Marginalizing Gaussian Process Hyperparameters using Sequential Monte Carlo

Andreas Svensson\textsuperscript{1}, Johan Dahlin\textsuperscript{2} and Thomas B. Schö"{o}n\textsuperscript{1}

\textsuperscript{1}Department of Information Technology, Uppsala University, Sweden, \{andreas.svensson, thomas.schon\}@it.uu.se
\textsuperscript{2}Department of Electrical Engineering, Linköping University, Sweden, johan.dahlin@liu.se

February 9, 2015

Abstract

Gaussian process regression is a popular method to model data using a non-parametric Bayesian approach. However, the hyperparameters encountered in the Gaussian process prior are often unknown, but they can still have a great influence on the posterior model. This work provides an off-the-shelf method for numerical marginalization of the hyperparameters, thus significantly alleviating the problem. Our method relies on the rigorous framework of sequential Monte Carlo, and is well suited for online problems. We demonstrate its ability to handle high-dimensional problems and compare it to other sampling methods. It is also concluded that our new method is a competitive alternative to the commonly used point estimates, empirical Bayes, both in terms of computational load and its ability to handle multimodal posteriors.

1 Introduction

The Gaussian process (GP) is a non-parametric probabilistic model for modeling an unknown function $f$, based on observed input data $x$ and (noisy) output data $y = f(x)$. No explicit functional form of $f$ is needed, but some assumptions on $f$ are encoded through a mean function $m_\theta(x)$, a covariance function $K_\theta(x, x')$, and their so-called hyperparameters $\theta \in \Theta$. In mathematical terms, the function $f$ is assumed to be distributed according

$$f(x) \sim \mathcal{G}(m_\theta(x), K_\theta(x, x')),$$

an infinite-dimensional Gaussian distribution. See [Rasmussen & Williams (2006)] for a general GP introduction.

The posterior distribution obtained by using the data $(y, x)$ and the prior in (1) is also a GP, which is a result of the conjugacy property inherent in the Gaussian distribution. That is, the prediction of the (noisy) output $y^*$ for the test input $x^*$ is a Gaussian distribution. All in all, the GP offers a flexible tool for probabilistic modeling, where assumptions (such as smoothness and trends) are encoded in the prior.

The GP posterior is often greatly influenced by the prior through the choices of hyperparameters $\theta$. A common approach is to choose $\theta$ as a maximum likelihood estimate of $p(y|x, \theta)$, commonly referred to as empirical Bayes (EB). However, finding a good point estimate may be problematic if the likelihood is multimodal, which, e.g., can happen with few data points. A fully Bayesian approach, where the hyperparameters are marginalized, integrated out, could therefore be beneficial in some situations. The GP posterior obtained by marginalization can be seen as the result of averaging the posterior over the ranges of hyperparameters supported by the data and the prior. A small toy example is given in Figure 1, illustrating the robustness against multimodality and initialization when using marginalization.

\begin{figure}[h]
\centering
\begin{subfigure}{0.49\textwidth}
\centering
\includegraphics[width=\textwidth]{figure1a.png}
\caption{A data set (left) has a multimodal posterior (right).}
\end{subfigure}
\begin{subfigure}{0.49\textwidth}
\centering
\includegraphics[width=\textwidth]{figure1b.png}
\caption{Regression using hyperparameters obtained by EB, with two different initialization points in the optimization routine.}
\end{subfigure}
\begin{subfigure}{0.49\textwidth}
\centering
\includegraphics[width=\textwidth]{figure1c.png}
\caption{Regression with marginalized hyperparameters, using our proposed method.}
\end{subfigure}
\caption{A small GP regression example illustrating the influence of the hyperparameters and the robustness against multimodal hyperparameter posteriors inherent in the proposed approach.}
\end{figure}
1.1 Problem formulation and contribution

The idea in marginalization is to treat \( \theta \) as a random variable with prior \( p(\theta) \) and likelihood \( p(y|x, \theta) \) giving rise to the posterior \( p(\theta|y, x) \propto p(y|x, \theta)p(\theta) \), and then integrate out \( \theta \). For example, the predictive distribution is computed by

\[
p(y^*|x^*, y, x) = \int p(y^*|x^*, y, x, \theta)p(\theta|y, x)d\theta,
\]

where \( N \) (weighted) samples \( \{\omega^{(i)}, \theta^{(i)}\}_{i=1}^{N} \) from \( p(\theta|y, x) \), the predictive distribution \( \hat{p}(y^*|x^*, y, x) \) can be approximated by

\[
\hat{p}(y^*|x^*, y, x) = \sum_{i=1}^{N} w^{(i)} p(y^*|x^*, y, x, \theta^{(i)}),
\]

which unfortunately is analytically intractable. However, with independent samples, this approximation converges to (2) as \( N \to \infty \), according to the weak law of large numbers.

Our contribution is a new method for sampling from the hyperparameter posterior distribution \( p(\theta|y, x) \). Sampling, when used for marginalization, is robust to multimodal hyperparameter posteriors, as illustrated already in the toy example in Figure 1.

