Fast approximate Bayesian computation for estimating parameters in differential equations

Sanmitra Ghosh · Srinandan Dasmahapatra · Koushik Maharatna

Received: 19 August 2015 / Accepted: 22 February 2016 / Published online: 26 March 2016
© Springer Science+Business Media New York 2016

Abstract Approximate Bayesian computation (ABC) using a sequential Monte Carlo method provides a comprehensive platform for parameter estimation, model selection and sensitivity analysis in differential equations. However, this method, like other Monte Carlo methods, incurs a significant computational cost as it requires explicit numerical integration of differential equations to carry out inference. In this paper we propose a novel method for circumventing the requirement of explicit integration by using derivatives of Gaussian processes to smooth the observations from which parameters are estimated. We evaluate our methods using synthetic data generated from model biological systems described by ordinary and delay differential equations. Upon comparing the performance of our method to existing ABC techniques, we demonstrate that it produces comparably reliable parameter estimates at a significantly reduced execution time.

Keywords Approximate Bayesian computation · Gaussian process regression · Non-linear differential equations · Non-parametric Bayesian · Sequential Monte Carlo

1 Introduction

The time evolution of the variables modelled in a variety of science and engineering branches are often described by ordinary differential equations that are characterised by model structure—the functions of the dynamical variables—and model parameters. The task of estimating these parameters from experimental observations is thus of paramount importance. It is also necessary in some cases to choose the most appropriate among competing models that describe the observations. For parameter estimation and model selection, statistical and pattern recognition techniques built upon a Bayesian framework have been shown to work extremely well for complex non-linear ordinary differential equations (ODE) in Calderhead et al. (2008), Dondelinger et al. (2013), Wang and Barber (2014) and Vyshemirsky and Girolami (2008).

To apply Bayesian techniques we need to integrate marginal likelihoods, which can be computationally intractable in non-linear differential equation models. For this reason some form of approximation such as Monte Carlo integration is generally preferred for parameter inference. Approximate Bayesian computation (ABC) based on sequential Monte Carlo (SMC) is one such approximate inference technique that has been applied to different classes of dynamical systems described by deterministic or stochastic differential equations for both parameter estimation and model selection in Toni et al. (2009). The ABC-SMC algorithm has been shown to work well for the examples considered in Toni et al. (2009). ABC-SMC produces reliable estimates of parameters and has been used to discriminate between a set of candidate models using Bayesian model selection criteria. Moreover, ABC-SMC enables the calculation of parameter sensitivities (Toni et al. 2009).

ABC methods prove to be most useful for large models with complex likelihood surfaces that are difficult to evaluate. The operating principle of ABC methods lies in replacing the evaluation of likelihoods with a simulation based procedure for inference, by using a generative model $M_\theta$ with parameters $\theta \in \Theta$ drawn from a prior distribution $\pi(\theta)$ to simulate observations $Y^i \sim M_\theta$ that are compared with the observed data $Y^d \in D$. If the likelihood $f(Y^d|\theta)$ of observed
data \( Y^d \) is intractable or infeasible to compute, then we can use the ABC algorithm to obtain samples from the following modified posterior density on \( \Theta \times D \)

\[
p_e (\theta, Y^s | Y^d) = \frac{\pi(\theta) \mathbb{1}_{A_{\epsilon,Y^d}} (Y^s) f(Y^s | \theta)}{\int_{A_{\epsilon,Y^d}} \mathbb{1}_{\Theta} \pi(\theta) f(Y^s | \theta) d\theta dY^s} \tag{1}
\]

where \( \epsilon > 0 \) is a tolerance level, \( \mathbb{1}_{B}(\cdot) \) is the indicator function of a given set \( B \) and \( p_e (\theta, Y^s | Y^d) = p(\theta, Y^s | \Delta(Y^d, Y^s) \leq \epsilon) \) where \( \Delta \) is a distance function. The set \( A_{\epsilon,Y^d} \) is the set of simulated observations close to the true observation in the sense of the distance function \( \Delta \). This set is defined as:

\[
A_{\epsilon,Y^d} = \{ Y^s \in D | \Delta(Y^d, Y^s) \leq \epsilon \}.
\tag{2}
\]

A good (enough) approximation of the true marginal posterior distribution is obtained when the distance \( \Delta(Y^d, Y^s) \) is within a predetermined small tolerance \( \epsilon \), i.e.,

\[
p_e (\theta | Y^d) = \int_{Y^s} p_e (\theta, Y^s | Y^d) dY^s \approx p(\theta | Y^d) \tag{3}
\]

Since ABC (including ABC-SMC) requires the generation of a number of simulated observations \( Y^s \sim M_\theta \), the generation of observations could be a computationally expensive process. Thus although ABC-SMC mitigates the intractability of evaluating the likelihood function through simulation, repeated simulation from complex models for inference can itself be burdensome. For the case of dynamical systems such simulations require explicit numerical solutions of non-linear differential equations. Thus, despite its attractive features the ABC-SMC algorithm suffers from a major drawback rooted in its computational burden for inference in differential equations. In particular, the acceptance criterion \( \mathbb{1}_{\Delta(Y^d, Y^s) \leq \epsilon} \) can lead to the generation of many unused simulations \( Y^s \sim M_\theta \), and various methods have been proposed in Filippi et al. (2013) to improve the acceptance rate and reduce the run-time of the algorithm.

In this paper we propose an alternate method of speeding up the ABC-SMC algorithm for parameter estimation in deterministic models described by ordinary differential equations (ODE) or delay differential equations (DDE) by reducing the time incurred in simulation. We achieve this speedup by: (i) completely circumventing the process of integrating the differential equation by operating on the derivative space and (ii) by smoothing the derivatives using Gaussian processes (GP). It should be noted that using Gaussian processes as functional emulators in the derivative space, as a concept, has been proposed in Calderhead et al. (2008), Dondelinger et al. (2013) for speeding up parameter estimation in deterministic differential equations. For parameter estimation, GP-based gradient matching has been used for ODEs and DDEs using a population Monte Carlo sampling (Calderhead et al. 2008); an adaptive variant of this approach is proposed for ODEs (Dondelinger et al. 2013). See Wang and Barber (2014) for a review and comparison between these approaches. The novelty of our proposed method is the fusion of GP regression with ABC-SMC. Our algorithm for fast parameter estimation can be easily incorporated into methods for model selection and recovering parameter sensitivities for deterministic differential equations.

This paper is organised as follows: in Sect. 2 we will introduce the ABC-SMC algorithm for parameter estimation in differential equations. In Sect. 3 we will show how the algorithm can be sped up by circumventing the need for the numerical solution of the differential equations. In Sect. 4 we will introduce GP for function estimation and subsequently in Sect. 5 we will use GPs within the ABC-SMC to operate on the derivative space. In Sect. 6 we will compare the performance of our proposed modification with the ABC-SMC algorithm (Toni et al. 2009) and also with its improved variant proposed in Filippi et al. (2013). We will discuss the computational aspects of the proposed method in Sect. 7 and in conclusion, we summarise our achievements in this paper and give some indication of future work in Sect. 8.

## 2 The basic ABC framework

ABC methods generally have the following algorithmic form:

1. Sample a candidate parameter vector \( \theta \) from a prior distribution \( \pi(\theta) \) and for each \( \theta \sim \pi(\theta) \), simulate a dataset \( Y^s \sim M_\theta \) from a generative model \( M_\theta \).
2. Compute a distance \( \Delta(Y^s, Y^d) \) between the simulated dataset, \( Y^s \) and the experimental data \( Y^d \). If \( \Delta(Y^d, Y^s) \leq \epsilon \), where \( \epsilon \geq 0 \) is the error tolerance of accepted solutions, then accept \( \theta \) and reject otherwise.

These scheme is repeated until \( N \) parameter values are accepted, which represent a sample from the approximate posterior distribution \( p_e (\theta | Y^d) \). Exact posterior can be obtained from this scheme when \( \epsilon = 0 \).

### 2.1 ABC-SMC for parameter estimation in ODE

If the prior distribution is very different from the posterior distribution, the basic ABC framework is very inefficient as it spends a considerable amount of time sampling from areas of low likelihood in parameter space, which makes the acceptance rate extremely low. In order to improve upon poor acceptance rates and facilitate exploration of the parameter space, ABC algorithms based on the SMC sampling
method were proposed in Sisson et al. (2007), Del Moral et al. (2012) and Sisson et al. (2009) and sequential importance sampling (SIS) in Toni et al. (2009), Beaumont et al. (2009). Toni et al. (2009) applied the ABC algorithm based on SIS for parameter estimation and model selection for a variety of dynamical systems including non-linear ODEs and DDEs, which will also be the focus in this paper. Although all the variants of ABC algorithms that come under the SMC category can potentially be used for inference in dynamical systems, we will specifically focus our attention on the ABC approach as adopted in Toni et al. (2009).

