Comparing data assimilation filters for parameter estimation in a neuron model

Nicola Politi
Department of Mathematics
Università degli Studi di Torino
and Politecnico di Torino
Torino (TO), Italy
Email: nicola.politi@polito.it

Jianfeng Feng
Centre for Computational Systems Biology
Fudan University, Shanghai 200433, China
and Department of Computer Sciences
The University of Warwick
Coventry, UK
Email: jff@fudan.edu.cn

Wenlian Lu
School of Mathematical Sciences
Centre for Computational Systems Biology
Fudan University, Shanghai, China
Email: w.l.lu@ieee.org

***Abstract*—Data assimilation (DA) has proved to be an efficient framework for estimation problems in real-world complex dynamical systems arising in geoscience, and it also initially showed its power to computational neuroscience. The ensemble Kalman filter (EnKF) is believed to be a powerful tool of DA in practice. In comparison to the other filtering methods of DA, such as the bootstrap filter (BF) and optimal sequential importance re-sampling (OPT-SIRS), it is more convenient and efficient to be implemented in a parallel way but with a theoretical flaw of Gaussian assumption. In this paper, we apply the EnKF to the estimation and prediction of a single computational neuron model with 10 parameters and conduct a comparison study of these three DA filtering methods on this model. It is numerically shown that the EnKF presents the best performance in both accuracy and computation load. We argue that the EnKF will be a promising tool in the large-scale DA problem occurring in computational neuroscience with experimental data.**

I. Introduction

The last few decades have witnessed the fast increasing interest in computational neuroscience, which heavily depends on the biophysical modeling of neurons initiated by the pioneering work by Hodgkin and Huxley [1]. So far, single neuron models are coupled into neuronal networks, which have been widely applied to describing, for instance, realistic hippocampal [2] and neocortical microcircuitry [3]. Nevertheless, most of these works are qualitative rather than quantitative, because techniques for experimental parameter assessing are still extremely invasive. Also, models still cannot be fine-tuned to given neuronal-population-specific data owing to the larger number of parameters occurring in these models. This is hardly treated by ordinary parameter estimation techniques.

Recently, an intuitively-simple and novel technique, named data assimilation (DA), has been widely employed to embed experimental data into mathematical models. In particular DA proved to be successful in a lot of complex realistic systems, such as weather forecasting [4], oceanography [5] and hydrology [6]. This technique is a direct realization of the standard Bayesian inference procedure, and it works comparing data simulated by models to real data. A lot of works can be found in developing and improving parameter estimation techniques of DA [7]–[9]. For more details, please refer to the textbooks [10], [11].

Only recently this method has been applied in the field of computational neuroscience to estimate model parameters. The bootstrap filter, also named self-organized state-space model [12], was used in parameter estimation of neuronal model [13]. Variational calculus, also termed 4DVAR in DA [14], was employed too [15]–[17]. Still, at our best knowledge the classic ensemble Kalman filter has not been applied to this field yet. In this paper, we present a numerical comparison and evaluation of the performance of three types of DA filters, the ensemble Kalman filter (EnKF), the bootstrap filter (BF) and its modified version, the optimal sequential importance re-sampling (OPT-SIRS), as well as the variational calculus method. This comparison is carried out in order to find the best method of DA in the task of estimating parameters in a neuronal model.

II. Bayesian Data Assimilation

Generally speaking, data assimilation aims at describing a physical system by a data-tuned mathematical model. Consider the following discrete-time stochastic dynamical model

$$\begin{cases} z_{j+1} &= g_j(z_j) + \epsilon_j, \quad \forall j \in \{0, \dots, J-1\}, \\ z_0 &\sim p(z_0). \end{cases} \tag{1}$$

Here, for $j \in \{0, \dots, J\}$, $z_j \in \mathbb{R}^d$ stands for the state variable of the *model dynamics* and $g_j : \mathbb{R}^d \longrightarrow \mathbb{R}^d$ is the forward map at time step j (note that the dynamics is time-dependent); $\{\epsilon_j\}_{j=0}^{J-1}$ is a sequence of random variables (r.v.) mutually independent and identically distributed (i.i.d) with $\epsilon_0 \sim \mathcal{N}(0, \Sigma)$; z_0 is the random initial condition with a given probability distribution $p(z_0)$, independent of the noise. Thus, one can conclude that $\{z_j\}_{j=0}^{J}$ forms a discrete-time Markov chain with the transition density function (p.d.f.)

$$p(z_{j+1} \mid z_j) \propto \exp\big(-\frac{1}{2}|z_{j+1} - g_j(z_j)|_\Sigma^2\big), \tag{2}$$

where $|\cdot|_A := |A^{-1}\cdot|$ is the A-induced norm on $\mathbb{R}^d$ for a positive-definite symmetric matrix $A \in \mathbb{R}^{d\times d}$. The density of z_{j+1}, $p(z_{0:j+1}) = p(z_{j+1} \mid z_j)p(z_j \mid z_{j-1}) \cdots p(z_0)$, is named *prior distribution*.

Let the data set $\{y_{j+1}\}_{j=0}^{J-1} = \{y_1, \dots, y_J\} \subseteq \mathbb{R}^q$ be a noisy observation of the state variable $\{z_j\}_{j=0}^{J}$:

$$y_{j+1} = H(z_{j+1}) + \eta_{j+1}, \quad j \in \{0, \dots, J-1\}. \tag{3}$$

Here $H(\cdot)$ is a linear function from the state space $\mathbb{R}^d$ to the data space $\mathbb{R}^q$; the measurement noise process $\{\eta_{j+1}\}_{j=0}^{J-1}$ is a i.i.d. sequence with $\eta_1 \sim \mathcal{N}(0, \Gamma)$ independent of both the initial condition z_0 and the dynamical noise $\{\epsilon_j\}_{j=0}^{J}$. It is called *likelihood* the conditional density function

$$p(y_{j+1} \mid z_{j+1}) \propto \exp\big(-\frac{1}{2}|y_{j+1} - H(z_{j+1})|_{\Gamma}^2\big). \tag{4}$$

In Bayesian DA, the likelihood is used in order to drive an uncertain stochastic dynamics towards an indirect empirical measure of such process.