Further advantages of the proposed method are a simplified tuning (compared with some other popular alternatives), a competitive computational load, online updating of hyperparameters as the data record grows and the ability to handle problems with many hyperparameters.

We propose a method based on Sequential Monte Carlo (SMC) samplers (Del Moral et al., 2006), a framework closely related to the popular SMC method known as the particle filter (Doucet & Johansen, 2011). SMC samplers have been extensively analyzed and their convergence properties are now well-understood, e.g., Del Moral et al. (2006, 2012), Chopin (2004), Whiteley (2012). It has recently been applied successfully to a number of applications, for instance Bayesian parameter estimation (Nguyen et al., 2013), approximate Bayesian computations (Jasra et al., 2012), medical image analysis (Petersen et al., 2012) and audio signal processing (Basaran et al., 2013).

In detail, we will use an adaptive SMC sampler. We use the adaption with respect to some internal parameters by (Fearnhead & Taylor, 2013), and we also develop an adaption of the number of samples, which possibly might be of interest for other SMC sampler applications as well.

We compare, in a non-trivial high-dimensional problem, our new method to existing state-of-the-art methods. We also apply it to two real-data problems. The first demonstrates that marginalization does not have to be more computationally demanding than EB, not even for large datasets. The second example, which targets a fault detection problem from industry, is possible only with a computationally efficient method for marginalization. Our proposed method (and all examples) are available as Matlab code via the first authors homepage.

1.2 Related work

Several methods for marginalization of the hyperparameters have previously been proposed in the literature. Bayesian Monte Carlo (BMC) was proposed to solve this problem by Osborne et al. (2008). To implement BMC, however, some non-trivial analytical calculations (depending on the hyperparameter prior) and tuning are required, which limits the flexibility. Unless the hyperparameter prior is independent between the dimensions, BMC does not scale well with the dimension.

Slice sampling was proposed by Murray & Adams (2010) as a way to sample hyperparameters \( \theta \) interchangeably with \( f(x^*) \) (\( y^* \) without noise), which indeed is interesting, but solving a slightly different problem. This approach could possibly be adopted to our formulation.

Further, Hamiltonian Monte Carlo has also been proposed; Neal (2010), Saatçi et al. (2010). This Markov chain Monte Carlo (MCMC) method might be cumbersome to setup and tune, but is indeed an alternative.

A simple method is also deterministic griding, used by, e.g., Saatçi et al. (2010). The gridding idea is simple, but suffers from the curse of dimensionality. Another fundamental sampling method is importance sampling, which requires well-designed proposals and is therefore hard to apply to our problem. A remedy to this cumbersome design problem is an adaptive formulation, see Petelin et al. (2014). The simplicity of this algorithm (compared to most other mentioned methods) is appealing, but it appears not to be very efficient for high dimensional problems.

Our proposed approach relies on the SMC sampler framework, which has close connections with the particle learning (PL) framework (Carvalho et al., 2010). PL has been proposed for learning of GP models by Gramacy & Polson (2011). However, Gramacy & Polson (2011) do not target the marginalization of hyperparameters directly, but are making assumptions on conjugate priors and certain model structure. The SMC framework also includes the transition sequences as well as the possibility to formulate the adaptive version (Fearnhead & Taylor, 2013).

Instead of marginalizing the hyperparameters, frequentistic point estimates, EB, may also be used, as discussed by, e.g., Rasmussen & Williams (2006) and MacKay (1999).

2 Sampling hyperparameters using SMC

For the numerical marginalization in (3), we require \( N \) samples, \( \text{particles} \), from the posterior. In this section, we discuss how to make use of particle methods to obtain these \( \hat{\theta}^{(i)} \) by generating a particle system \( \{\hat{\theta}^{(i)}, w^{(i)}\}_{i=1}^{N} \), where \( w^{(i)} \) is the weight of particle \( i \).

We make use of the SMC sampler (Del Moral et al., 2006, Chopin, 2002). The idea is to construct a sequence of probability densities \( \{p_{0}, \ldots, p_{P}\} \), starting in an easy-to-sample diffuse distribution, the prior, and ending up in the posterior. Readers familiar to continuation methods (Richter & DeCarlo, 1983) may recognize the idea. The
particles are then ‘guided’ through the sequence. We detail these two steps in the subsequent sections.

2.1 Constructing a sequence \( \{\pi_0, \ldots, \pi_P\} \)

To construct a sequence \( \{\pi_0, \ldots, \pi_P\} \), we may use the fact that \( p(\theta|y,x) \) depends on the data \((y,x)\). The sequence can be constructed by clustering the data points in \( P \) disjoint batches \( B_n \), and adding them sequentially as \( \pi_n(\theta) \propto p(y|x, \theta) \phi_n p(\theta) \). An example is given in Figure 2 where the sequence evolves over the iterations from a diffuse prior to the posterior in a fairly smooth transition. We refer to this as a data-tempered sequence.

