results in numerical difficulties. To avoid problems with numerical precision, we instead use an estimate of the log-likelihood

\[
\hat{\ell}(\theta) = \log \hat{\mathcal{L}}(\theta) = \sum_{t=1}^{T} \log \left[ \sum_{i=1}^{N} w_t^{(i)} \right] - T \log N. \tag{6}
\]

The resulting complete algorithm for estimating the log-likelihood using a PF is presented in Algorithm 1.

Note that, by taking the logarithm of \( \hat{\mathcal{L}}(\theta) \), we introduce a bias into the estimator. However, by the second-order delta method [Casella and Berger, 2001], the asymptotic normality carries over to the log-likelihood estimate,

\[
\sqrt{N} \left[ \hat{\ell}(\theta) - \ell(\theta) \right] \xrightarrow{d} \mathcal{N}\left(0, \gamma^2(\theta)\right), \tag{7}
\]

where \( \gamma(\theta) = \psi(\theta)/\mathcal{L}(\theta) \). Motivated by this, we make the assumption that the log-likelihood estimates are Gaussian distributed and centered around the true log-likelihood value. That is, we can write

\[
\hat{\ell}(\theta) = \ell(\theta) + z, \quad z \sim \mathcal{N}(0, \sigma^2). \tag{8}
\]

Similar normality assumptions have previously been used by Pitt et al. [2012] and Doucet et al. [2012]. The unknown variance \( \sigma^2 \) is treated as a free parameter that is estimated on-the-fly as we run the proposed estimation algorithm. That is, we do not have to estimate \( \sigma^2 \) by making any initial test runs. We return to this in the sequel.

We validate the Gaussian assumption (8) using a small numerical experiment to illustrate the bias and variance, at a finite number of particles. We calculate 1000 estimates of the log-likelihood \( \hat{\ell}(0.5) \) for the model in (4). This is done by running Algorithm 1 independently 1000 times with \( N = 1000 \) particles.

In Figure 1, we present the distribution of the error in the estimates together with a QQ-plot. Both plots validate that the estimates are approximately distributed according to a Gaussian distribution. Also a Lilliefors hypothesis test [Lilliefors, 1967] does not reject the null hypothesis, that the measurements are drawn from a Gaussian distribution at significance level \( \alpha = 0.05 \).

Fig. 1. Left: the histogram and kernel density estimate (blue line) of the estimation error of the log-likelihood in the LGSS model (4) at \( \theta = \theta^* \). Right: the QQ-plot of the data with the theoretical quantiles marked with the solid blue line.

4. MODELLING THE SURROGATE FUNCTION

From the previous, we consider a naive approach to solve (2) by creating a grid of the parameter space and estimating the log-likelihood in each grid point. The parameter estimate is then obtained as the grid point that maximises the objective function. The problem here is that as the dimension of the parameter space increases, an exponentially increasing number of grid points is required to retain the accuracy of the estimate.

Furthermore, using finite differences to compute the gradient of the log-likelihood is problematic due to the noise in (8). This problem can be mitigated by using a particle smoother, as previously discussed in e.g. Poyiadjiis et al. [2011], but this is even more computationally expensive than running the particle filter. Instead, we construct a model of the noisy log-likelihood evaluations in Step (ii). This model then serves as a surrogate for the actual objective function.

4.1 Gaussian process model

In this paper, we use a GP for this purpose, as these processes are possibly flexible enough to capture the overall structure of the log-likelihood for many SSMs. GPs can be seen as a generalisation of the multivariate Gaussian distribution and are commonly used as priors over functions. In this view, the resulting posterior obtained by conditioning upon some observations, describes the functions that could have generated the observations. This makes GPs a popular class of nonparametric models used for e.g. regression, classification and optimisation, see e.g. Rasmussen and Williams [2006] and Murphy [2012].

In the following, we model the log-likelihood \( \ell(\theta) \) as being a priori distributed according to a GP. That is,

\[
\ell(\cdot) \sim \mathcal{GP}(m(\cdot), \kappa(\cdot, \cdot)), \tag{9}
\]

where the process is fully described by the mean function \( m(\cdot) \) and the covariance function \( \kappa(\cdot, \cdot) \).

4.2 Updating the model and the hyperparameters

To ease the presentation, we here consider a particular iteration \( k \) of the GP and the PF. Let \( \mathcal{D}_k = \{\theta_k, \hat{\ell}_k\} = \{\theta_j, \hat{\ell}(\theta_j)\}_{j=1}^k \) denote a set of iterates, where \( \theta_k \) and \( \hat{\ell}_k \)
denote vectors obtained by stacking the \( k \) parameters and noisy log-likelihood estimates, respectively.