The estimation, also named *posterior distribution*, is computed by by the Bayes' theorem, $p(z \mid y) \propto p(y \mid z)p(z)$. There are generally two classes of Bayesian DA algorithms depending on what variable Bayes' theorem is applied to. The first one is filtering, which updates the posterior *filtering distribution* at each time $j+1$, $p(z_{j+1}|y_1, \dots, y_{j+1})$, only using the preceding time step j, $p(z_j|y_1, \dots, y_j)$. We henceforth use the shorthand notation $y_{1:j} = \{y_1, \dots, y_j\}$ and $z_{0:j} = \{z_0, \dots, z_j\}$. The filtering update is typically performed in the following two-steps procedure. At the first step, named *prediction step*, the filtering distribution is pushed-forward in time only relying on model dynamics, i.e.

$$p(z_{j+1} \mid y_{1:j}) = \int p(z_{j+1} \mid z_j) p(z_j \mid y_{1:j}). \tag{5}$$

This results in the predicted distribution $p(z_{j+1} \mid y_{1:j})$. At the second step, named *analysis step*, the filtering distribution $p(z_{j+1} \mid y_{1:j+1})$ is computed by multiplying the predicted distribution times the likelihood, i.e.

$$p(z_{j+1} \mid y_{1:j+1}) \propto p(y_{j+1} \mid z_{j+1}) p(z_{j+1} \mid y_{1:j}).$$

according to the Bayes' theorem. These two steps run iteratively.

On the other hand, smoothing methods aim at approximating the posterior *smoothing distribution* $p(z_{0:J} \mid y_{0:J})$. The marginal of the smoothing distribution is

$$p(z_j \mid y_{0:J}) = \int p(z_{0:J} \mid y_{0:J}) dz_0 \cdots dz_{j-1} dz_{j+1} \cdots dz_J$$

and it fully characterizes the posterior distribution at time j.

In what follows we describe in detail three filtering algorithms, namely the EnKF, the BF, and the OPT-SIRS, and briefly present `minAone`, a smoothing method.

Ensemble Kalman filter

The EnKF is an approximate Guassian filter, which generalizes the linear Kalman filter [18] to non-nonlinear and non-Gaussian problems. In the prediction step, the EnKF draws $N \in \mathbb{N}$ independent samples of the dynamical noise $\epsilon^{(n)}$ and use them to build the predicted *ensemble* $\{\hat{z}_{j+1}^{(n)}\}_{n=1}^{N}$ through model dynamics (1). Then, the ensemble covariance $\hat{C}_{j+1}$ is used to update the filtering ensemble $\{z^{(n)}\}_{n=1}^{N}$ by implementing a Kalman-filter-type analysis step.

The complete EnKF algorithm is as follows.

Step 0) *Initialization:* draw the initial ensemble by independently sampling N particles from the initial distribution $p(z_0)$: for $n = 1, \dots, N$, draw N initial values $z_0^{(n)} \sim p(z_0)$.

Then, for time points $j = 0, \dots, J-1$ apply:

Step 1) *Prediction step:* build the predicted ensemble and compute its empirical covariance matrix,

$$\begin{cases} \text{For} & n = 1, \dots, N: \\ \text{draw} & \epsilon^{(n)} \sim \mathcal{N}(0, \Sigma), \\ \text{set} & \hat{z}_{j+1}^{(n)} = g_j(z_j^{(n)}) + \epsilon_j^{(n)}. \\ \text{Set} & \hat{m}_{j+1} = \frac{1}{N}\sum_{n=1}^{N} \hat{z}_{j+1}^{(n)}, \\ & \hat{C}_{j+1} = \frac{1}{N-1}\sum_{n=1}^{N} (\hat{z}_{j+1}^{(n)} - \hat{m}_{j+1})(\hat{z}_{j+1}^{(n)} - \hat{m}_{j+1})^T \end{cases}$$

Step 2) *Analysis step:* compute the EnKF filtering ensemble,

$$\begin{cases} \text{Set} & S_{j+1} = H\hat{C}_{j+1}H^T + \Gamma, \\ & K_{j+1} = \hat{C}_{j+1}H^T S_{j+1}^{-1}. \\ \text{For} & n = 1, \dots, N: \\ \text{draw} & \eta^{(n)} \sim \mathcal{N}(0, \Gamma), \\ \text{set} & y_{j+1}^{(n)} = y_{j+1} + \eta_{j+1}^{(n)}, \\ & z_{j+1}^{(n)} = \hat{z}_{j+1}^{(n)} + K_{j+1}(y_{j+1}^{(n)} - H\hat{z}_{j+1}^{(n)}). \end{cases}$$

Bootstrap filter

The BF is a particular version of the particle filter [19], which introduces a particle ensemble to approximate the filtering distribution $p(z_{j+1}|y_{1:j+1})$ by the Monte Carlo estimator

$$p(z_{j+1}|y_{1:j+1}) \approx \mu_{j+1}^{N}(z_{j+1}) := \sum_{n=1}^{N} w_{j+1}^{(n)} \delta_{z_{j+1}^{(n)}}.$$

Here, $\delta_{z_{j+1}^{(n)}}$ stands for the Dirac delta function centered on $z_{j+1}^{(n)}$ and $w_{j+1}^{(n)}$ is the corresponding *importance weight* quantifying the probability of particle $z_{j+1}^{(n)}$. In the prediction step, the ensemble particles $z_j^{(n)}$ are pushed-forward in time by sampling the importance sampling distribution $\pi(z_{j+1} \mid z_j^{(n)}, y_{1:j+1}) = p(z_{j+1} \mid z_j^{(n)})$, which yields to the proposed ensemble $\{\hat{z}_{j+1}^{(n)}\}_{n=1}^{N}$. In the analysis step, the importance weights are updated using the likelihood (4). This generates the following algorithm.