An alternative is a likelihood-tempered transition from the prior to the posterior through a geometric path \( \pi_n(\theta) \propto p(y|x, \theta) \phi_n p(\theta) \), with \( 0 \leq \phi_0 < \cdots < \phi_P = 1 \).

2.2 Guiding the particles

To guide the samples through the smooth sequence \( \{\pi_0, \ldots, \pi_P\} \), we will iteratively apply the three steps weighting, resampling and propagation.

In the weighting, the ‘usefulness’ of each particle is evaluated. To ensure convergence properties the particles can be evaluated as [Del Moral et al. 2006, section 3.3.2]

\[
 w_n^{(i)} = \frac{\pi_n(\theta_n^{(i)})}{\pi_{n-1}(\theta_{n-1}^{(i)})} w_{n-1}^{(i)}. \tag{4}
\]

To avoid that eventually all particles but one have weight 0, the particles have to be resampled based on the particle weights. The idea is to duplicate particle with high weights, and discard particles with low weights. We suggest to use residual resampling, which decreases the variance in the particle system compared to standard multinomial resampling [Chopin 2004].

A MCMC method is used to propagate the particles \( \theta_{n-1} \) from \( \pi_{n-1} \) to \( \pi_n \) by applying a Metropolis-Hastings kernel \( K : \Theta \rightarrow \Theta \) with invariant distribution \( \pi_n \). The procedure of propagating \( \theta_{n-1} \) (a sample of \( \pi_{n-1} \)) to \( \theta_n \) (a sample of \( \pi_n \)) by the kernel \( K \) can be described as follows:

(i) Use a proposal \( q_h(\theta_{n-1}|\theta) \), e.g., a random walk with variance \( h \), to propose a new sample \( \theta' \). (ii) Adjust for the discrepancy between \( \pi_n \) and \( q_h \) by setting \( \theta_n = \theta' \) with probability

\[
 a_h(\theta_n, \theta') = \min \left\{ 1, \frac{\pi_n(\theta') q_h(\theta_n|\theta')}{\pi_n(\theta_n) q_h(\theta'|\theta_n)} \right\}, \tag{5}
\]

and otherwise \( \theta_n = \theta_{n-1} \). This can be repeated \( K \) times to update \( \theta_n \) for each \( n \), to increase the mixing. For this, the notation \( \theta_{n-1} = \theta_0 \rightarrow \theta_1 \rightarrow \cdots \rightarrow \theta_K = \theta_n \) is used.

We now have a machinery, an SMC sampler, to obtain samples from the hyperparameter posterior, as illustrated in Figure 3. Figure 3 shows the suitability to online applications is clear: if another data point is added, the sequence can easily be extended to \( \pi_n \) including the new data point as well, and only the transition from \( \pi_3 \) to \( \pi_4 \) needs to be performed.

(a) The densities in Figure 2 with particles obtained from the SMC sampler.

(b) GP regression with marginalized hyperparameters from the corresponding posterior, obtained as a by-product of the data-tempered transition. The samples used are the one shown in Figure (a) right above. From left to right, 0 data points (i.e., the prior), 3 data points, 6 data points, and 9 data points. As we formulated the problem, only the rightmost figure is of interest. This illustrates, however, how this method can be used in online problem in a natural way.

Figure 2: A data-tempered transition from the prior \( p(\theta) \) to the posterior \( p(\theta|y,x) \) for the data in Figure 1 obtained by adding 3 data points in each step to the likelihood.

Figure 3: Illustration of the SMC sampler, as it evolves from the prior (no data) to the posterior (all data).

2.3 Adaptive SMC sampler

The performance of our proposed method relies on some user-design choices, e.g., the proposal \( q_h \). To decrease the number of design choices, an extension is added to automatically adapt the parameter \( h \) in the proposal [Fearnhead & Taylor 2013].

The aim is to increase the mixing of \( K \) by adapting \( h \) such that a measure of the mixing, \( \Lambda_n \), is maximized. A common measure of the mixing is the expected square jumping distance given by

\[
 \Lambda_n = \frac{1}{K} \sum_{k=0}^{K-1} \| \theta_k - \theta_{k+1} \|^2, \tag{6}
\]

describing how efficiently the Markov chain explores \( \Theta \). The increased mixing of \( K \) is achieved by sampling \( h \) as if it was a part of \( \pi_n \) by introducing the construction

\[
 \tilde{\pi}_n(h) \propto \sum_{i=1}^{M} f(\lambda_i) R(h - h_n^{(i)}), \tag{6}
\]

where \( R(h - h_n^{(i)}) \) is a density in \( h \) centered around \( h_n^{(i)} \), and \( f \) is a function for weighting of \( h_n^{(i)} \). Furthermore, \( \Lambda_n \)

\footnote{The formulation by [Fearnhead & Taylor 2013] concerns both the choice of proposal structure and the proposal parameter tuning. For brevity, we will only focus on the parameter tuning.}
denotes a Rao-Blackwellized estimate of $\Lambda_n$ given by

$$\hat{\Lambda}_n = \frac{1}{K} \sum_{k=0}^{K-1} \alpha_h(\theta_n^{k,(i)}, \theta_n^{k,(i)}) \Lambda_n(\theta_n^{k,(i)}, \theta_n^{k,(i)})$$

(7)

where $\theta_n^{k,(i)}$ is the proposed particle from $\theta_n^{k,(i)}$. As suggested by Fearnhead & Taylor (2013, Section 3), a linear function $f(\hat{\Lambda}) = c + \hat{\Lambda}$, $c \geq 0$, is used.