We shall apply ABC-SMC to models of the evolution of state $X(t) = (X_1(t), \ldots, X_K(t))$ that are governed by ODEs or DDEs $\frac{dX(t)}{dt} = f(X(t - t_d), \theta)$, where $t_d$ stands for the time delay in DDEs, with $t_d = 0$ for ODEs, and $\theta$ is a vector of parameter values. We express the integrated solution of the differential equations $X(t, X_{in}; \theta)$ as a map $\psi_t(X_{in}; \theta)$ that generates state trajectories $X(t)$ given a set of parameters $\theta$ and initial conditions $X_{in} \equiv X(t_d \leq t \leq 0)$. To generate the samples $Y^s$, we obtain the solutions $X(t, X_{in}; \theta)$ considering the differential equation as the generative model $\mathcal{M}_\theta$:

$$Y^s \sim \mathcal{M}_\theta \Leftrightarrow Y^s = X(t, X_{in}; \theta). \quad (4)$$

to be used in the ABC framework.

A collection of parameter values, called particles $\theta$ are sampled from the prior $\pi(\theta)$ to instantiate the generative model $\mathcal{M}_\theta$. To decide whether a particular choice of parameters $\theta \sim \pi(\theta)$ is accepted, we need to compare if the simulated trajectory $Y^s \sim \mathcal{M}_\theta$ is within a tolerance level $\epsilon$ of the observed trajectory $Y^d$, for which we introduce the distance function

$$\Delta(Y^d, Y^s) = \sum_{i=1}^L \sum_{k=1}^K (Y^d_k(t_i) - X_k(t_i))^2, \quad (5)$$

where we assumed that the data were collected, and the state evaluated, at discrete time points $t^L \equiv \{t_i\}_{i=1,\ldots,L}$. Note that for dynamical systems such distance functions are generally built by considering the entire time-series data instead of some sufficient statistics.

The sequential stage of this algorithm involves replacing a single tolerance value $\epsilon$ by a sequence of tolerance values $\epsilon_t$, where $\tau = 0, \ldots, S_{MC}$ denotes the sequential steps, and $\epsilon_t > \epsilon_{t+1}$. The particles $\theta_t$ are indexed by $\tau$ labelling the tolerance level, and are sampled from the posterior distribution obtained from the previous sequential step, thus introducing a step-wise procedure for generating parameters from a sequence of more informative distributions, starting at $\tau = 0$ with the prior distribution $\pi(\theta)$. To accept or reject the sampled particles for sequence index $\tau$, the generated trajectories from the model with parameters $\theta_t$ must be closer to the observed data $Y^d$ than those generated from the model with parameters $\theta_{t-1}$ in step $\tau - 1$. The generative mechanism for the particles $\theta_t$ differs from that of $\theta_{t-1}$ in that they are sampled from the $N$ particles $\{\theta_{t-1}^{(i)}\}_{i=1,\ldots,N}$ with importance weights ($\text{Toni et al. 2009}$) $w_{\tau-1}^{(i)}$ and each $\theta^* \sim \{\theta_{t-1}^{(i)}\}_{i=1,\ldots,N}$ is perturbed by a perturbation kernel $K_{\tau}(\theta|\theta^*)$ ($\text{Toni et al. 2009}$).

For each $\tau = 0, \ldots, S_{MC}$, the $N$ particles meeting the acceptance criterion $\Delta(Y^s, Y^d) \leq \epsilon_t$ represent a point-wise approximation for the posterior distribution over the parameter values:

$$p_{\epsilon_t}(\theta|Y^d) \approx \frac{1}{N} \sum_{i=1}^N w_{\tau}^{(i)} \delta(\theta - \theta_{\tau}^{(i)}), \quad (6)$$

where $w_{\tau}^{(i)}$ is the importance weight of the particle $i$ ($\text{Toni et al. 2009}$) and $\delta(\theta - \theta^{(i)})$ is a product of Dirac delta functions, one for each component of $\theta$. For a small value of $\epsilon_{S_{MC}}$ the final collection of particles should be a good point-wise approximation to the true posterior distribution. The ABC-SMC algorithm is listed in Algorithm 1. One way of speeding up the ABC-SMC algorithm is by increasing the acceptance rate. To this end a range of perturbation kernels were proposed in Filippi et al. (2013) that result in a noticeable change in terms of acceptance rates and run-time. However, the major computational burden stems from the numerical integration of the differential equation and thus a faster simulation method within the ABC-SMC is likely to speed up this algorithm possibly more than what is gained by clever choice of perturbation kernels.

### 3 Gradient based parameter estimation in differential equations

We have mentioned previously that the computational bottleneck stems from the explicit integration carried out in each simulation step. In order to avoid the integration one could essentially use a gradient based estimation. If the temporal variations in observations $Y^d(t)$ are believed to be less smooth than the underlying state evolution that is modelled by differential equations, we shall introduce the target state variable $\hat{X}(t)$ to be the smoothed version $\hat{X}(t) \equiv S(Y^d)$ of the observations. Here, $S$ represents any smoothing procedure, and we will use GP regression to perform the smoothing below. In the ABC framework, we shall accept the trajectories $X(t)$ from the model $\mathcal{M}_\theta$ (see (4)) if they are close to $\hat{X}(t)$. Once we have $\hat{X}(t)$ we can compute its numerical derivative to obtain the empirical vector field $V^d(t)$ of the dynamical system $\mathcal{M}_\theta$:

$$\hat{X}(t) \equiv S(Y^d), \quad V^d(t) \equiv \frac{d}{dt} \hat{X}(t). \quad (7)$$
Algorithm 1 ABC-SMC as proposed in Toni et al. (2009)

1. Given \(Y^d, \pi(\theta), M_0\).
2. Initialise \(\epsilon_1 > 0, \tau = 0, \ldots, S_{MC}, \epsilon_\tau > \epsilon_{\tau+1}\). Set \(\tau = 0\).
3. Set \(i = 1\).
4. if \(\tau = 0\) then
5. sample \(\theta^{**}\) independently from \(\pi(\theta)\):
6. \(\theta^{**} \sim \pi(\theta)\).
else
7. from the previous population \(\{\theta^{(j)}_t\} = 1, \ldots, N\)
   sample \(\theta^{*} \sim \{\theta^{(j)}_t\} = 1, \ldots, N\) with associated weights \(w_{\tau}^{(j)}\) and 
   use the perturbation kernel \(K_\tau(\theta|\theta^{*})\) to produce \(\theta^{**} \sim K_\tau(\theta|\theta^{*})\).
8. end if
9. if \(\pi(\theta^{**}) = 0\) then
10. go to 4.
else
12. Simulate a candidate dataset \(Y^t\) from the model \(M_0\) with parameter 
   \(\theta^{**}: Y^t \sim M_\theta|\theta^{**}\).
13. end if
14. if \(\Delta(Y^d, Y^t) \geq \epsilon_\tau\) then
15. go to 4.
else
17. Set \(\theta^{(j)}_t \leftarrow \theta^{**}\) and calculate the weight for particle \(\theta^{(j)}_t\),
   \[w_{\tau}^{(j)} = \begin{cases} 1, & \text{if } \tau = 0 \\ \frac{\pi(\theta^{(j)}_t)}{\sum_{j=1}^N w_{\tau-1}^{(j)} K_\tau(\theta^{(j)}_t|\theta^{(j)}_{\tau-1})}, & \text{if } \tau > 0 \end{cases}\]
18. end if
19. if \(i < N\) then
20. Set \(i \leftarrow i + 1\) and go to 4.
else
22. Normalise the weights.
23. end if
24. if \(\tau < S_{MC}\) then
25. Set \(\tau \leftarrow \tau + 1\) and go to 3.
26. else
27. return particles \(\hat{\theta}^{(j)}_{SMC}\) at \(\tau = S_{MC}\).
28. end if

In addition, while the left hand side of the ODE \(\frac{d}{dt} X(t) = f(X(t), \theta)\) is estimated by the empirical derivative \(V^d(t)\), it should be matched by the model vector field \(f(X(t), \theta)\) on the right hand side, when evaluated on the smoothed state data \(f(X(t) = \hat{X}(t), \theta)\).