It follows that the posterior distribution is given by
\[
\ell(\theta) | D_k \sim N\left( \mu(\theta | D_k), \sigma^2(\theta | D_k) + \sigma_z^2 \right),
\]
where \( \mu(\theta | D_k) \) and \( \sigma^2(\theta | D_k) \) denote the posterior mean and variance given the iterates \( D_k \), respectively. By standard results for the Gaussian distribution, we have
\[
\mu(\theta | D_k) = m(\theta) + \kappa(\theta, \theta_k)\Gamma^{-1} \left( \hat{\ell}_k - m(\theta) \right),
\]
\[
\sigma^2(\theta | D_k) = \kappa(\theta, \theta) - \kappa(\theta, \theta_k)\Gamma^{-1}\kappa(\theta_k, \theta),
\]
with \( \Gamma = \kappa(\theta_k, \theta_k) + \sigma_z^2 I_{k \times k} \), and where \( I_{k \times k} \) denotes a \( k \times k \)-identity matrix. Here we note that the posterior distribution can be sequentially updated to save computations, see the aforementioned references for details.

In the GP model presented, we use some mean function and covariance function that possibly depend on some unknown hyperparameters. Also, we need to estimate the unknown noise variance \( \sigma_z^2 \) in (8). For this, we adopt the empirical Bayes (EB) procedure to estimate these quantities. This is done by numerically optimising the marginal likelihood of the data with respect to the hyperparameters.

4.3 Example of log-likelihood modelling

We end this section by an example to illustrate the usefulness of GPs in modelling the log-likelihood. In the upper part of Figure 2, we show the posterior distribution of the log-likelihood of the model in (4). The posterior is estimated using three (left) and six (right) samples of the log-likelihood drawn at some randomly selected parameters. With information from only six samples, the mean of the surrogate function passes close to the observed iterates with a reasonable confidence interval.

5. ACQUISITION RULES

The remaining problem in the proposed algorithm is how to select the parameters at which the log-likelihood should be evaluated in step (iii). A simple choice would be to consider a random sampling approach, which works well when the dimension of the parameters is small. However, when the dimension increases, we are faced with the curse-of-dimensionality and independent sampling is inefficient.

As previously discussed, we instead use acquisition rules that balances exploration and exploitation of the parameter space and makes use of the posterior distribution obtained from the GP. These heuristics are well-studied in GPO and simulation-based comparisons are presented in e.g. Lizotte [2008]. In this paper, we follow their general recommendations and use the expected improvement (EI) from Jones [2001].

5.1 Expected improvement

Consider the predicted improvement defined as
\[
I(\theta) = \max \left\{ 0, \ell(\theta) - \mu_{\text{max}} - \zeta \right\},
\]
where \( \zeta \) is a user-defined coefficient that balances exploration and exploitation. Also, introduce the expected peak of the log-likelihood function,

\[
\mu_{\text{max}}(\theta) = \max_{\theta \in \Theta} \mu(\theta | D_k),
\]
over the previous iterates. Here, we again consider a particular iteration \( k \) in the notation for brevity.

Finally, by using the posterior distribution obtained from the GP, we can write the EI as
\[
E[I(\theta) | D_k] = \sigma(\theta) \left[ Z(\theta) \Phi(Z(\theta)) + \phi(Z(\theta)) \right],
\]
(14)

where we drop the dependence on \( D_k \) for brevity. Here, \( \Phi \) and \( \phi \) denote the CDF and PDF of the standard Gaussian distribution, respectively. An acquisition rule follows by the maximising argument
\[
\theta_{k+1} = \arg\max_{\theta \in \Theta} E[I(\theta) | D_k],
\]
(15)
i.e. we sample the likelihood in \( \theta_{k+1} \) during the next iteration of the algorithm.

In the lower part of Figure 2, the expected improvements are shown for the situation discussed in the previous example. The two situations correspond to an exploration step (left) and an exploration step (right), respectively. In the former, we sample in the neighbourhood of the current predicted peak. In the latter, we sample in an area where the uncertainty is large to determine if there is a peak in that area.

From the expression in (14), we expect a high value of EI for parameters where the variance \( \sigma(\theta) \) is large. If also the predictive mean \( \mu(\theta) \) is larger than \( \mu_{\text{max}} \), then the EI assumes even larger values for these parameters. This gives the desired behaviour of the acquisition function discussed previously.
Algorithm 2 Particle-based parameter inference in nonlinear SSMs using Gaussian process optimisation

Inputs: Algorithm 1, $K$ (no. iterations) and $\theta_1$ (initial parameter).

Output: $\hat{\theta}$ (est. of the parameter).

1. Initialise the parameter estimate in $\theta_1$.
2. for $k = 1$ to $K$ do
3. Sample $\ell(\theta_k)$ using Algorithm 1.
4. Compute (10) and (11) to obtain $f(\theta|D_k)$.
5. Compute (13) to obtain $\mu_{\text{max}}$.
6. Compute (15) to obtain $\theta_{k+1}$.
7. end for
8. Compute the maximiser $\mu(\theta|D_K)$ to obtain $\hat{\theta}$.