Step 0) *Initialization:* initialize the BF filtering distribution: for $n = 1, \dots, N$, draw $z_0^{(n)} \sim p(z_0)$ and set $\mu_0^N(z_0) = \sum_{n=1}^{N} \frac{1}{N}\delta_{z_0^{(n)}}$. Then, for $j = 0, \dots, J-1$ apply:

Step 1) *Prediction step:* update the ensemble particles according to the prior (2),

$$\begin{cases} \text{For} & n = 1, \dots, N: \\ \text{draw} & \hat{z}_{j+1}^{(n)} \sim p(z_{j+1} \mid z_j^{(n)}). \end{cases}$$

Step 2) *Analysis step:* update the ensemble weights and set the predicted filtering distribution at time $j+1$,

$$\begin{cases} \text{For} & n = 1, \dots, N: \\ \text{set} & \hat{w}_{j+1}^{(n)} = \frac{p(y_{j+1}|\hat{z}_{j+1}^{(n)})}{\sum_{n=1}^{N} p(y_{j+1}|\hat{z}_{j+1}^{(n)})}. \\ \text{Set} & \hat{\mu}_{j+1}^{N}(z_{j+1}) = \sum_{n=1}^{N} \hat{w}_{j+1} \delta_{\hat{z}_{j+1}^{(n)}}. \end{cases} \tag{6}$$

Step 3) *Re-sampling:* re-sample and set the BF filtering distribution at time $j+1$

$$\begin{cases} \text{For} & n = 1, \dots, N: \\ \text{draw} & z_{j+1}^{(n)} \sim \hat{\mu}_{j+1}^N(z_{j+1}). \\ \text{Set} & \mu_{j+1}^N(z_{j+1}) = \sum_{n=1}^N \frac{1}{N} \delta_{z_{j+1}^{(n)}}. \end{cases}$$

It was proved that the Monte Carlo estimator $\mu_{j+1}^N(z_{j+1})$ converges to the true posterior distribution under some hypothesis (see [11, Ch. 4] and references therein) and it is well-posed, i.e. Lipschitz-continuous with respect to the data set $y_{1:J}$ in some probability metrics ([11, Ch. 2]).

Optimal sequential importance re-sampling

Different particle filters are obtained if a different importance sampling distribution is chosen. Here we present the case

$$\pi(z_{j+1} \mid z_j^{(n)}, y_{1:j+1}) = p(z_{j+1} \mid z_j^{(n)}, y_{j+1}), \tag{7}$$

which results in the optimal sequential importance re-sampling (OPT-SIRS). Its name is due to the fact that this choice minimizes the variance of the ensemble weights $w_{j+1}^{(n)}$ conditional upon sample $z_j^{(n)}$ and data $y_{1:j+1}$ (see [20] for further details). With this optimal definition of importance sampling distribution, the importance weight updating in (6) becomes

$$\hat{w}_{j+1}^{(n)} = w_j^{(n)}\, p(y_{j+1} \mid z_j^{(n)}). \tag{8}$$

The complete OPT-SIRS algorithm is

Step 0) *Initialization:* initialize the OPT-SIRS filtering distribution: for $n = 1, \dots, N$, draw $z_0^{(n)} \sim p(z_0)$ and set $\mu_0^N(z_0) = \sum_{n=1}^N \frac{1}{N} \delta_{z_0^{(n)}}$. Then, for $j = 0, \dots, J-1$ apply:

Step 1) *Prediction step:* update the ensemble particles according to the optimal importance distribution (7)

$$\begin{cases} \text{Set} & \hat{\Sigma} = (\Sigma^{-1} + H^T \Gamma^{-1} H)^{-1}. \\ \text{For} & n = 1, \dots, N: \\ \text{set} & \hat{m}^{(n)} = \hat{\Sigma}\big(H^T \Gamma^{-1} y_{j+1} + \Sigma^{-1} g_j(z_j^{(n)})\big), \\ \text{draw} & \hat{z}_{j+1}^{(n)} \sim \mathcal{N}(\hat{m}^{(n)}, \Sigma) \end{cases}$$

Step 2) *Analysis step:* update the ensemble weights and set the proposed filtering distribution at time $j+1$

$$\begin{cases} \text{For} & n = 1, \dots, N: \\ \text{set} & \hat{w}_{j+1}^{(n)} = \exp\Big(-\frac{1}{2} |y_{j+1} - H g_j(z_j^{(n)})|^2_{\Gamma + H\Sigma H^T}\Big). \\ \text{Normalize } \hat{w}_{j+1}^{(n)} & \text{so that } \sum_{n=1}^N \hat{w}_{j+1}^{(n)} = 1. \\ \text{Set} & \hat{\mu}_{j+1}^N(z_{j+1}) = \sum_{n=1}^N \hat{w}_{j+1}^{(n)} \delta_{\hat{z}_{j+1}^{(n)}}. \end{cases}$$

Step 3) *Re-Sampling:* re-sample and set the OPT-SIRS filtering distribution

$$\begin{cases} \text{For} & n = 1, \dots, N: \\ \text{draw} & z_{j+1}^{(n)} \sim \hat{\mu}_{j+1}(z_{j+1}). \\ \text{Set} & \mu_{j+1}^N(z_{j+1}) = \sum_{n=1}^N \frac{1}{N} \delta_{z_{j+1}^{(n)}}. \end{cases}$$

`MinAone`

`MinAone` [21] is an annealing implementation of the variational calculus method 4DVAR. This smoothing algorithm performs data assimilation by minimizing the negative-log-probability cost function $A(z_{1:J}; y_{1:J}) = -\log(p(z_{1:J} \mid y_{1:J}))$ (also called action). Global minima of the action (which is nonlinear in general) are indeed peaks of the posterior distribution, i.e. the modes of the smoothing distribution. In the present paper, we do not delve further into its details but just mention that it numerically minimize the cost function running several instances of the publicly available optimization program IPOPT [22]. We refer to [14] for further details on 4DVAR.

III. Parameter Estimation in Neuron Model with Twin Experiment

In this paper we consider a two-dimensional neuronal model [23]

$$\begin{cases} C\dot{V} &= -g_K a (V - E_K) - g_{Na} b_\infty(V)(V - E_{Na}) \\ & \quad -g_L(V - E_L) + I_{ext}(t) \\ \dot{a} &= \frac{a_\infty(V) - a}{\tau_a}, \end{cases}, \tag{9}$$

defined for $t \in [0, T_f]$, where V is the membrane potential and the ionic-channel activation variable a represents the opening probability of the trans-membrane potassium channel. Parameters of this model include the membrane capacity C and the time scale constant τ_a, the maximal conductance of the potassium, sodium and of the leakage ionic current (g_K, g_{Na} and g_L respectively), and the corresponding equilibrium potential E_K, E_{Na} and E_L. Besides,

$$a_\infty(V) = \frac{1}{1 + \exp\Big((V_{1/2}^{(a)} - V)/K^{(a)}\Big)}, \tag{10}$$

where $K^{(a)}$ is a steepness parameter and $V_{1/2}^{(a)}$ is such that $a_\infty(V_{1/2}^{(a)}) = 0.5$; $b_\infty(V)$ satisfies a functional relation analogous to (10) with corresponding parameters $V_{1/2}^{(b)}$ and $K^{(b)}$. $I_{ext}(t)$ is a preassigned time-dependent externally-applied current, which is assumed to be a piece-wise constant function, i.e.

$$I_{ext}(t) = \begin{cases} I_1 & t \in [0, T^{(1)}) \\ I_i & t \in [T^{(i-1)}, T^{(i)}), \quad i = 2, ..., i_{\max} \\ I_{i_{\max}} & t \in [T^{(i_{\max})}, T_f], \end{cases}$$

where the jump time point set $\{T^{(i)}\}_1^{i_{\max}}$ is a Poisson process with a given rate λ, i.e. $\mathbb{E}[T^{(i+1)} - T^{(i)}] = 1/\lambda$. The corresponding step-current values $\{I_i\}_1^{i_{\max}}$ are modeled as an i.i.d. random sequence with uniform distribution over the interval $[I_{\text{low}}, I_{\text{upp}}]$.