The adoption of this method in this paper does not fulfill all assumptions of the asymptotic convergence proof in Fearnhead & Taylor (2013), but we have not experienced any practical problems related to this.

Algorithm 1 Hyperparameter posterior sampler

Input: Data $(y, x)$, covariance and mean functions, and hyperparameter prior $p(\theta)$.

Output: $N$ samples $\{\theta(i)\}_{i=1}^N$ from $p(y|x, \theta)p(\theta)$.

All statements with $i$ are for $i = 1, \ldots, N$.

1: Define $\pi_0(\theta) = p(y_n \mid x_n, \theta)p(\theta)$ by clustering the data in $P$ batches $B_1, \ldots, B_P$.

2: Sample $\theta(0)^{(i)}$ from $p(\theta)$ ($= \pi_0(\theta)$).

3: Sample $h_{10}^{(i)}$, e.g., uniformly on $[0,1,1]$, and attach one to each particle randomly.

4: for $n = 1$ to $P$ do

5: Update weights according to (4).

6: if resampling needed then

7: Sample $a_{n1}^{(i)}$ from a categorical distribution $\propto \{w_{n,j}^{(i)}\}_{j=1}^N$ and set $\theta_n^{(i)} = \theta_{n-1}^{(i)}$ and $w_n^{(i)} = \frac{1}{N}$.

8: end if

9: for $k = 1$ to $K$ do

10: Propose $\theta(i)^{(k)}$ from $q_{h_{n-1}^{(i)}}(\theta(i)^{(k-1)}, \theta(i))$.

11: Set $\theta_n^{(i)} = \theta(i)$ with prob. $\alpha(\theta_n^{(i-1)}, \theta(i))$.

12: end for

13: Sample $h_{nn}^{(i)}$ from (6) and attach one to each particle randomly.

14: end for

User design choices:

- See Section 2.1 for alternative choices.
- Sample $h$ from an interval about the ‘size’ of $p(\theta)$.
- To determine need of resampling, the effective sample size from the particle filter literature can be used, or use the method proposed in Section 2.6.
- $q_{h}(\theta(0)) \sim \mathcal{N}(\theta, h)$ is a simple standard choice.
- See Fearnhead & Taylor (2013) for general comments on the choice of $f$ and $K$.

2.4 Final algorithm

We now have all pieces to sample from the posterior distribution $p(\theta | y, x)$. To summarize, the proposed approach is given as Algorithm 1. A Matlab implementation of the algorithm, compatible with the gpml toolbox (Rasmussen & Williams 2006), is provided via the first authors homepage.

2.5 Computational aspects

The computational cost of Algorithm 1 is mainly governed by the number of evaluations of the likelihood $p(y|x, \theta)$. The important parameters are the number of samples $N$, number of SMC steps $P$, and number of MCMC-moves per SMC-step $K$. A sampling based likelihood-tempered transition is possible to do with $NPK$ evaluations of $p(y|x, \theta)$, whereas a data-tempered transition requires $2NPK$, since $p(y|x, \theta)$ is evolving for a data-tempered transition.

The ‘optimal’ choice of $N$, $P$ and $K$ is problem dependent. Higher values are typically required if the number of hyperparameters is high and $p(\theta | y, x)$ is multimodal, for instance. Our practical experience suggests that $N \approx 15 - 30$, $P \approx 5 - 10$ and $K \approx 2 - 5$ often works well.

2.6 Adaptation of $N$

To reduce the computational load of Algorithm 1 an online adaptation of $N$ could be beneficial. Such an adaptation has to be based on the information in the currently available particles, not to introduce more computations. Considering Figure 3 in Fearnhead & Taylor (2013) a decrease in the number of particles representing the prior (leftmost) would probably cause a loss of information. However, the posterior (rightmost) could probably be satisfactory described with just a couple of particles. (The smaller number of dots in that figure, compared to the leftmost, is because many particles are stacked on each other.) Based on this intuition, we introduce a measure $m$ of how well-spread and, hence, ‘important’ the $N$ particles $\{\theta(i), w(i)^{N}\}_{i=1}^N$ are, as

$$m = \sum_{i=1}^{N} \frac{1}{w(i)^{(i)}} \frac{1}{N^2},$$

(8)

with $d(\theta(i)) = \min_{j \neq i} g(||\theta(i) - \theta(j)||)$, where $\| \cdot \|$ denotes the Euclidean distance and $g$ is a monotonically increasing function with $g(0) \geq 0$ and $g(t) \to 1$ as $t \to \infty$. For the situation with well-spread particles with similar weights (each particle is 'carrying a lot of information'), $m$ is close to 1. On the opposite, with clustered particles and/or uneven weights (the same information could probably be carried by fewer particles), $m$ is large.