6. NUMERICAL ILLUSTRATIONS

Finally, we are ready to combine the methods discussed in the previous sections into the final algorithm and it is presented in Algorithm 2. In the following, we use an LGSS model and a nonlinear model to illustrate the behaviour and the performance of the proposed algorithm. We compare the proposed method in the latter model with the SPSA algorithm [Spall, 1987]. This algorithm is selected as it also only makes use of zero-order information (the log-likelihood estimates) and is known to perform well in many problems, see e.g. Spall [1998].

6.1 Implementation details

For the GP, we use a constant mean function and the Matérn kernel with $\nu = 3/2$. Note that, other choices of mean functions and kernels (especially the combination of kernels) can possibly improve the performance of the algorithm. This is especially important in models where the log-likelihood in non-isotropic.

The GPML toolbox [Rasmussen and Williams, 2006] is used for estimation of the hyperparameters by EB and for the computation of the predictive distribution in (10). For the acquisition function, we use the EI with $\zeta = 0.01$ following the recommendations in Lizotte [2008].

The optimisation in (15) is non-convex and therefore difficult to carry out in a global setting. Two common approaches in GPO are to use multiple local search algorithms in a Monte Carlo setting [Lizotte, 2008] or using a global optimisation algorithm [Brochu et al., 2010]. In this paper, we use the latter method with the gradient-free DIRECT global optimisation algorithm [Jones et al., 1993] and the implementation written by Daniel E. Finkel, available from http://www4.ncsu.edu/~ctk/Finkel_Direct/. A maximum of 500 iterations and (cheap) evaluations of the surrogate function are used in the DIRECT algorithm for each optimisation.

6.2 Linear Gaussian state space model

We begin with the LGSS model using one parameter in (4), as this enables us to investigate the behaviour of the proposed algorithm in detail. We use $N = 1000$ particles, $K = 50$ iterations and the initial parameter $\theta_1 = -0.98$. In Figure 3, we present the surrogate function and the expected improvement at different iterations. The algorithm converges rather quickly for this simple toy example with the parameter estimate $\hat{\theta} = 0.48$. As a comparison, the MLE obtained by the Kalman filter by maximisation on a grid of parameter values is $\hat{\theta}_{\text{MLE}} = 0.44$.

6.3 Nonlinear stochastic volatility model

Consider the Hull-White stochastic volatility model [Hull and White, 1987],

\[
\begin{align}
    x_{t+1} | x_t &\sim N(\mu, \sigma^2) , \\
    y_t | x_t &\sim N(\gamma x_t, \sigma^2) ,
\end{align}
\]

where the parameters are $\theta = \{\mu, \sigma^2, \gamma\} =$ $\{0.9, 0.2, 0.01\}$. We use $\Theta = \Theta_1 \times \Theta_2 = [-1, 1] \times [0, 2]$, $T = 250$ time steps, $N = 1000$ particles, $K = 300$ iterations and the initial parameter $\theta_1 = \{0.5, 0.5\}$. We implement the SPSA algorithm as suggested by Spall [1998] using the recommended settings for the parameters $\alpha, \gamma$ and $C$. We manually tune the parameters $\alpha = 0.03$ and $c = 0.04$ to achieve good performance for our problem.

The GPO algorithm again converges rather quickly after about 50 evaluations of the log-likelihood and returns the parameter estimate $\hat{\theta} = \{0.896, 0.187\}$. The SPSA algorithm converges slower and requires more than 200 evaluations of the log-likelihood to reach the neighbourhood of the true parameters. Even more iterations are required for the estimates to stabilise. This shows, for this particular example, that the GPO algorithm could be a competitive choice for maximum likelihood estimation.

7. CONCLUSIONS

The results in the previous section indicate that the proposed method does not require many estimates of the intractable log-likelihood. This is due to the GP model
that captures the overall structure well and enables an efficient sampling mechanism in the form of the acquisition rule. With this and the comparison with SPSA in mind, we hope that this algorithm shall turn out to be a competitive alternative to more advanced algorithms.

Important future work includes benchmarking of the proposed method, alternative acquisition rules and investigating possibilities for bias-compensation of the log-likelihood estimate. Also, the Gaussian process models can be useful as an alternative to compute the gradient (score function) and negative Hessian (the observed information matrix) of the log-likelihood. Estimating the latter is an important problem in e.g. nonlinear input design, and this approach could decrease the variance in such estimates.

At http://users.isy.liu.se/en/rt/johda87/, we provide source code to reproduce some of the numerical illustrations in this paper.

ACKNOWLEDGEMENTS

The authors would like to thank Prof. Thomas B. Schön, Dr. Carl E. Rasmussen, Roger Frigola and Andrew McHutchon for interesting discussions and suggestions that greatly improved this paper.

REFERENCES