Model dynamics

Our task is to estimate both the parameter vector $\theta = \big(g_{\text{Na}}, E_{\text{Na}}, g_{\text{K}}, E_{\text{K}}, g_{\text{L}}, E_{\text{L}}, K^{(a)}, V_{1/2}^{(a)}, K^{(b)}, V_{1/2}^{(b)}\big)^T$ and the state variable $x = (V, a)^T$. Parameters C and τ_a are assumed

TABLE I
TRUE PARAMETER VALUES AND UNIT MEASURE

$\theta^\dagger$	**True Value**	**Unit**
g_{Na}	20	$\mathrm{mS/cm^2}$
E_{Na}	60	mV
g_{K}	10	$\mathrm{mS/cm^2}$
E_{K}	-90	mV
g_{L}	8	$\mathrm{mS/cm^2}$
E_{L}	-78	mV
$V_{1/2}^{(b)}$	-20	mV
$K^{(b)}$	15	mV
$V_{1/2}^{(a)}$	-45	mV
$K^{(a)}$	5	mV
τ_a	1	ms
C	1	$\mu\mathrm{F/cm^2}$

to be fixed and known, and their value is henceforth set to be as in TABLE I. To apply the DA algorithms, the parameter vector θ is regarded as a state variable with dynamic equation $\dot{\theta} = 0$ [12]. System (9) becomes a discrete-time dynamics by applying a numerical integration scheme over a time mesh $\{t_j\}_{j=0}^{J}$. The parameters and state variables are then turned into random processes by considering the presence of an additive random noise in the model dynamics, which describes the impreciseness of the models. Hence, we have the following model dynamics for $(x_j, \theta_j)^T$

$$\begin{cases} x_{j+1} = f_j(x_j, \theta_j) + \epsilon_{X,j} \\ \theta_{j+1} = \theta_j + \epsilon_{\theta,j}, \end{cases} , \tag{11}$$

for $j \in \{0, \dots, J-1\}$. Note that

- x_j and θ_j are the discrete-time approximate (and noisy) value corresponding to $x(t_j)$ and $\theta(t_j)$ respectively;
- $f_j : \mathbb{R}^{d_X} \times \mathbb{R}^{d_\theta} \longrightarrow \mathbb{R}^m$ represents the discrete version of vector field (9) obtained by applying a given numerical integration method. Since the neuron model's vector field is non-autonomous, f_j is a time-dependent map invoking the correct values of the input current $I_{ext}(t_j)$ (depending on the chosen numerical algorithm).
- the dimension of variables component x_j is $d_X = 2$ and the dimension of parameter component θ_j is $d_\theta = 10$;
- $\{\epsilon_{X,j}\}_{j=0}^{J-1}$ and $\{\epsilon_{\theta,j}\}_{j=0}^{J-1}$ are two mutually independent sequences of i.i.d. random variables with $\epsilon_{X,0} \sim \mathcal{N}(0, \Sigma_X)$ and $\epsilon_{\theta,0} \sim \mathcal{N}(0, \Sigma_\theta)$.

Also, we set the initial condition to be a Gaussian r.v.

$$\begin{cases} x_0 \sim \mathcal{N}(m_{X,0}, C_{X,0}) \\ \theta_0 \sim \mathcal{N}(m_{\theta,0}, C_{\theta,0}), \end{cases}$$

where $m_{X,0}$ and $m_{\theta,0}$ are the variables and parameter component of the initial mean, respectively, whereas $C_{X,0}$ and $C_{\theta,0}$ are the corresponding covariance matrices.

Along with system (11), we are given a set of noisy observation $\{y_{j+1}\}_{j=0}^{J-1}$ linked to the state variable through the data model

$$y_{j+1} = H_X x_{j+1} + H_\theta \theta_{j+1} + \eta_{j+1}.$$

Here, $H_X : \mathbb{R}^{d_X} \longrightarrow \mathbb{R}^q$ and $H_\theta : \mathbb{R}^{d_\theta} \longrightarrow \mathbb{R}^q$ are linear operators (i.e. a $q \times d_X$ and $q \times d_\theta$ matrix, respectively) and $\{\eta_{j+1}\}_{j=0}^{J-1}$ is a i.i.d. sequence with $\eta_1 \sim \mathcal{N}(0, \Gamma)$. Since parameters cannot be directly observed H_θ is set to be a $d_\theta \times d_\theta$ matrix with zero-entries.

Twin experiment design

The objective of this paper was to estimate both membrane potential V and activation variable a as well as parameter values of the neural model (9) in a twin experiment setting as in [15]. In twin experiments, instead of using experimental recording, noisy data are generated from the same mathematical model which is then used to perform data assimilation. This guarantees a controlled environment where only DA method performance is tested rather than specific model-dynamics suitability for a given data set.

In order to produce the data set, we firstly fixed the external current $I_{ext}(t)$ by drawing a single instance of $\{T^{(i)}\}$ and $\{I_i\}$ with $\lambda = 1\,\mathrm{ms}^{-1}$ and $I_{\mathrm{low}} = -5\,\mu\mathrm{A/cm^2}$ and $I_{\mathrm{upp}} = 40\,\mu\mathrm{A/cm^2}$. Then, we also set the *true parameter vector* $\theta^\dagger$ to have the entries listed in TABLE I. We finally used a fourth-order Runge-Kutta method to solve the system of ordinary differential equations (9) over the time mesh $\{t_j\}_{j=0}^{J}$, where $t_j = j \cdot \Delta t$, $\Delta t = 0.01\,\mathrm{ms}$, and $T_f = t_J = 500\,\mathrm{ms}$. The initial condition was set to be $\big(V(0), a(0)\big)^T = \big(V_0^\dagger, a_0^\dagger\big)^T$, where $V_0^\dagger = -64\,\mathrm{mV}$ and $a_0^\dagger = a_\infty(V_0^\dagger)$. In what follows we call *true trajectory* the resulting approximate solution and write $\{x_j^\dagger\} = \{(V_j^\dagger, a_j^\dagger)^T\}_{j=0}^{J}$.