Upon introducing a new distance measure between \(V^d(t)\) obtained from the smoothing and \(f(\hat{X}(t), \theta)\) obtained from the vector field we can eliminate the original distance metric for ABC-SMC between observed, \(Y^d\), and the simulated, \(Y^s\), trajectories, thus unburdening ABC-SMC of ODE integration at each simulation step. The gradient based method was first suggested in Varah (1982) where a spline-based smoothing was used to denoise the observed data. In this method a cost function was built using the distance metric in derivative space and optimisation was used to minimise this cost function in order to obtain point estimates. Recent developments of these methods are described in Ramsay et al. (2007). All these approaches suffer from similar problems of using additional regularisation parameters for smoothing and often the estimates are sub-optimal point estimates. Although porting the derivative based distance within an ABC scheme alleviates the computational bottleneck, this approach suffers from an inherent shortcoming that is rooted in obtaining a numerical derivative as this might lead to information loss.

In our approach, we replace the numerical differentiation with a zero mean GP prior on the state \(X(t)\) given by

\[p(X(t)|\phi) \sim GP(0, K(t, t'; \phi)), \tag{8}\]

where \(K(t, t'; \phi)\) denotes a covariance function with hyper-parameters \(\phi\). Once such a prior is established then GP regression techniques can be applied to estimate both the state vector \(\hat{X}(t)\) and also the derivative process \(V^d(t)\). Using GP regression the derivative process can be inferred within a probabilistic framework. Hence, we propose to use a distance function in the derivative space where the state \(\hat{X}(t)\) and derivative \(V^d(t)\) is modelled using GP regression, within the ABC-SMC algorithm. In this way our proposed method is based on the GP construction in the derivative space as in Calderhead et al. (2008), combined with the ABC-SMC algorithm as proposed in Toni et al. (2009).

We like to point out the fact that there is a distinction between the approach of Calderhead et al. (2008) to that of Dondelinger et al. (2013) and Wang and Barber (2014). The former approach is a two step method where in a GP is fitted to the data and then the smoothed state and derivative information are used to sample the ODE parameters from a distribution \(p(\theta|\hat{X}(t), V^d(t))\). Thus the GP inference process has no information about the ODE dynamics. The latter however induces a coupling between the GP hyperparameters and the ODE parameters by sampling the state from a distribution \(p(X(t)|\theta, \phi)\) conditioned on both the GP and ODE hyperparameters. Such a conditioning forces the ODE system dynamics to influence the GP inference. Thus for a very noisy dataset the GP can adapt itself using information from the ODE to better estimate the state trajectories, which in turn improves the ODE parameter inference. Our proposed method is built on the former two step approach as in Calderhead et al. (2008). Next, we will briefly introduce GP regression and will apply this to the ABC-SMC algorithm.

4 Gaussian processes

For real-valued functions \(f : \mathcal{A} \rightarrow \mathbb{R}\) of one or more input variables defined over \(\mathcal{A}\), a Gaussian process (GP) is a Bayesian non-parametric model that specifies a distribution \(p(f)\) (O’Hagan and Kingman 1978; MacKay 1998; Neal 1998; Rasmussen and Williams 2006) characterised by a mean function \(\mu(x)\) and a covariance function or kernel,
\( K(x, x'; \phi) \):

\[
f(x) \sim \mathcal{GP}(\mu(x), K(x, x'; \phi)),
\]

where \( \phi \) are hyperparameters. For example, the squared exponential covariance function, which we use below, is given by

\[
K_{SE}(x, x') = \sigma^2_{\text{ker}} \exp \left( -\frac{1}{2} \frac{(x - x')^2}{l^2} \right),
\]

with hyperparameters \( \sigma^2_{\text{ker}} \) and \( l^2 \) (variance and characteristic length-scale).

For a finite number \( n \) of inputs \( x^* = (x_1, \ldots, x_n) \), \( x_i \in \mathcal{A} \) and for \( f(x) \sim \mathcal{GP}(\mu(x), K(x, x'; \phi)) \), the \( n \)-dimensional vector of function values evaluated at \( n \) points \( f(x^*) \triangleq (f(x_1), \ldots, f(x_n)) \) is a random vector drawn from a multivariate Gaussian distribution:

\[
p(f(x^*)|x^*) = \mathcal{N}(\mu(x^*), K(x^*, x^*)).
\]

For performing regression, observations \( y(x) \triangleq (y(x_1), \ldots, y(x_L)) \) at \( L \) training points \( x = (x_1, \ldots, x_L) \) are fit to function \( f \) evaluated at \( x \):

\[
y(x) = f(x) + \eta,
\]

where \( \eta \sim \mathcal{N}(0, \sigma^2 \mathbb{I}) \). Given the training data \( (x_i, y(x_i)), i = 1, \ldots, L \) the conditional predictive distribution of the function \( f^* \triangleq f(x^*) \) evaluated at the test points \( x^* \) is a Gaussian with mean and variance given by Rasmussen and Williams (2006)

\[
E[f^*|y, x, x^*, \Phi] = \mu(f^*) + [K(x^*, x)(K(x, x) + \sigma^2 \mathbb{I})]^{-1} (y - \mu(f^*))]

\]

\[
\text{Var}[f^*|y, x, x^*, \Phi] = K(x^*, x^*) - [K(x^*, x)(K(x, x) + \sigma^2 \mathbb{I})]^{-1} K(x, x^*)
\]

where \( \Phi = \{\phi, \sigma\} \).

The hyperparameters \( \Phi \) are inferred as point estimates by optimising the logarithm of the marginal likelihood, with a zero mean assumption Rasmussen and Williams (2006):

\[
\log p(y|\Phi) = \log \int p(y|f, \Phi)p(f|\Phi)df = -\frac{1}{2}y^T (K(x, x) + \sigma^2 \mathbb{I})^{-1}y - \frac{1}{2} \log |K(x, x) + \sigma^2 \mathbb{I}| - \frac{L}{2} \log(2\pi).
\]

4.1 Derivative Gaussian processes

Since differentiation is a linear operator, the derivative of a GP is another Gaussian process (Solak et al. 2002). This makes it possible to include derivative observations in the GP model, or to compute predictions about derivatives. We have

\[
E \left[ \frac{\partial f(x)}{\partial x} \right] = \frac{\partial E [f(x)]}{\partial x}.
\]

And likewise the covariance between partial derivative and a function value can be written as

\[
K \left( \frac{\partial f(x)}{\partial x}, f(x^*) \right) = \frac{\partial}{\partial x} K \left( x, x^* \right),
\]

and the covariance between partial derivatives follows

\[
K \left( \frac{\partial f(x)}{\partial x}, \frac{\partial f(x^*)}{\partial x^*} \right) = \frac{\partial^2}{\partial x \partial x^*} K \left( x, x^* \right)
\]

For example considering the squared exponential covariance function given in (10), we can write the covariance between partial derivative and a function value as

\[
P \left( \frac{\partial f(x^*)}{\partial x^*} | f(x) \right) = \mathcal{N}(m, \Sigma),
\]

where we have the mean function \( m \) and covariance function \( \Sigma \) (obtained by using (15) and (16) and considering the prior mean of the test and training function to be zero) given as

\[
m = \frac{\partial K (x^*, x)}{\partial x^*} \left[ K(x, x) + \sigma^2 \mathbb{I} \right]^{-1} f(x),
\]

\[
\Sigma = \frac{\partial^2 K (x^*, x^*)}{\partial x^* \partial x^*} - \frac{\partial K (x^*, x)}{\partial x^*} \frac{\partial}{\partial x^*} \left[ K(x, x) + \sigma^2 \mathbb{I} \right]^{-1} \frac{\partial K (x, x^*)}{\partial x^*}.
\]

5 ABC-SMC with derivative GP

In this section we apply the machinery reviewed in the previous sections to the task of inferring parameters in differential equation models whose solution is the state trajectory \( X(t) \). If we assign a GP prior to the state evaluated at time points...
training data satisfies a differential equation. We can use GP regression to
take the set of values of the state $X(t^L)$ on a Gaussian prior distribution:

$$p \left( X(t^L) | t^L \right) = \mathcal{N}(X(t^L)|0, K(t^L, t^L)).$$  \hspace{1cm} (21)$$