From this, we propose a thresholding scheme for adapting $N$: when $m > m_1$, decrease $N$, and vice versa when $m < m_1$. However, absolute bounds on $N$, say $N_u$ and $N_l$, are also suggested for implementation and variance reasons. We will now adapt a central limit theorem to theoretically justify a varying number of particles.

Theorem 1: Convergence for SMC sampler with adaptive $N$.

Assume the assumptions of Del Moral et al. (2006, Proposition 2) holds. Let $\varphi$ denote an arbitrary $\pi_n$-integrable test function. Further, let $E_{n^\infty} \{ \cdot \}$ denote the expectation under a $N_n$-particle approximation of $\pi_n$ obtained by $n$ iterations of an SMC sampler with an adaptive number of particles, and let $E_{n^{N_n \text{min}}} \{ \cdot \}$ denote the expectation for an SMC sampler with a fix number of particles.
\[ N_{\text{min}} \leq N_n \text{ for all } n. \] It then follows that
\[ (N_n)^{1/2} \left( \mathbb{E}_{\pi_n} [\varphi] - \mathbb{E}_{\pi_n} [\varphi] \right) \leq \]
\[ (N_{\text{min}})^{1/2} \left( \mathbb{E}_{\pi_{N_{\text{min}}}} [\varphi] - \mathbb{E}_{\pi_n} [\varphi] \right) \overset{D}{\to} \]
\[ N\left(0, \sigma^2_{\text{SMC},n}(\varphi)\right) \quad \text{as } N_{\text{min}} \to \infty \]
(9)

(9) (and, hence, \( N_n \to \infty \))

where \( \sigma^2_{\text{SMC},n} \) is given by Del Moral et al. (2006, eq. (37)).

Proof: For \( N_n = N_{\text{min}} \) (for all \( n \)), the statement is equivalent to Del Moral et al. (2006, Proposition 2). Revisiting the proof by Chopin (2004, Appendix A.1), neither the induction structure nor the separate Lemmas require \( N_n = N_{n-1} \) etc, and the inequality follows.

3 Numerical example with synthetic data

We consider three examples to demonstrate the benefits of our proposed approach. In this section, we compare with four common methods on a synthetic example and then provide two examples with real-world data in the subsequent section.

In the synthetic example, we focus on the performance of different sampling methods, as well as the sensitivity to initialization for EB, and compare it to our proposed method. We consider a small regression problem of 5 data points (with one-dimensional input), an affine mean function (2 hyperparameters), a covariance function being a sum of a constant, a squared exponential and noise covariance function (4 hyperparameters) and a Gaussian measurement noise (1 hyperparameter). The hyperparameter prior is a Gaussian.

For random sampling methods, the variance between consecutive runs of the same algorithm is a performance measure; it is good if repeated runs of the same algorithm (with the random seed as the only difference) give similar results.

We apply Algorithm 1 Bayesian Monte Carlo (BMC) (Osborne et al., 2008), adaptive importance sampling (AIS) (Petelin et al., 2014), and (deterministic) griding, for marginalization of the hyperparameters.

The results for 15 runs are presented in Figure 4. The variations between the runs is clearly decreasing faster (as a function of computational time) for Algorithm 1 than the comparable methods. For problems with lower dimensionality, the AIS and griding might be competitive methods, but they are suffering from the curse of dimensionality. We have not managed to obtain competitive results with BMC for any problem size, and it appears to be very sensible to its tuning parameter. It should, however, be noted that the computational load of BMC can be substantially decreased if the hyperparameter prior is independent between the dimensions (which was not utilized here).

As a reference, also results for the conceptually different EB is presented in Figure 5. The initialization for the optimization algorithm was drawn from the hyperparameter prior. Although it is a deterministic method, it is clearly very sensitive to the initialization. To robustify against the initialization, the algorithm can be run several times with different initialization (as we have done). That would, however, cause a high computational load, whereas our proposed method is a much less initialization-sensitive alternative.

![Figure 4: Comparison between 15 runs of Algorithm 1 (SMC), Bayesian Monte Carlo (BMC), adaptive importance sampling (AIS), and griding. The predictions (mean, solid line, and 3 standard deviations, dashed line) are shown, together with the data points (red dots). There are 7 underlying hyperparameters. The number of particles/samples/grid points is denoted by \( N \), while \( K \) and \( P \) are algorithm specific tuning. The mean computation time (for a standard desktop computer) is shown. All axis are equally scaled.](image-url)
4 Numerical examples with real-world data

We now turn to two industry-relevant applications with real data. First, we use a benchmark data set to illustrate that marginalization using Algorithm 1 indeed is a competitive alternative to EB, even for large data sets with high-dimensional hyperparameter spaces. We then show how we can make use of our solution within a GP-based online change point detection algorithm. The application requires marginalization of the hyperparameters, so an efficient hyperparameter posterior sampler is indeed a key enabler for this. The online nature of the problem also fits well to the possibility to update the samples in Algorithm 1 online, as discussed in Section 2.2.