A data set $\{y_{j+1}\}_{j=0}^{J-1}$ was produced by drawing a single instance of the measurement noise $\{\eta_{j+1}\}_{j=0}^{J-1}$ with standard deviation $\Gamma^{1/2} = 1\,\mathrm{mV}$, and setting

$$y_{j+1} = V_{j+1}^\dagger + \eta_{j+1}.$$

This corresponds to set $q = 1$ and $H_X = (1, 0)$.

Filtering methods illustrated in Bayesian Data Assimilation section were applied to system (11) with time window length $J = 50000$, sample size $N = 2000$ and components of the initial condition mean $m_{X,0} = (V_0^\dagger, a_0^\dagger)^T$ and $m_{\theta,0} = \theta^\dagger$. The forward map f_j was built by applying to system (9) the same fourth-order Runge-Kutta method we used to generate the data set. Initial covariance matrices were set to be $C_{X,0} = \mathrm{diag}(25\,\mathrm{mV}^2, 0.1)$ and $C_{\theta,0} = 25\,U_{d_\theta}$ (units as in TABLE I), where in general U_α stands for the $\alpha \times \alpha$ unit matrix with ones on the principal diagonal and zeros elsewhere. Random dynamical noises of the BF and the OPT-SIRS were designed to have the same covariance matrices $\Sigma_{0,X} = 10^{-4}\,U_{d_X}$ and $\Sigma_{0,\theta} = 10^{-5}\,U_{d_\theta}$, whereas for the EnKF $\Sigma_X = 10^{-6}\,U_{d_X}$ and $\Sigma_\theta = 10^{-6}\,U_{d_\theta}$.

The ensemble Kalman filter, the bootstrap filter and the optimal sequential importance re-sampling were implemented in MATLAB® (2015b, The MathWorks Inc., Natick, Massachusetts). Results described in the following section were obtained by running one hundred instances of every filtering algorithm. In each run, all random variables involved in the method were sampled independently. The hardware used was

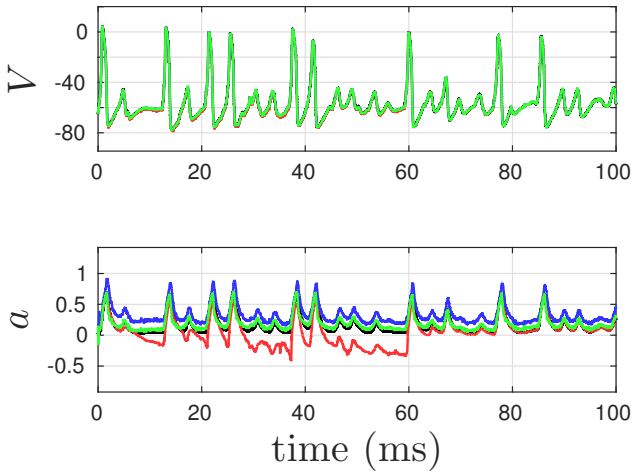

Fig. 1. True trajectory $(V_j^\dagger, a_j^\dagger)^T$ (black line) and variables component of filtering mean $m_{X,j}$ for the ensemble Kalman filter (red line) the bootstrap filter (blue line) and the optimal sequential importance re-sampling (green line). Membrane potential V is measured in millivolt (mV).

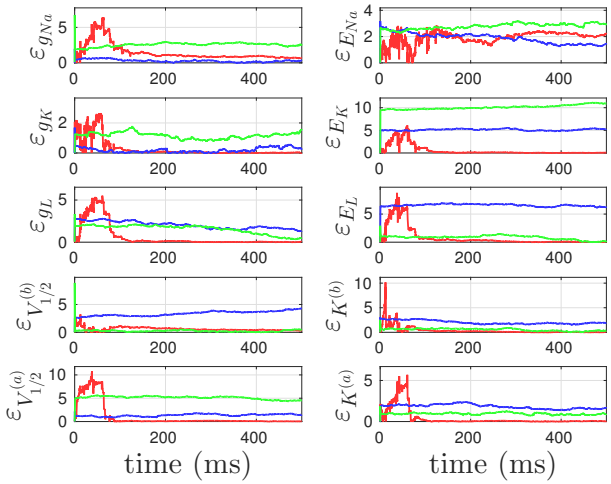

Fig. 2. Parameter component of the filtering error $\varepsilon_{\theta,j} = |m_{\theta,j} - \theta^\dagger|$ for the ensemble Kalman filter (red line) the bootstrap filter (blue line) and the optimal sequential importance re-sampling (green line)

a Dell® PowerEdge T630 Tower Server with two ten-core Intel® Xeon® E52650 v3 CPUs and 128GB RAM running Ubuntu Server 14.04.

Finally, one hundred independent runs of `minAone` were launched with the constraint that the parameter vector lies inside the hypercube centered in $\theta^\dagger$ and side length 10.

IV. Results

Estimation

Fig. 1 illustrates each filter's performance by plotting the variables component of the filtering mean in a representative run. Note that the plot is restricted to the first 100 ms of $[0, T_f]$.

As shown in the top panel, in a typical run the true membrane potential $V^\dagger$ substantially overlaps the mean values of all DA methods, which implies that the three filters can recover the true membrane potential values with good accuracy. However, in the lower panel of Fig. 1, the true the unobserved state variable $a^\dagger$ is well estimated by the OPT-SIRS (green line) from the very beginning of the time window $[0, T_f]$, but precisely estimated by the EnKF only after the first 60 ms (red line), and by the BF only after 200 ms in the time window (not visible here). This suggests that the time window length J plays an important role affecting the estimation quality of unobserved variables.

In Fig. 2 the parameter component of the filtering mean is plotted in $[0, T_f]$. It shows that the errors of almost all parameters estimated by the EnKF are relatively large at the beginning, but they do approach zero by the end of the data assimilation window. For neural parameters g_{L}, E_{L}, $V_{1/2}^{(b)}$, $K^{(b)}$, and $K^{(a)}$, the OPT-SIRS produces parameter errors comparable to the EnKF, but the errors for the other parameters are larger. However, in this run the BF gives the lowest performance with only few parameter errors tending to zero.