The modelling task is to represent the experimental data as $Y^d = \{X(t^L) + \eta^L\}$ where $\eta^L$ refers to $L$ i.i.d. samples from $\mathcal{N}(0, \sigma^2 I_L)$, here $I_L$ is a $L \times L$ identity matrix, and $X(t)$ satisfies a differential equation. We can use GP regression to obtain the expectation and variance of the posterior (given training data $Y^d$ at $t^L$) state $X(t^*)$ for some test input time point $t^*$ as in Sect. 4 (Rasmussen and Williams 2006):

$$E \left[ X(t^*) | Y^d \right] = K(t^*, t^L)(K(t^L, t^L) + \sigma^2 I_L)^{-1} Y^d,$$
$$\text{Var}[X(t^*)] = K(t^*, t^*) - K(t^*, t^L)(K(t^L, t^L) + \sigma^2 I_L)^{-1} K(t^L, t^*).$$  \hspace{1cm} (22)

This expected posterior state variable $X(t^*)$ for arbitrary choice of $t^*$ models the smoothed evolution of the state $\hat{X}(t)$ introduced above, and where it is assumed that observational noise accounts for deviations from the smoothed time course. The smoothed state estimation enables us to compute the velocity field using the derivative GP (as in Sect. 4.1):

$$E \left[ \frac{d}{dt} X \right] = \partial_t K(t, t)(K(t, t) + \sigma^2 I_L)^{-1} E[X].$$  \hspace{1cm} (23)

This completes the procedure for deriving the empirical velocity field $V^d(t) = E[\frac{d}{dt} X]$.

To apply this derivative process within the ABC framework we need to define a distance metric $\Delta( V^d(t), f(\hat{X}(t), \theta))$ between the smoothed velocity field derived from the observed data, and the velocity field postulated in a differential equation model, where the expected state estimation $\hat{X}(t)$ has been substituted for the state variable. Hence our proposed fast alternative ABC-SMC based on GP gradient distance (GP-ABC-SMC) works as follows:

1. Having given data $Y^d$ as a noisy observation of the true state variable $X(t)$, assign a GP prior on $X(t)$ using (8) and choose a covariance function, with some unknown hyperparameters, needed to define the GP prior.
2. Learn the hyperparameters of the covariance function from the original noisy experimental data $Y^d$ using maximum likelihood estimation and then run GP regression to obtain an estimation of the smoothed state evolution $\hat{X}(t) = E[X]$ using (22) and the experimental time points $t$ as both the training and test input points.
3. Construct the first derivative of the covariance matrix and estimate the derivative process $V^d(t) = E \left[ \frac{d}{dt} X | X = \hat{X} \right]$ using (23).

4. Run the ABC-SMC algorithm with a modified distance metric $\Delta( V^d(t), f(\hat{X}(t), \theta)) \leq \epsilon_\tau$ for tolerance schedule $\{\epsilon_\tau\}$, where at each simulation step the simulated data $Y^s = f(\hat{X}(t), \theta)$ is generated by evaluating the velocity field on the right hand side of the differential equation. This yields the posterior distribution of the parameters $p(\theta | Y^d)$ (6).

Note that no explicit solution of differential equation is required to generate the simulated data within the iterations of the GP-ABC-SMC algorithm. Also note the fact that, in order to run the GP-ABC-SMC algorithm, no knowledge of the initial condition is required.

Note that for the specific case of gradient matching applications, such as the proposed GP-ABC-SMC, if the experimental time points are irregularly spaced then the covariance pertaining to the training points will become singular. In such cases we can choose the training and test time points $t_{\text{equidist}}$, $t_{\text{equidist}}^*$ in (22) and (23) to be equidistant to perform the GP regression, although the experimental time points $t^L$ are irregularly spaced. The resultant state $X(t_{\text{equidist}})$ and derivative $V^d(t_{\text{equidist}})$ trajectories can be used along with the right hand side $f(\hat{X}(t_{\text{equidist}}), \theta)$ to create the distance metric $\Delta( V^d(t_{\text{equidist}}), f(\hat{X}(t_{\text{equidist}}), \theta)) \leq \epsilon_\tau$ without using a singular covariance matrix altogether.

For the sake of conformity with the ABC terminologies, we will persist in using the terms $Y^d$ for observed data and $Y^s$ for simulated data, as before. However, within the context of GP-ABC-SMC algorithm, observed and simulated data refer to $V^d(t)$ and $f(\hat{X}(t), \theta)$ respectively. That is what $Y^d$ and $Y^s$ will refer to for the rest of the paper.

### 5.1 Algorithmic settings

The success of ABC-SMC algorithm both in terms of computational complexity and quality of the solution depends on the choice of the $\epsilon_\tau$ schedule and the perturbation kernel $K_\tau$. In this section we will briefly describe how we have chosen these two algorithmic settings. A detailed discussion concerning the effects of these settings can be found in Filippi et al. (2013).

#### 5.1.1 Tolerance schedule

Until recently, tolerance values were manually tuned in practice based on prior empirical knowledge about the model. An adaptive choice of the tolerance values has been proposed in Del Moral et al. (2012) and Drovandi and Pettitt (2011). In an adaptive schedule the value of the tolerance $\epsilon_\tau$ is chosen as the $\alpha$-th quantile, where $0 \leq \alpha \leq 1$ of the distances between the observed data $Y^d$ and simulated data $Y^s_{\tau-1}$ generated at
the previous algorithmic time. For all our simulation studies we have used this adaptive schedule.

5.1.2 Perturbation kernel

Perturbation kernels hold the key to the acceptance rates in ABC-SMC and the speed of the algorithm as exploited in Filippi et al. (2013). Perturbation kernels can be broadly divided into two categories: a component-wise perturbation kernel and a multivariate perturbation kernel. In a component-wise perturbation kernel $\theta \sim N(\theta, \Sigma_t)$ where $\Sigma_t$ is a diagonal covariance matrix whose diagonal entries $\sigma^2_{\tau,j}$ $j = 1, \ldots, d$ are chosen adaptively according to the previous population labelled by $\tau − 1$ (Beaumont et al. 2009; Didelot et al. 2011; Filippi et al. 2013).

A component-wise perturbation kernel is, by construction, unable to generate particles with correlated components; therefore, for models with strongly correlated parameters the ABC-SMC sample generator will not be able to reflect the structure of the posterior and the acceptance rate will be low. Thus, in order to capture such correlations the particles can be perturbed according to a multivariate normal distribution with a non-diagonal covariance matrix $\Sigma_t$ that depends on the covariance of the particles as reflected in the population in the previous sequential step $(\tau − 1)$ (Filippi et al. 2013). Furthermore, a multivariate perturbation kernel operating on a subset of size $N'$ of the $N$ particles (a local kernel) was also shown (Filippi et al. 2013) to produce a noticeable improvement in the acceptance rate. In order to define this kernel we will introduce some notation. Let $Y_t(i)$ denote the simulated data generated from $\mathcal{M}_i$ with particle $\theta \leftarrow \theta(i)$, $i = 1, \ldots, N$ from a population of size $N$ at algorithmic time $\tau$. The corresponding importance weights are denoted as $w_t(i)$. We collect all such particles (along with the weights) from algorithmic time $\tau − 1$ for which $Y_t(i)$ is not only within distance $\epsilon_{\tau−1}$ of the observed data $Y^d$ but also within distance $\epsilon_\tau$ of it. We denote such particles as $\theta(i)_{\tau−1}$:

$$\left\{\theta(i)_{\tau−1}\right\}_{1 \leq i \leq N'} = \left\{\theta(i)_{\tau−1} | \Delta(Y^d, Y_t(i)) \leq \epsilon_\tau, 1 \leq i \leq N\right\},$$

(24)

with associated normalised weights $\bar{w}(i)_{\tau−1} \triangleq (w(i)_{\tau−1}/\bar{w})$, with $\bar{w} \triangleq \sum_i w(i)_{\tau−1}$.

Having defined the pairs $\left(\theta(i)_{\tau−1}, \bar{w}(i)_{\tau−1}\right)$ we can now use a multivariate normal distribution $\mathcal{N}(\theta(i)_{\tau−1}, \Sigma_t)$, with a local covariance $\Sigma_t$ (termed the optimal local covariance in Filippi et al. 2013), to perturb a particle $\theta(i)_{\tau−1}$, where local refers to particle $i$. This covariance is given by

$$\Sigma_t = \sum_{j=1}^{N'} w(j)_{\tau−1} (\tilde{\theta}(j)_{\tau−1} - \theta(i)_{\tau−1}) (\tilde{\theta}(j)_{\tau−1} - \theta(i)_{\tau−1})^T.$$  

(25)

In the next section we will implement the GP-ABC-SMC algorithm to infer parameters of some standard non-linear differential equations through some toy examples and in that process we will compare and contrast our GP gradient based approach to that of ABC-SMC algorithm with explicit integration.