4.1 Learning a 7 dof robot arm model

We consider the problem of learning the inverse dynamics of a seven degrees-of-freedom (dof) SARCOS anthropomorphic robot arm, a problem introduced by Vijayakumar & Schaal (2000), and used as a benchmark by Rasmussen & Williams (2006). The problem is to model the mapping from 7 joint positions, velocities and accelerations (in total 21 input variables) to joint torques. We use the same setup as reported by Rasmussen & Williams (2006, Section 2.5 and 8.3.7). We consider only the mapping to the first joint torque. To this end we use the squared exponential covariance function with separate length-scale for each input dimension, and a Gaussian distributed noise. This gives the non-trivial setting involving 23 hyperparameters.

The data set consists of 44,484 training data points and 4,449 test data points. To handle the size of the data set, we use two of the methods described by Rasmussen & Williams (2006): (i) Subset of datapoints, simply selecting m training data points at random and ignore the rest, and (ii) Subset of regressors, described by Rasmussen & Williams (2006, Section 8.3.1). To demonstrate the proposed method, we marginalize the hyperparameters by sampling them from the posterior of a subset with m data points using Algorithm 1 instead of using EB.

For the sake of comparability, we have also reproduced the results from Rasmussen & Williams (2006) for our particular random choice of the m data points. The results are reported in Table 1. For Algorithm 1, N = 15 particles, P = 20 transition steps and K = 5 MCMC steps were used. The priors of the logarithms of the length-scale and the signal variance hyperparameters are N(3, 3) (i.e., fairly vague), and the prior for the logarithm of the noise variance hyperparameter is N(1, 1).

Table 1 presents the standardized mean square error (SMSE, the mean square error divided by the variance of the test outputs; only dependent on the predicted mean) and the mean square logarithmic loss (MSLL, a measure also considering the predictive variances). MSLL is approximately 0 if the predictions are not better than the mean and variance of the training data, and negative otherwise. Numerical problems were experienced for subset of regressors (Rasmussen & Williams 2006, eq. (8.15)) for large m, for both methods. Therefore * indicates runs were only one (out of 10) obtained hyperparameters resulted in numerically invertible matrices, and no interval could be reported.

Table 1 indicate no significant difference between the performance of our method and EB. The reason is probably that the posterior is fairly well-peak for this problem and the particles are close to the point estimate. It is, however, interesting to note that the computational load of our proposed method is of the same magnitude as the EB: approximately 400 s for m = 2048 using the provided Matlab implementation of Algorithm 1 and the optimization routine provided by the gpml package (Rasmussen & Williams 2006) on a standard desktop computer. As Algorithm 1 makes an equally good job in finding relevant hyperparameters as the EB optimization, it is a confirmation that our proposed method is not only applicable, but also a good alternative even for large problems.

4.2 Fault detection of oxygen sensors

We consider data from the wastewater treatment plant Käppalaverket, Sweden. An oxygen sensor measures the dissolved oxygen (in mg/l) in a bioreactor, but the sen-

Table 1: Mean and standard deviations over 10 runs for SMSE and MSLL for the SARCOS data using two different approximations to handle the size of the data set.

| Method | m | SMSE (×10⁻²) | MSLL |
|--------|---|-------------|------|
| EB     | 256 | 8.36 ± 0.80 | -1.38 ± 0.04 |
| SMC    | 256 | 8.10 ± 1.32 | -1.38 ± 0.56 |
| EB     | 512 | 6.36 ± 1.13 | -1.51 ± 0.05 |
| SMC    | 512 | 6.13 ± 0.91 | -1.49 ± 0.04 |
| EB     | 1024 | 4.31 ± 0.16 | -1.66 ± 0.02 |
| SMC    | 1024 | 4.54 ± 0.33 | -1.61 ± 0.03 |
| EB     | 2048 | 2.99 ± 0.08 | -1.78 ± 0.03 |
| SMC    | 2048 | 3.33 ± 0.28 | -1.69 ± 0.06 |
| EB     | 256 | 3.67 ± 0.17 | -1.63 ± 0.02 |
| SMC    | 256 | 3.55 ± 0.28 | -1.65 ± 0.05 |
| EB     | 512 | 2.77 ± 0.44 | -1.79 ± 0.07 |
| SMC    | 512 | 2.89 ± 0.20 | -1.77 ± 0.03 |
| EB     | 1024 | 2.03 ± 0.11 | -1.95 ± 0.03 |
| SMC    | 1024 | 2.00* | -1.95* |

© The SARCOS data set is available at http://www.gaussianprocess.org/gpml/data/
sor becomes clogged because of suspended cleaning, see Figure 6. The identification of such events is relevant to the control of wastewater treatment plants [Olsson et al., 2014]. We apply the GP-based online change point detection algorithm by Saatçî et al. (2010), where the hyperparameters are marginalized using our proposed method. Figure 6: Measurements of dissolved oxygen (in mg/l) in a bioreactor with a sampling period of 15 minutes. The regular cleaning of the sensor was suspended for a period, and it was gradually clogged. After slightly more than 7 days, the sensor was cleaned.