In addition, it can be observed that filtering parameter errors generally pass through some initial transient states, and eventually stabilize on some asymptotic value. This fact is exploited in order to get a scalar estimate of every parameter by an average estimator $\tilde{\theta} = \frac{1}{J-J_1+1}\sum_{j=J_1}^{J} m_{\theta,j}$. We take $J_1 = 35000$ iterations in practice, i.e. the average is performed over the last three tenths of $[0, T_f]$.

TABLE II
Mean and standard deviation of estimated parameters

$\tilde{\theta}$		EnKF	BF	OPT-SIRS	`minAone`
g_{Na}	mean	18.56	20.74	19.86	20.77
	st. dev.	1.41	4.40	3.80	
E_{Na}	mean	61.43	60.23	59.99	58.86
	st. dev.	2.11	4.76	5.00	
g_{K}	mean	9.99	12.97	12.41	9.99
	st. dev.	0.06	3.55	3.89	
E_{K}	mean	−89.90	−90.62	−90.17	−90.00
	st. dev.	0.25	4.73	4.72	
g_{L}	mean	7.65	7.14	6.50	8.10
	st. dev.	0.35	3.72	3.10	
E_{L}	mean	−77.27	−78.58	−77.31	−78.25
	st. dev.	0.63	4.41	4.83	
$V_{1/2}^{(b)}$	mean	−20.77	−20.89	−19.96	−19.58
	st. dev.	0.75	5.02	4.83	
$K^{(b)}$	mean	14.61	16.84	16.24	15.14
	st. dev.	0.35	2.53	2.72	
$V_{1/2}^{(a)}$	mean	−45.01	−44.65	−44.34	−45.00
	st. dev.	0.05	5.02	4.87	
$K^{(a)}$	mean	4.92	5.92	6.77	5.00
	st. dev.	0.05	4.20	4.07	

A more complete analysis of the estimation results is available in TABLE II, where the estimated parameters' means and the standard deviations are listed. Here, such values are computed by evaluating the sample mean and standard deviation from the 100 independent runs we launched for every method. Also, the last column displays the estimated parameter values provided by `minAone`.

For every parameter value, we performed a one-sample t-test to check whether the difference between the mean estimate and the corresponding true value is statistically significant. Results proved that for EnKF this is indeed the case (for all parameters

TABLE III
MEAN OF PARAMETER ESTIMATION RELATIVE ERROR

$\lvert\frac{\tilde{\theta}-\theta^\dagger}{\theta^\dagger}\rvert$	**EnKF**	**BF**	**OPT-SIRS**	minAone
g_{Na}	$8.28\cdot10^{-2}$	$1.79\cdot10^{-1}$	$1.43\cdot10^{-1}$	$3.83\cdot10^{-2}$
E_{Na}	$3.34\cdot10^{-2}$	$6.10\cdot10^{-2}$	$6.98\cdot10^{-2}$	$1.90\cdot10^{-2}$
g_{K}	$3.90\cdot10^{-3}$	$3.62\cdot10^{-1}$	$3.65\cdot10^{-1}$	$8.51\cdot10^{-4}$
E_{K}	$1.99\cdot10^{-3}$	$4.38\cdot10^{-2}$	$4.29\cdot10^{-2}$	$6.56\cdot10^{-6}$
g_{L}	$5.12\cdot10^{-2}$	$3.82\cdot10^{-1}$	$3.47\cdot10^{-1}$	$1.30\cdot10^{-2}$
E_{L}	$1.04\cdot10^{-2}$	$4.46\cdot10^{-2}$	$5.19\cdot10^{-2}$	$3.26\cdot10^{-3}$
$V_{1/2}^{(b)}$	$4.36\cdot10^{-2}$	$2.05\cdot10^{-1}$	$1.89\cdot10^{-1}$	$2.11\cdot10^{-2}$
$K^{(b)}$	$2.94\cdot10^{-2}$	$1.72\cdot10^{-1}$	$1.54\cdot10^{-1}$	$9.39\cdot10^{-3}$
$V_{1/2}^{(a)}$	$8.81\cdot10^{-4}$	$8.92\cdot10^{-2}$	$9.09\cdot10^{-2}$	$1.07\cdot10^{-4}$
$K^{(a)}$	$1.71\cdot10^{-2}$	$6.70\cdot10^{-1}$	$6.94\cdot10^{-1}$	$4.41\cdot10^{-4}$
Average	$2.75\cdot10^{-2}$	$2.21\cdot10^{-1}$	$2.15\cdot10^{-1}$	$1.06\cdot10^{-2}$

$p < 5\%$). On the other hand, the same tests for both the BF and the OPT-SIRS showed that this difference is statistically significant only for four out of ten parameter values (g_{K}, g_{L}, $K^{(b)}$, $K^{(a)}$).

As for variability, the standard deviations of EnKF's estimate are significantly smaller than those of BF or OPT-SIRS. This is further confirmed observing that the average coefficient of variation[1] of the EnKF ($\overline{CV}_{EnKF} = 0.024$) is one order of magnitude smaller than both BF's ($\overline{CV}_{BF} = 0.240$) and OPT-SIRS's coefficient of variation ($\overline{CV}_{OPT} = 0.230$). In fact, the large difference in standard deviations could explain the result of t-tests.

TABLE III further investigates the matter of parameter estimation preciseness by presenting each method's mean relative error for every parameter value. Overall, the EnKF performs substantially better than both particle filters. In fact, the average mean relative error of the EnKF is about 8 times smaller than both BF's and OPT-SIRS's average error. The Friedman test with the method as column effect, and the ten parameters as row effect, it was proved that the difference between EnKF's relative error and both particle filters is indeed statistically significant (p-value smaller than 5%), whereas the difference between the BF and the OPT-SIRS is not.

Prediction

For every filtering method, the parameter estimator $\tilde{\theta}$ was validated by checking whether it could be used to give a good prediction of the model dynamics beyond the original time window $[0, T_f]$. In practice, we firstly partitioned the whole time window into two equal intervals, set $T_i := \frac{T_f}{2} = J_i\,\Delta t$, and named the resulting time interval $[T_i, T_f]$ *generalization time window*. System (9) was then solved numerically in this interval using the estimated parameter values and with initial data $(V(T_i), a(T_i))^T = m_{J_i,X}$. We write $\tilde{x}_j = (\tilde{V}_j, \tilde{a}_j)^T$ to denote the estimated trajectory at time j.