6 Evaluation of the algorithm on benchmark models

To evaluate the GP-ABC-SMC algorithm we have chosen three benchmarking differential equations: the Lotka Volterra predator–prey model (Murray 2002), the Hes1 loop model (Monk 2003) and signal transduction cascade model (Vyshemirsky and Girolami 2008; Wang and Barber 2014). Each is a set of non-linear differential equations modelling biological systems and show non-trivial dynamical phenomena such as limit cycle oscillations and non-stationary time evolution. For all these examples we have used the distance function given by (5) and have run the ABC-SMC algorithm with explicit integration using a component-wise univariate normal kernel (ABC-SMC-Comp) (Toni et al. 2009) as well as a multivariate normal kernel with the optimal local covariance matrix (ABC-SMC-OLCM) (Filippi et al. 2013). For our proposed GP-ABC-SMC we have also used both the aforementioned perturbation kernels. We refer these as the GP-ABC-SMC and GP-ABC-OLCM respectively. We believe a comparison between these four variants of ABC-SMC is required to capture the difference in speed of execution between the GP based ABC-SMC and the previous approaches reported in (Toni et al. 2009; Filippi et al. 2013), while comparing posterior estimates of the parameters. For all the examples presented here, we ran all these variants of ABC-SMC, including the proposed GP based ones, with $N = 100$ particles using an adaptive tolerance schedule set to the $\alpha = 0.1$ quantile of the distances in the previous populations.

The ABC-SMC routines are written in MATLAB and for the GP regressions the GPML package (Rasmussen and Nickisch 2010) for MATLAB is used in the predator–prey and Hes1 loop example. For the signal transduction cascade model the GPMat toolbox for MATLAB https://github.com/SheffieldML/GPmat is used which has an implementation of the multi-layer perceptron (MLP) covariance kernel (Wang and Barber 2014), required to handle the non-stationarity of some of the state variables. The explicit integrations are carried out using MATLAB’s built in ODE and DDE solver routines.
6.1 ODE: the predator prey model

The Lotka Volterra Murray (2002) model depicts an ecological system that is used to describe the interaction between a predator and prey species. This ODE given by
\[
\begin{align*}
\dot{x} &= \alpha x - xy \\
\dot{y} &= \beta xy - y,
\end{align*}
\tag{26}
\]
shows limit cycle behaviour and has been used for benchmarking in (Toni et al. 2009; Dondelinger et al. 2013). \( \theta = (\alpha, \beta) \) is the set of parameters and \( X(t) = (x(t), y(t)) \) is the state vector comprising the concentrations of the predator and the prey species respectively. To create a realistic dataset we generated 11 uniformly spaced samples between the time interval \( 0 \leq t \leq 10 \) from the model with parameters \( \theta = (1, 1) \) and added random Gaussian noise with zero mean and standard deviation \( \sigma = 0.5 \) to each point. The initial values of the ODE for generating the synthetic data are chosen as \( X(t = 0) = (1.0, 0.5) \). In order to inspect the consistency of our proposed algorithm we created two more datasets obtained by adding two other realizations of the random noise to the ODE time courses. Thus we have three sets of artificial data (denoted as Dataset 1, 2 and 3), each of which has been corrupted by Gaussian noise with zero mean and standard deviation \( \sigma = 0.5 \) and sampled separately. Note that the GP-ABC-SMC algorithm does not require the estimation of additional nuisance parameters related to the initial values. The time evolution of the state and its derivative is predicted through the GP regression as described in Sect. 5.

We have used the squared exponential covariance function given by (10) for the GP regression in this example.

From the synthetic data we perform the task of parameter inference using the four different variants of ABC-SMC discussed in the last section to compare their performance. Both \( \alpha \) and \( \beta \) are chosen from uniform prior distributions \( \mathcal{U}(-10, 10) \) in all cases. The number of algorithmic iterations, the value of \( S_{MC} \) is set to \( S_{MC} = 6 \) for the ABC-SMC-Comp and ABC-SMC-OLCM while it is set to \( S_{MC} = 5 \) for GP-ABC-SMC and GP-ABC-OLCM. The values differ because we have chosen these on the criteria of minimum number of adaptive iteration required for estimating a reliable posterior distribution. As the ABC-SMC with integration and the GP based variants operate on different spaces thus setting same values for \( S_{MC} \) does not produce comparable results. The specific values of \( S_{MC} \) for this and subsequent examples are chosen on the basis of multiple trials of all the four ABC-SMC algorithm on each of the datasets.

The resulting parameter estimates are listed in Table 1 and the evaluation of the performance in Table 2 (top). We show
in Table 1 the mean with 95% confidence intervals of the last population of parameters, approximating the marginal posterior, for each variants of the ABC-SMC. The approximate marginal posterior distributions of each of the two parameters learnt by the GP-ABC-OLCM and the ABC-SMC-OLCM are shown in Fig. 1. The histograms in Fig. 1 are based on the final particle populations generated by the respective algorithms after they are run on Dataset 1. Note that the run-time of the GP-ABC-SMC and the GP-ABC-OLCM algorithms is the sum of the run-time of the ABC and the GP regression (including the estimation of covariance hyperparameters). The value of $\sigma$ is estimated as part of the GP regression. These estimated values are $\sigma = \{0.4752, 0.8090\}, \sigma = \{0.6219, 0.3940\}$ and $\sigma = \{0.6432, 0.4592\}$ for the dataset 1, 2 and 3 respectively.

### 6.2 DDE: the Hes1 model

Our proposed algorithm is also able to estimate parameters of delay differential equations. The Hes1 model system is used in systems biology to provide a simplified account of the oscillatory behaviour of the concentrations $(\mu(t), p(t))$ of a species of mRNA and its corresponding protein. The model, introduced in Monk (2003), is described by the following DDE:

\[
\begin{align*}
\dot{\mu} &= \frac{1}{1 + (p(t - t_d)/p_0)^n} - \mu_m \mu \\
\dot{p} &= \mu - \mu_p p,
\end{align*}
\]  

\hspace{1cm} (27)

where the parameters $\mu_m$ and $\mu_p$ are decay rates, $p_0$ is the repression threshold, $n$ is the Hill coefficient and $t_d$ is the time delay. We generated data from the above model with parameters $\mu_m = 0.03, \mu_p = 0.03, p_0 = 100$ and $t_d = 25$ and initial conditions $\mu(t_0) = 3$ and $p(t_0) = 3$ for the concentrations between the interval $(0 \leq t \leq 300)$ with uniform spacing of $\Delta t = 2$ by numerically solving the DDE. $n$ is fixed at a value of 5 (Monk 2003). We estimated the standard deviations $\sigma_{\mu} = 6.0020$ and $\sigma_p = 121.7670$ of the generated data, for each of the concentrations $\mu(t)$ and $p(t)$.
We then added noise, with standard deviation set to 0.1 times these estimated standard deviations $\sigma_p$ and $\sigma_{\mu}$, to the data to create the artificial datasets. As in the previous example we created three datasets in a similar fashion.

For comparison of performance of the four methods in the parameter estimation task, we keep the same algorithmic settings, as well as the same covariance function for the GP regression as in the previously example. Unlike the ODE case where our algorithm does not need to guess the initial state values, it does need a history function for $X(t \leq 0)$ for DDEs in order to work. In most practical cases the initial history function is taken as a constant function. Thus in order to make our algorithm work, we shifted the first element of the estimated state evolution backward in time to create the history function. The four variants of ABC-SMC having the same settings as before, are applied to this artificial dataset. We chose uniform priors for each of the parameters: $\mu_m \sim U(-2, 2)$, $\mu_p \sim U(-2, 2)$, $p_0 \sim U(0, 200)$ and $t_d \sim U(0, 50)$. The number of iterations are chosen as $S_{MC} = 14$ while running ABC-SMC-Comp and ABC-SMC-OLCM. For GP-ABC-SMC and GP-ABC-OLCM this is chosen as $S_{MC} = 9$. As in the previous example these $S_{MC}$ values are found through multiple trials of each of the algorithms on these datasets and inspecting the quality of the posterior estimates.