The GP-based change point detection presented by Saatçî et al. (2010) can be summarized as follows: If data \( y_{1:T} \) undergoes changes at time \( r \), it is of interest to (online) detect \( r \), i.e., estimate \( p(r | y_{1:t}) \). The algorithmic idea is a recursive message passing scheme, updating the probability \( p(r_t, y_{1:t}) \), where \( r_t \in \{1, \ldots, t\} \) is the last change point at time \( t \):

\[
\gamma(r_t) \quad t = p(r_t, y_{1:t}) = \sum_{r_{t-1}} p(r_t, r_{t-1}, y_{1:t}) = \sum_{r_{t-1}} \text{prior of } r \cdot \text{GP prediction} \cdot \gamma(r_{t-1}). \tag{10}
\]

Although it might not be entirely clear from this very compact description, this algorithm has an online nature. For doing the predictions using a GP model, the hyperparameters either have to be fixed across all data segments, or marginalized. As we will see, it is not relevant to use fixed hyperparameters for this problem, and an efficient sampling algorithm is therefore a key enabler in solving this problem. As the consecutive predictions \( p(y_t | r_{t-1}, y_{r_t:t-1}) \) and \( p(y_{t+1} | r_{t-1}, y_{r_t:t}) \) are both needed, our approach fits this problem well, as discussed in Section 2.2. We also used an adaptive number of particles (Section 2.6), between 5 and 40, with \( q(t) = \frac{1}{3}(1 + 2 \tanh(4t)) \), \( m_l = 1.3 \) and \( m_u = 100 \). On average, sampling the hyperparameters, i.e., one run of Algorithm 1 took 0.47 seconds on a standard desktop computer.

Our results are presented in Figure 7(a). From the data collection, we can expect two change points; the suspension and the resuming of the cleaning of the sensor, which indeed are indicated in the result. An interpretation of the result might be given by converting the obtained probabilities to point estimates by thresholding, and looking at the GP regression for each individual segment, see Figure 7(b).

Note the data-driven nature of the algorithm, as no explicit model of the sensor was used at all. The tuning parameters are the covariance and mean functions, the prior of the change points (taken to be geometrically distributed), and the hyperparameter priors; a mean around 2, a length scale of hours, etc. Also note the different hyperparameters for the different segments in Figure 7(b), illustrating why it is relevant not to fix, but marginalize, them.

![Figure 7: Results for the GP-based change point detection.](image-url)
5 Conclusion

We have proposed a new off-the-shelf method to sample from the hyperparameter posterior in GP problems, which can be used for numerical marginalization of the hyperparameters. Our method relies on the rigorous SMC framework (Del Moral et al., 2006), and it is well suited for online problems. Through several examples, we have demonstrated its ability to handle high-dimensional problems, and its advantages over other sampling methods. We have also shown that it is a competitive alternative also to the commonly used – but theoretically questionable – EB, both in terms of computational load and in ability to handle problems with multimodal posteriors.

There are more developments to the SMC sampler framework than we have mentioned, e.g., Nguyen et al. (2014), Jasra et al. (2008). As a future direction, such improvements could be incorporated into Algorithm 4 and possibly improve its performance. It would also be interesting to analyze the proposed adaption of $N$ (Section 2.6) further, as well as applying our method to a challenging GP optimization problem (Osborne et al., 2009), such as identification of dynamical systems (Dahlin & Lindsten, 2014).

Acknowledgments

This work was supported by the project Probabilistic modeling of dynamical systems (Contract number: 621-2013-5524) funded by the Swedish Research Council (VR) funded by the Swedish Research Council (VR). We would also like to thank Oscar Samueksson and Dr. Jesús Zambrano for providing the sensor data in Section 4.2.

References

Basaran, Dogac, Cemgil, A Taylan, and Anarim, Emin. SMC samplers for multiresolution audio sequence alignment. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 201–205, Vancouver, Canada, May 2013.

Carvalho, Carlos M, Johannes, Michael S, Lopes, Hedibert F, Polson, Nicholas G, et al. Particle learning and smoothing. Statistical Science, 25(1):88–106, 2010.

Chopin, Nicolas. A sequential particle filter method for static models. Biometrika, 89(3):539–552, 2002.

Chopin, Nicolas. Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. Annals of Statistics, 36(6):2385–2411, 2004.

Chopin, Nicolas, Jacob, Pierre E, and Papaspiliopoulos, Omiros. SMC$^2$: an efficient algorithm for sequential analysis of state space models. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 75(3):397–426, 2013.

Dahlin, Johan and Lindsten, Fredrik. Particle filter-based Gaussian process optimisation for parameter inference. In Proceedings of the 19th IFAC World Congress, pp. 8675–8680, Cape Town, South Africa, August 2014.

Del Moral, Pierre, Doucet, Arnaud, and Jasra, Ajay. Sequential Monte Carlo samplers. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 68(3):411–436, 2006.