Fig. 3 shows that the EnKF overlaps almost perfectly the true trajectory, whereas both the BF and the OPT-SIRS have a rather similar profile which is close to the true one, but not as much as the EnKF.

[1]The coefficient of variation (CV) of a random variable with mean μ and standard deviation σ is defined as $CV = \frac{\sigma}{|\mu|}$

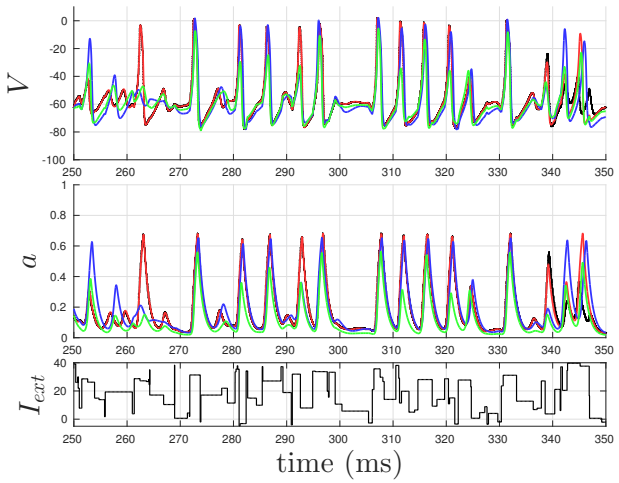

Fig. 3. Forecast skill in the first 100 ms of the generalization time window $[T_i, T_f]$ of the ensemble Kalman filter (red line) the bootstrap filter (blue line) and the optimal sequential importance re-sampling (green line) along with the true trajectory (black line in upper panels) and the time-dependent input current $I_{ext}(t)$ (black line in lower panel)

TABLE IV
MEAN L^1-ERROR IN GENERALIZATION AND PREDICTION TIME WINDOW (TOP PANEL) AND RELATIVE ESTIMATION ERROR (LOWER PANEL)

$\sum_j \lvert\tilde{x}_j - x_j^\dagger\rvert\Delta t$		**EnKF**	**BF**	**OPT-SIRS**
Generalization	V (mean)	223.3	4783.4	4576.1
window	a (mean)	2.2	59.2	54.6
Prediction	V (mean)	807.3	19075.2	18133.0
window	a (mean)	7.8	236.8	217.7

$\frac{\sum_j \lvert\tilde{V}_j - V_j^\dagger\rvert}{\sum_j \lvert\tilde{V}_j - V_j^\dagger\rvert + \sum_j \lvert V_j^\dagger - y_j\rvert}$		**EnKF**	**BF**	**OPT-SIRS**
Generalization	mean	52.21%	95.38%	95.13%
window	st. dev.	8.62%	1.69%	1.80%
Prediction	mean	48.97%	95.35%	95.10%
window	st. dev.	8.33%	1.73%	1.82%

After verifying the parameter estimator $\tilde{\theta}$ produced a good estimate of the system dynamics within the data assimilation window, prediction was further investigated. Estimates over the generalization time window proceeded "in the future" over the *prediction time window* $[T_f, 3\,T_f]$ and tested against the continuation of the true solution. Notice that the prediction time window is twice as long as $[0, T_f]$.

Quantitative measures of how close estimated trajectories are to the true one are presented in TABLE IV. These include the mean L^1-error[2] for each variable in the generalization window (upper panel, first row), in the prediction window (second row), and the mean and standard deviation of the

[2]The $L^1([T_i, T_f])$-error of the membrane potential component over the generalization time window is defined as $d_1(\tilde{V}, V^\dagger) = \int_{T_i}^{T_f} |V(t) - V^\dagger(t)|dt \approx \sum_{j=J_i}^{J} |\tilde{V}_j - V_j^\dagger|\Delta t$. The right hand side is the way this error is actually computed and it represents the value of the integral approximated by the Euler integration method. $L^1([T_i, T_f])$-errors for variable a and $L^1([T_f, 3\,T_f])$-errors in the prediction time window are defined analogously.

normalized d_N-error[3] in both generalization and prediction time windows (lower panel).

The top panel shows that the EnKF presents a cumulative mean $L^1([T_i, 3\,T_f])$-error of variable V which is $1/22$ of the OPT-SIRS and $1/23$ of the BF. In addition, the OPT-SIRS presents a mean L^1-error of variable a which is smaller than the BF in both generalization and prediction time window, but still one order of magnitude larger than the EnKF. Nonetheless, no statistically significant difference between the BF and the OPT-SIRS was detected applying the Friedman test.

Remarkably, in the lower panel of TABLE IV, the first column shows that the EnKF's mean relative error is approximately 0.5, i.e. in average, the L^1-error is as large as the truth-data error. This means that the EnKF produces such a good estimate that the mean L^1-error is comparable to the measurement error. This is true in the generalization window, but also in the prediction time window. Recall that the dummy data set in the prediction time window were not used in the DA methods application. The relatively small standard deviations prove the robustness of this result. Again, the BF and the OPT-SIRS are hardly distinguishable, with a much larger relative error than the EnKF and a small standard deviation.

V. Discussion and conclusions

In this paper, we compared three filtering data assimilation methods in a problem of simultaneous parameters and unmeasured variables estimation of a neuronal model. This study was performed in a twin experiment setting in which data were artificially generated by numerical simulation of the neural model. Our results demonstrate that the ensemble Kalman filter, the bootstrap filter and the optimal sequential importance re-sampling are all suitable methods for parameter estimation and they all possess the capability to predict the future activity of a single neuron.

As we saw, t-tests showed the mean EnKF's estimate is significantly different from the true value. On the contrary, the known fact that particle filters can recover the true filtering distribution in the limit for $N \to \infty$ is consistent with BF and OPT-SIRS being unbiased. Nonetheless, there is no guarantee that a particle filter can provide a better performance in any single realization.

In fact, the EnKF is by far the best of these methods in the more telling task of signal estimation prediction. Both particle filters provide similar results, with the OPT-SIRS performing slightly better than the BF but with no statistically significant difference between them. Besides, further analysis of the parameter estimation performance demonstrated that the EnKF has a much smaller relative error that the two particle filters.

In addition, in the Results section we found hints that the data-assimilation-window length J can be an important factor for estimation accuracy. However, it should be highlighted that in all simulations that we ran the performance of the EnKF was robust with respect to J and other preassigned quantities such as $m_{X,0}$, $m_{\theta,0}$, $C_{X,0}$ and $C_{\theta,0}$.