The results are listed in Tables 2 (bottom) and 3. The marginal posterior distributions of each of the four parameters learnt by the GP-ABC-OLCM and the ABC-SMC-OLCM are shown in Fig. 2. As in the previous example the histograms in Fig. 2 are based on the final particle populations generated by the respective algorithms. In this example we see a huge speedup while using our proposed GP-ABC-SMC and GP-ABC-OLCM algorithms, demonstrating the benefits of this approach. As in the previous example we noticed higher acceptance rates (fewer generated particles) for the GP variants of ABC-SMC. The noise is estimated as $\sigma_{\mu} = 6.8080$, $\sigma_p = 128.6910$, $\sigma_{\mu} = 6.9280$, $\sigma_{\mu} = 123.9290$ and $\sigma_{\mu} = 6.1220$, $\sigma_p = 128.1290$ for the dataset 1, 2 and 3 respectively. Furthermore, from Table 3 it is apparent that although the means of the parameters have similar estimates, their corresponding confidence intervals are different between the proposed GP variants and the original ABC-SMC algorithms. However it should be considered that for GP-ABC-SMC (with both the perturbation kernels) no knowledge of the initial history function is required. Thus in a practical setting we believe a GP based ABC-SMC algorithm is the optimal choice among these four methods for parameter estimation in DDEs.

### Table 3 Estimated parameters of the Hes1 loop model

| Parameters | True value | ABC-SMC-Comp | ABC-SMC-OLCM | GP-ABC-SMC | GP-ABC-OLCM |
|------------|------------|--------------|--------------|------------|-------------|
| $\mu_m$    | 0.03       | 0.0307 ± 2.1608 × 10^{-4} | 0.0305 ± 2.9381 × 10^{-4} | 0.0295 ± 1.3929 × 10^{-4} | 0.0293 ± 1.1127 × 10^{-4} |
|            | 0.0341 ± 6.5563 × 10^{-5} | 0.0342 ± 4.3428 × 10^{-5} | 0.0304 ± 2.6599 × 10^{-4} | 0.0302 ± 2.3190 × 10^{-4} |
| $\mu_p$    | 0.03       | 0.0336 ± 4.1649 × 10^{-5} | 0.0336 ± 8.5782 × 10^{-5} | 0.0291 ± 1.8721 × 10^{-4} | 0.0292 ± 1.8657 × 10^{-4} |
|            | 0.0294 ± 2.0457 × 10^{-4} | 0.0297 ± 2.8397 × 10^{-4} | 0.0300 ± 3.8592 × 10^{-6} | 0.0300 ± 2.8604 × 10^{-6} |
| $p_0$      | 100        | 99.4130 ± 0.0537 | 99.4518 ± 0.0697 | 99.5991 ± 0.2946 | 99.6997 ± 0.2058 |
|            | 102.1872 ± 0.0362 | 102.2306 ± 0.0281 | 100.8624 ± 0.2628 | 100.8624 ± 0.2302 |
|            | 101.2097 ± 0.0279 | 101.2549 ± 0.0495 | 100.0403 ± 0.2796 | 100.0593 ± 0.2459 |
| $t_d$      | 100        | 25.1318 ± 0.0109 | 25.1580 ± 0.0151 | 25.0496 ± 0.1139 | 25.0502 ± 0.0871 |
|            | 25.2317 ± 0.081 | 25.2428 ± 0.0056 | 25.9357 ± 0.2031 | 25.6215 ± 0.1589 |
|            | 25.0730 ± 0.0055 | 25.0714 ± 0.0107 | 25.3187 ± 0.1414 | 25.4469 ± 0.1519 |

6.3 ABC variability

Our proposed method comprises of two levels of approximation, one induced through the GP regression and the other one resulting from the approximate inference scheme. Thus in order to check the robustness of our proposed algorithm we repeated the GP-ABC-SMC and GP-ABC-OLCM parameter inference steps for 50 runs on each of the three artificial datasets for both the Lotka Volterra and Hes1 models. We used the same algorithmic settings and prior distributions as in the previous examples. Figures 3 and 4 summarize the distributions of the sample mean and variance (corresponding to the final particle population for each run of GP-ABC-SMC and GP-ABC-OLCM on the three artificial datasets) across all the 50 runs on the data from Lotka Volterra and Hes1 respectively. It is evident from Fig. 3 that the GP-ABC-OLCM algorithm produces fewer outliers compared to the GP-ABC-SMC for both the mean and variance estimates. This can be attributed to the local moves in the parameter space caused by the multivariate (OLCM) perturbation kernel. Furthermore, note that the distributions of the variances...
resulting from each of the algorithms are skewed in a similar manner with the outliers located at the same direction (for both GP variants) from the median. Thus these outliers represent greater variance of posterior distribution of the parameters. However, in more than 90% out of the 50 runs the moments for both the parameters lie within the interquantile range.

In case of the Hes1 model it is apparent from Fig. 4 that the distributions are less variable across multiple runs and variants of the algorithms. Moreover, in this case we notice that the distribution of the variances have very few outliers indicating greater accordance among the posteriors learnt after each run of the algorithms. However, it should be noted that an adaptive tolerance schedule results in different (marginally) tolerance values for each new run of the ABC-SMC.

Thus some amount of variability in the moments corresponding to different runs is attributed to the differing tolerances.

Figures 5 and 6 show the learnt state trajectories of the Lotka Volterra and Hes1 model compared against the true state trajectories for each of the datasets. The true trajectories correspond to the true parameters and the reconstructed trajectories are generated by solving the Lotka Volterra (26) and Hes1 (27) model equations. While solving (numerically integrating) these differential equations the parameters are taken as the median of the parameters learnt by the GP-ABC-SMC algorithm considering all the 50 runs. The median value is considered here to reflect the effect of variability (in parameter learning by the GP-ABC-SMC) in reconstructing the dynamics of the considered models.
Fig. 3 The boxplots represent the distributions of the mean and variances (across 50 runs) of the final population representing the marginal approximate posterior parameter distributions learnt by the GP-ABC-SMC and the GP-ABC-OLCM from the three artificial datasets of Lotka-Volterra model. a Distribution of the mean for Lotka-Volterra across 50 runs of GP-ABC-SMC and GP-ABC-OLCM. b Distribution of the variance for Lotka-Volterra across 50 runs of GP-ABC-SMC and GP-ABC-OLCM.

6.4 Signal transduction cascade

We have, so far, used the benchmarking examples to compare our proposed GP-based ABC-SMC approach to others of that ilk that exist in the literature. In this example we will compare the parameter estimation results for the proposed GP based ABC-SMC with other (methods not falling under ABC) recent GP based approximate inference methods for...
Fig. 4 Distributions of the mean and variances learnt by the GP-ABC-SMC and the GP-ABC-OLCM from the three artificial datasets of Hes1 model. 

(a) Distribution of the mean for Hes1 across 50 runs of GP-ABC-SMC and GP-ABC-OLCM.

(b) Distribution of the variance for Hes1 across 50 runs of GP-ABC-SMC and GP-ABC-OLCM.
parameter estimation in ODEs. For this purpose we have chosen the signal transduction cascade model (Vyshemirsky and Girolami 2008). Using this model, a comparison between the competing GP based approaches were reported in Wang and Barber (2014). Thus evaluating the proposed GP based ABC-SMC algorithm on this model (with identical settings to those in Wang and Barber 2014) will enable us to draw comparisons with these other methods. This model is described by a 5-dimensional coupled ODEs given by

\[
\begin{align*}
\frac{d[S]}{dt} &= -k_1[S] - k_2[S][R] + k_3[RS] \\
\frac{d[S_d]}{dt} &= k_1[S]
\end{align*}
\]
Fig. 6 Reconstructed and true state trajectories of the Hes1 model. a Trajectories of $\mu(t)$. b Trajectories of $\rho(t)$. (Color figure online)

\[
\frac{d[R]}{dt} = -k_2[S][R] + k_3[RS] + \frac{V[R_{pp}]}{K_m + [R_{pp}]}
\]

\[
\frac{d[RS]}{dt} = k_2[S][R] - k_3[RS] - k_4[RS]
\]

\[
\frac{d[R_{pp}]}{dt} = k_4[RS] - \frac{V[R_{pp}]}{K_m + [R_{pp}]},
\]