Del Moral, Pierre, Doucet, Arnaud, Jasra, Ajay, et al. On adaptive resampling strategies for sequential Monte Carlo methods. Bernoulli, 18(1):252–278, 2012.

Doucet, Arnaud and Johansen, Adam M. A tutorial on particle filtering and smoothing: Fifteen years later. In Crisan, D. and Rozovsky, B. (eds.), Nonlinear Filtering Handbook, pp. 656–704. Oxford University Press, Oxford, 2011.

Fearnhead, Paul and Taylor, Benjamin M. An adaptive sequential Monte Carlo sampler. Bayesian Analysis, 8(2):411–438, 2013.

Gramacy, Robert B and Polson, Nicholas G. Particle learning of Gaussian process models for sequential design and optimization. Journal of Computational and Graphical Statistics, 20(1), 2011.

Jasra, Ajay, Doucet, Arnaud, Stephens, David A, and Holmes, Christopher C. Interacting sequential Monte Carlo samplers for trans-dimensional simulation. Computational Statistics & Data Analysis, 52(4):1765–1791, 2008.

Jasra, Ajay, Singh, Sumeetpal S, Martin, James S, and McCoy, Emma. Filtering via approximate Bayesian computation. Statistics and Computing, 22(6):1223–1237, 2012.

MacKay, David J C. Comparison of approximate methods for handling hyperparameters. Neural computation, 11(5):1035–1068, 1999.

Murray, Iain and Adams, Ryan P. Slice sampling covariance hyperparameters of latent Gaussian models. In Advances in Neural Information Processing Systems (NIPS) 23, pp. 1732–1740, Vancouver, Canada, December 2010.

Neal, Radford M. MCMC using Hamiltonian dynamics. In Brooks, S., Gelman, A., Jones, G., and Meng, X-L. (eds.), Handbook of Markov Chain Monte Carlo. Chapman & Hall/ CRC Press, 2010.

Nguyen, Thi Le Thu, Septier, François, Peters, Gareth W, and Delignon, Yves. Bayesian model selection and parameter estimation in penalized regression model using SMC samplers. In Proceedings of the 21st European Signal Processing Conference (EUSIPCO), pp. 1–5, Marrakech, Morocco, September 2013.

Nguyen, Thi Le Thu, Septier, François, Peters, Gareth W, and Delignon, Yves. Improving SMC sampler estimate by recycling all past simulated particles. In IEEE Workshop on Statistical Signal Processing (SSP), pp. 117–120, Gold Coast, Australia, June 2014.
Olsson, G, Carlsson, B, Comas, J, Copp, J, Gernaey, K V, Ingildsen, P, Jeppsson, U, Kim, C, Rieger, L, Rodriguez-Roda, Ignasi, Steyer, J-P, Takács, I, Vanrolleghem, P A, Vargas Casillas, A, Yuan, Z, and Åmand, L. Instrumentation, control and automation in wastewater – from London 1973 to Narbonne 2013. *Water Science and Technology*, 69(7):1373–1385, 2014.

Osborne, Michael A, Roberts, Stephen J, Rogers, Alex, Ramchurn, Sarvapali D, and Jennings, Nicholas R. Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes. In *Proceedings of the 7th international conference on information processing in sensor networks*, pp. 109–120, St. Louis, MO, USA, April 2008.

Osborne, Michael A, Garnett, Roman, and Roberts, Stephen J. Gaussian processes for global optimization. In *3rd international conference on learning and intelligent optimization (LION3)*, pp. 1–15, Trento, Italy, January 2009.

Petelin, Dejan, Gašperin, Matej, and Šmíd, Václav. Adaptive importance sampling for Bayesian inference in Gaussian process models. In *Proceedings of the 19th IFAC World Congress*, pp. 5011–5015, Cape Town, South Africa, August 2014.

Petersen, Kersten, Ganz, Melanie, Mysling, Peter, Nielsen, Mads, Lillemark, Lene, Crimi, Alessandro, and Brandt, Sami S. A Bayesian framework for automated cardiovascular risk scoring on standard lumbar radiographs. *IEEE Transactions on Medical Imaging*, 31(3):663–676, 2012.

Rasmussen, Carl Edward. and Williams, Christopher K. I. *Gaussian processes for machine learning*. MIT Press, Cambridge, MA., 2006.

Richter, Stephen L and DeCarlo, Raymond A. Continuation methods: Theory and applications. *IEEE Transactions on Systems, Man and Cybernetics*, 13(4):459–464, 1983.

Saatçi, Yunus, Turner, Ryan D, and Rasmussen, Carl E. Gaussian process change point models. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pp. 927–934, Haifa, Israel, June 2010.

Vijayakumar, Sethu and Schaal, Stefan. Locally weighted projection regression: Incremental real time learning in high dimensional space. In *Proceedings of the 7th International Conference on Machine Learning (ICML)*, pp. 1079–1086, Stanford, CA, USA, June 2000.

Whiteley, Nick. Sequential Monte Carlo samplers: error bounds and insensitivity to initial conditions. *Stochastic Analysis and Applications*, 30(5):774–798, 2012.