The computation loads were also compared. It was shown that all filtering methods consume similar computing CPU time for each run ($2\,\mathrm{min}\,47\,\mathrm{s} \pm 12\,\mathrm{s}$ for the EnKF, $2\,\mathrm{min}\,14\,\mathrm{s} \pm 7\,\mathrm{s}$ for the BF, and $2\,\mathrm{min}\,17\,\mathrm{s} \pm 5\,\mathrm{s}$ for the OPT-SIRS). Note that we took advantage of the automatic parallelization of Matlab 2015b.

It was not surprising that the smoothing method `minAone` can estimate parameters with very good accuracy, since all historic data are used in variational calculus. However, the computational time consumption of `minAone` is much higher than the filtering methods. As we found, a single run took more than $2\,\mathrm{h}$. Therefore, with a balance of accuracy of estimation and economy of the computing time, we conclude that the EnKF is the best choice in this example.

In our future work, we plan to further develop the application of Bayesian DA methods in computational neuroscience, investigating some more biologically accurate neuron models and a neuronal network model of a large number of neurons with experimental data.

[3] We define the *relative-error metric* d_N as the L^1-error for variable V, normalized with the intrinsic truth-data error $d_1(V^\dagger, y)$, i.e.

$$d_N(\tilde{V}, V^\dagger) = \frac{d_1(\tilde{V}, V^\dagger)}{d_1(\tilde{V}, V^\dagger) + d_1(V^\dagger, y)}.$$

This distance is a $[0, 1)$-valued function which tends to one if $d_1(\tilde{V}, V^\dagger) \gg d_1(V^\dagger, y)$, and approaches zero if the estimation error is much smaller than the model-intrinsic measurement error. Note that in the prediction time window a new dummy data set is generated from the continuation of the true solution.

References

[1] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve", *J. of Physiology*, vol. 117, pp. 500–44, 1952.

[2] V. Cutsuridis, S. Cobb and B. P. Graham, "Encoding and Retrieval in a Model of the Hippocampal CA1 Microcircuit", *Hippocampus*, vol. 20, pp. 423–446, 2010.

[3] H. Markram *et al.*, "Reconstruction and Simulation of Neocortical Microcircuitry", *Cell*, vol. 163, pp 456–492, 2015.

[4] C. L. Keppenne, M. M. Rienecker, N. P. Kurkowski and D. A. Adamec, "Ensemble Kalman filter assimilation of temperature and altimeter data with bias correction and application to seasonal prediction", *Nonlinear Processes in Geophysics*, vol. 12, pp. 491–503, 2005.

[5] R. H. Reichle, "Data assimilation methods in the Earth sciences", *Advances in water resources*, vol. 31, pp. 1411–1418, 2008.

[6] M. G. Shirangi. History matching production data and uncertainty assessment with an efficient TSVD parameterization algorithm. *J. Petroleum Sci. & Eng.*, vol.113, pp. 54–71, 2014.

[7] J. A. Hansen and C. Penland, "On stochastic parameter estimation using data assimilation", *Physica D*, vol. 230, pp. 888, 2007. doi:10.1016/j.physd.2006.11.006

[8] J. O. Ramsay, G. Hooker, D. Campbell and J. Cao, "Parameter estimation for differential equations: a generalized smoothing approach", *J. of Roy. Statistical. Soc.* ser. B, vol. 69, no. 5 , pp. 741–796, 2007.

[9] P. J. Smith, S. L. Dance and N. K. Nichols, "A hybrid data assimilation scheme for model parameter estimation: Application to morphodynamic modelling", *Comput. and Fluids*, vol. 46, pp. 436–441, 2011.

[10] G. Evensen. Sequential data assimilation. Springer, Berlin Heidelberg, 2009.

[11] K. J. H. Law, A. M. Stuart and K. C. Zygalakis, *Data Assimilation: A Mathematical Introduction* (Texts in Applied Mathematics, vol. 62). New York: Springer, 2015.

[12] G. Kitagawa, "A Self-Organizing State-Space Mode", *J. of the Amer. Statist. Assoc.*, vol. 93, pp. 1203–1215, 1998.

[13] D. V. Vavoulis, V. A. Straub, J. A. D. Aston and J. Feng, "A Self-Organizing State-Space-Model Approach for Parameter Estimation in Hodgkin-Huxley-Type Models of Single Neurons", *PLoS Computational Biology*, vol. 8, no. 3, pp. e1002401, 2012.

[14] A. C. Lorenc and T. Payne "4D-Var and the butterfly effect: Statistical four-dimensional data assimilation for a wide range of scales", *Quart. J. of the Roy. Meteorological Soc.*, vol. 133, pp. 607–614, 2007.

[15] H. D. I. Abarbanel, *Predicting the Future: Completing Models of Observed Complex Systems*. New York: Springer, 2013.

[16] B. A. Toth, M. Kostuk, C. D. Meliza, D. Margoliash and H. D. I. Abarbanel, "Dynamical estimation of neuron and network properties I: variational methods", *Biological Cybernetics*, vol. 105, pp. 217–237, 2011. doi:10.1007/s00422-011-0459-1

[17] M. Kostuk, B. A. Toth, C. D. Meliza, D. Margoliash and H. D. I. Abarbanel, "Dynamical estimation of neuron and network properties II: path integral Monte Carlo methods", *Biological Cybernetics*, vol. 106, pp. 155–167, 2012. doi:10.1007/s00422-012-0487-5

[18] R. E. Kalman, "A new approach to linear filtering and prediction problems", *J. Basic. Eng.*, ser. D, vol. 82, pp. 35–45, 1960.

[19] A. Doucet, N. de Freitas and N. Gordon, *Sequential Monte Carlo Methods in Practice*, New York: Springer-Verlag, 2001.

[20] A. Docuet, S. Godsill and C. Andriey, "On sequential Monte Carlo sampling methods for Bayesian filtering", *Stat. and Computing*, vol. 10, pp. 197–208, 2000.

[21] J. Ye, N. Kadakia, P. J. Rozdeba, H. D. I. Abarbanel and J. C. Quinn, "Improved variational methods in statistical data assimilation", *Nonlinear Processes in Geophysics*, vol. 22, pp. 205–213, 2015.

[22] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming", *Math. Programming*, vol. 106, pp. 25–57, 2006.

[23] E.M. Izhikevich, *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting* (Computational Neuroscience). Cambridge MA: MIT Press, 2007, ch. 4, pp. 89–158.