where $\theta = (k_1, k_2, k_3, k_4, V, k_m)$ are the parameters of this model and $X(t) = ([S], [S_d], [R], [RS], [R_{pp}])$ are the concentrations of the state variables. Following Wang and Barber (2014) we generated data from the model between the time interval $0 \leq t \leq 100$ with parameters $\theta = (0.07, 0.6, 0.05, 0.3, 0.017, 0.3)$ and initial values
of the state variable $[S] = 1$, $[S_d] = 0$, $[R] = 1$, $[RS] = 0$, $[R_{pp}] = 0$. We then sampled the data at time $t_L = \{0, 1, 2, 4, 5, 7, 10, 15, 20, 30, 40, 50, 60, 80, 100\}$ and added random noise with standard deviation $\sigma_{[S]}$, $\sigma_{[S_d]}$, $\sigma_{[R]}$, $\sigma_{[RS]}$, $\sigma_{[R_{pp}]}$ set to 0.1 for generating the synthetic data. For inferring parameters in this example we apply the GP-ABC-OLCM algorithm from our study with multiple runs, where we found this algorithm to provide a stable and fast inference mechanism. The non-stationarity in the time evolution of the state variables is captured by the MLP covariance function given by

$$k(t, t') = \sigma^2_k \frac{\sigma^2_{w} t^T t' + \sigma^2_{b}}{\sqrt{\sigma^2_{w} t^T t + \sigma^2_{b} + 1}} \cdot$$

where the kernel variance $\sigma^2_k$, the neural network weight variance $\sigma^2_{w}$, and the bias variance $\sigma^2_{b}$ are the hyperparameters of the covariance function. The derivative of this kernel with respect to the input time $t$ is given by

$$\frac{\partial k(t, t')}{\partial t} = \sigma^2_k \frac{2 \sigma^2_{w} t' + \sigma^2_{b}}{\sqrt{1 - Z^2}} \frac{\partial Z}{\partial t},$$

where

$$Z = \frac{\sigma^2_{w} t^T t' + \sigma^2_{b}}{Z_{norm}}$$

with $Z_{norm} = \sqrt{\sigma^2_{w} t^T t + \sigma^2_{b} + 1}$. All the other algorithmic settings were kept the same. The prior distributions are chosen as $k_1 \sim U(0.05, 0.09)$, $k_2 \sim U(0.4, 0.8)$, $k_3 \sim U(0.03, 0.07)$, $k_4 \sim U(0.1, 0.5)$, $V \sim U(0.015, 0.0195)$ and $k_m \sim U(0.1, 0.5)$. In this example $S_{MC}$ is set to 3.

The resulting parameter estimates are furnished in Table 4 along with the parameter estimates obtained from other GP based algorithms run on the same model. These algorithms are the GP-ODE method proposed in Wang and Barber (2014), the adaptive gradient matching (AGM) proposed in Dondelinger et al. (2013) and the gradient matching (GM) proposed in Calderhead et al. (2008). In Table 4 we have summarised the GP-ABC-OLCM output using mean and standard deviation where the choice of the latter has been made to facilitate comparison. We have compared the true state trajectories with the reconstructed trajectories in Fig. 7. We generated the reconstructed trajectories by solving (28) using the mean of the final population of GP-ABC-OLCM, representing the marginal posterior densities of the parameters. The estimated values of the standard deviations are $\sigma_{[S]} = 0.96$, $\sigma_{[S_d]} = 0.081$, $\sigma_{[R]} = 0.0707$, $\sigma_{[RS]} = 0.0591$ and $\sigma_{[R_{pp}]} = 0.0754$. We avoided the comparison of run-time or acceptance rates as the GP-ABC-OLCM and other GP based algorithms depend on completely different approximate inference scheme. However, GP-ABC-OLCM is significantly faster than the other approaches. The GP-ABC-OLCM finishes the estimation in around 20 s while the other methods were run for 30 min to obtain a properly mixed Markov chain.

The ratio of the last two parameters $V/k_m$ (Dondelinger et al. 2013) is a crucial quantity that determines the reconstruction accuracy. GP-ABC-OLCM is able to infer this quantity with the best (based on the estimated posterior means of $V$ and $k_m$) accuracy among all the GP based algorithms.

It is interesting to note that the variance in estimates for the parameters are less than other GP based approaches which use Markov chain Monte Carlo (MCMC). Since we use ABC for inference our estimated variance should be bigger compared to MCMC based inference results. One reason for this could be particle degeneracy which drives all but a few weights to near zero values. Particle degeneracy could be monitored through quantifying the effective sample size (ESS) (Del Moral et al. 2012) given at SMC step $\tau$ by

$$ESS(w_\tau^{(i)}) = \left( \sum_{i=1}^{N} (w_\tau^{(i)})^2 \right)^{-1},$$

Table 4 Estimated parameters of the signal transduction cascade by all the GP based approaches including the GP-ABC-OLCM

| Parameters | True value | GP-ABC-OLCM | GP-ODE | AGM | GM |
|------------|------------|-------------|--------|-----|----|
| $k_1$      | 0.070      | 0.0708 ± 0.0086 | 0.0747 ± 0.0130 | 0.0771 ± 0.0130 | 0.0762 ± 0.0130 |
| $k_2$      | 0.6        | 0.5806 ± 0.0706 | 0.6230 ± 0.1246 | 0.5460 ± 0.1259 | 0.5632 ± 0.1256 |
| $k_3$      | 0.05       | 0.0480 ± 0.0074 | 0.0530 ± 0.0135 | 0.0593 ± 0.0111 | 0.0594 ± 0.0115 |
| $k_4$      | 0.3        | 0.3439 ± 0.0659 | 0.2960 ± 0.0281 | 0.3750 ± 0.0999 | 0.3754 ± 0.1051 |
| $V$        | 0.017      | 0.0170 ± 0.0009 | 0.0177 ± 0.0014 | 0.0172 ± 0.0015 | 0.0173 ± 0.0014 |
| $k_m$      | 0.3        | 0.3110 ± 0.0774 | 0.4220 ± 0.0690 | 0.4090 ± 0.0911 | 0.4186 ± 0.0953 |

The estimates for GP-ODE, AGM and GM are taken from Wang and Barber (2014).
where \( w_i^{(1)} \) is the weight of particle \( i \) and \( N \) is the population size. The value of ESS lies between 1 and \( N \) and its interpretation is that inference based on \( N \) weighted samples is approximately equivalent inference based on \( ESS(w_i^{(1)}) \) perfect samples from the intermediate distribution at SMC step \( \tau \). Although ESS is not a perfect measure, it does provide a probe into the behaviour of the algorithm. Generally when ESS falls below a threshold \( N' \), generally greater than \( N/2 \) the particles are resampled. In the ABC-SMC the importance sampling is performed with the weights and thus resampling is not required unlike SMC samplers (Del Moral et al. 2006). However, for corner cases such as this example where the priors have very narrow support, monitoring degeneracy through ESS is useful. Thus we calculated the ESS values for each of the SMC steps (plotted in Fig. 8, blue curve) and from these values, indicating the number of healthy particles, it is clear that the posterior estimates have less than half of the total particles representing a believable sample approximating the true posterior. This specific example highlights a crucial fallacy that the ABC-SMC, for unnatural priors, can potentially fall prey to particle degeneration. To be absolute sure about the choice of priors leading to this problem we ran the ABC-SMC keeping the same GP estimates of the states and the velocities with increasing the support of the priors. The new priors are thus chosen as \( k_1 \sim U(0, 2) \), \( k_2 \sim U(0, 2) \), \( k_3 \sim U(0, 2) \), \( k_4 \sim U(0, 2) \), \( V \sim U(0, 0.1) \) and \( k_m \sim U(0.1, 2) \). We have kept the prior for \( V \) comparatively narrow to ensure reconstruction accuracy. The \( S_{MC} \) is set to 4. The resulting posterior distributions are shown as histogram plots of the final population of particles in Fig. 9. We have also monitored ESS (shown in Fig. 8 brown curve) which with these choices of prior distributions do not show particle degeneracy. The reconstruction accuracies of the state time courses is shown in Fig. 10. Clearly the recon-

![Fig. 7 Results of GP-ABC-OLCM for the signal transduction cascade (\([S], [S_x], [R], [RS], [R_pp] \) in (a), (b), (c), (d) and (e) respectively. In all the plots observations are the black stars, the true state trajectory is the red (dashed) curve and the blue curve shows the reconstructed trajectory. We have also plotted the GP mean function as the magenta curve and the 95% confidence region is shown as the shaded area. The reconstructed trajectory is generated by numerically integrating (28) with the parameters set to the mean of the posterior distribution estimated by the GP-ABC-OLCM algorithm. (Color figure online)](image)

![Fig. 8 The ESS values indicating number of healthy or alive particles at each of the SMC steps. Except the first SMC step the number of alive particles fall well below 50 particles indicating particle degeneracies when using narrow priors (blue dashed curve) centred on the true values of the parameters. While using a wider prior we see a marked improvement in the ESS above 90% (brown dashed curve). (Color figure online)](image)
Posterior distributions shown as histogram plots. The standard deviations \( std \) for each parameters’ final particle populations are shown on top of each of the plot. Clearly the posterior variances are bigger than those using MCMC in Table 4.

construction suffers in comparison to Wang and Barber (2014), although in our case we use a much less informative priors. Also note that our proposed algorithm lacks a feedback mechanism from the ODE system while carrying out GP regression and thus since some of the states are not well estimated by the GP model, the resulting posterior estimates suffer when uninformative priors are chosen. Thus a critical comparison of GP-ABC-SMC and GP-ODE (which use the aforementioned feedback mechanism) is required for these choices of priors to benchmark our two-step against adaptive methods such as GP-ODE. However, such a critical comparison is not feasible with the time and resources at our disposal.

7 Computational limitations of GP-ABC-SMC

The number of observations creates a significant problem for any GP method in terms of memory and computational time. Each step of likelihood optimisation and the inference requires inverting a matrix of size \( n \times n \) with complexity \( \mathcal{O}(n^3) \) and storage \( \mathcal{O}(n^2) \). For GP-ABC-SMC we need to carry out GP regression once and thus as long as we are able to complete GP regression within a suitable time (\( \approx 200 \) s for the Lotka Volterra model as an example) and without running out of memory, the rest of the SMC steps involving simulating \( f(\hat{X}(t), \theta) \) is always faster than solving the corresponding non-linear ODE. On our machine (Intel(R) Core(TM) i7-2600 CPU 3.40 GHz, 16 GB RAM) using the GPML package (Rasmussen and Nickisch 2010), running GP regression on one of the species of Lotka Volterra for \( n = 3000 \) input time points require 305 s while using exact inference for the GP. Generation of \( \approx 13000 \) simulated observations as required while running ABC-SMC-OLCM (see Table 2) takes around 2h, where we have calculated the time for generating a single simulated data, by solving the Lotka Volterra ODE, to be \( \approx 0.08 \) s using MATLAB’s ODE45 routine. Hence for a similar model with \( \leq 3000 \) input time points we recommend the use of GP-ABC-
SMC over ABC-SMC using exact inference for the GP regression.

8 Conclusion

In this paper we have proposed a method that significantly speeds up the task of parameter inference in comparison with state of the art methods that use ABC and SMC based approaches when applied to several benchmark dynamical system models that are described by ODE and DDE. We achieve this speed-up by circumventing the need to numerically integrate the differential equations, a task that is repeatedly required in other ABC methods to generate samples from candidate models for comparison with the observed data. The key idea behind our method lies in building on Calderhead et al. (2008), Ramsay et al. (2007) and Wang and Barber (2014) to work directly with the vector field of the dynamical system, which we model using GP regression, and thus create a distance function in derivative space for use in the ABC-SMC algorithm as proposed in Toni et al. (2009). Thus we proposed a modified ABC-SMC algorithm for parameter estimation in ODEs or DDEs.

We benchmarked the benefits of this approach by evaluating our proposed approach on toy problems where we observed a significant speed-up of the parameter estimation process. We also compared our proposed approach with other GP based methods proposed in recent literature and found that our proposed GP-ABC-SMC (with the local multivariate perturbation kernel) performs significantly faster to obtain similar estimates. However, careful choices of prior distributions should be made in order to achieve true comparison between the ABC (GP-ABC-SMC) and exact MCMC (GP-ODE, GP-GM, GP-AGM) based gradient matching algorithms. Furthermore, improvements of ABC-SMC through perturbation kernels as proposed in Filippi et al. (2013) and has been integrated with our approach to obtain enhanced performance. Thus, our fast approximate inference process can accommodate the useful features of ABC-SMC (as shown in Toni et al. 2009) such as model selection and sensitivity analysis. Our proposed approach is only limited by the ability of the Gaussian process regression in smoothing the observed time series data while retaining the essential characteristics that are meant to be captured by the dynamical system model. Thus in those cases where smoothing the experimental data by GP regression introduces artefacts, the GP-ABC-SMC algorithm would produce poor parameter estimates. As future work we wish to extend the GP-ABC-SMC algorithm for stochastic differential equation by relating the GP regression technique with drift estimation technique in Ruttor et al. (2013). We will also include models with hidden variables, as the smoothing procedure on observed data can no longer provide complete information.

Acknowledgements We want to thank the anonymous reviewers for their valuable comments. This work was supported by project PLants Employed As SEnsor Devices (PLEASED), EC grant agreement number 296582.
References

Beaumont, M.A., Cornuet, J.M., Marin, J.M., Robert, C.P.: Adaptive approximate Bayesian computation. Biometrika 96(4), 983–990 (2009)

Calderhead, B., Girolami, M., Lawrence, N.D.: Accelerating Bayesian inference over nonlinear differential equations with Gaussian processes. In: Proceedings of Advances in Neural Information Processing Systems 21, pp 217–224 (2008)

Del Moral, P., Doucet, A., Jasra, A.: Sequential Monte Carlo samplers. J. R. Stat. Soc. Ser. B 68(3), 411–436 (2006)

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

Didelot, X., Everitt, R.G., Johansen, A.M., Lawson, D.J.: Likelihood-free estimation of model evidence. Bayesian Anal. 6(1), 49–76 (2011)

Dondelinger, F., Filippone, M., Rogers, S., Husmeier, D.: ODE parameter inference using adaptive gradient matching with Gaussian processes. In: Proceedings of 16th International Conference on Artificial Intelligence and Statistics, pp 216–228 (2013)

Drovandi, C.C., Pettitt, A.N.: Estimation of parameters for macroparasite population evolution using approximate Bayesian computation. Biometrics 67(1), 225–233 (2011)

Filippi, S., Barnes, C.P., Cornebise, J., Stumpf, M.P.: On optimality of kernels for approximate Bayesian computation using sequential Monte Carlo. Stat. Appl. Genet. Mol. Biol. 12(1), 87–107 (2013)

MacKay, D.J.C.: Introduction to Gaussian processes. NATO ASI Ser. F 168, 133–166 (1998)

Monk, N.A.M.: Oscillatory expression of Hes1, p53, and NF-κB driven by transcriptional time delays. Curr. Biol. 13, 1409–1413 (2003)

Murray, J.D.: Mathematical Biology: I. An Introduction. Springer, New York (2002)

Neal, R.M.: Regression and classification using Gaussian process priors. Bayesian Stat. 6, 475–501 (1998)

O’Hagan, A., Kingman, J.F.C.: Curve fitting and optimal design for prediction. J. R. Stat. Soc. Ser. B 40, 1–42 (1978)

Ramsay, J.O., Hooker, G., Campbell, D., Cao, J.: Parameter estimation for differential equations: a generalized smoothing approach. J. R. Stat. Soc. Ser. B 69(5), 741–796 (2007)

Rasmussen, C.E., Nickisch, H.: Gaussian processes for machine learning (GPML) toolbox. J. Mach. Learn. Res. 9999, 3011–3015 (2010)

Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)

Ruttor, A., Batz, P., Opper, M.: Approximate Gaussian process inference for the drift of stochastic differential equations. In: Proceedings of Advances in Neural Information Processing Systems 26, pp 2040–2048 (2013)

Sisson, S.A., Fan, Y., Tanaka, M.M.: Sequential Monte Carlo without likelihoods. Proc. Natl Acad. Sci. U.S.A. 104(6), 1760–1765 (2007)

Sisson, S.A., Fan, Y., Tanaka, M.M.: Correction for Sisson sequential Monte Carlo without likelihoods. Proc. Natl Acad. Sci. U.S.A. 106(39), 16,889 (2009)

Solak, E., Murray-Smith, R., Leithead, W., Rasmussen, C., Leith, D.: Derivative observations in Gaussian process models of dynamic systems. In: Proceedings of Advances in Neural Information Processing Systems 15, pp 1033–1040 (2002)

Toni, T., Welch, D., Strelkowa, N., Ipsen, A., Stumpf, M.P.: Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. J. R. Soc. Interface 6(31), 187–202 (2009)

Varah, J.M.: A spline least squares method for numerical parameter estimation in differential equations. SIAM J. Sci. Stat. Comput. 3(1), 28–46 (1982)

Vyshemirsky, V., Girolami, M.A.: Bayesian ranking of biochemical system models. Bioinformatics 24(6), 833–839 (2008)

Wang, Y., Barber, D.: Gaussian processes for Bayesian estimation in ordinary differential equations. In: Proceedings of 31st International Conference on Machine Learning, pp 1485–1493 (2014)