Tellus SERIES A DYNAMIC METEOROLOGY AND OCEANOGRAPHY
PUBLISHED BY THE INTERNATIONAL METEOROLOGICAL INSTITUTE IN STOCKHOLM

Using global Bayesian optimization in ensemble data assimilation: parameter estimation, tuning localization and inflation, or all of the above

By SPENCER LUNDERMAN[1]*, MATTHIAS MORZFELD[2], and DEREK J. POSSELT[3],
[1]*Department of Mathematics, University of Arizona, Tucson, AZ, USA;* [2]*Institute of Geophysics and Planetary Physics, Scripps Institution of Oceanography, University of California, San Diego, CA, USA;* [3]*Jet Propulsion LaboratoryCalifornia Institute of Technology, Pasadena, CA, USA*

(Manuscript Received 13 November 2020; in final form 21 March 2021)

ABSTRACT

Global Bayesian optimization (GBO) is a derivative-free optimization method that is used widely in the tech-industry to optimize objective functions that are expensive to evaluate, numerically or otherwise. We discuss the use of GBO in ensemble data assimilation (DA), where the goal is to update the state of a numerical model in view of noisy observations. Specifically, we consider three tasks: (i) the estimation of model parameters; (ii) the tuning of localization and inflation in ensemble DA; (iii) doing both, i.e. estimating model parameters while simultaneously tuning the localization and inflation of the ensemble DA. For all three tasks, the GBO works 'offline', i.e. a set of 'training' observations are used within GBO to determine appropriate model or localization/inflation parameters, which are subsequently deployed within an ensemble DA system. Because of the offline nature of the technique, GBO can easily be combined with existing DA systems and it can effectively decouple (nearly) linear/Gaussian aspects of a problem from highly nonlinear/non-Gaussian ones. We illustrate the use of GBO in simple numerical experiments with the classical Lorenz problems. Our main goals are to introduce GBO in the context of ensemble DA and to spark an interest in GBO and its uses for streamlining important tasks in ensemble DA.

Keywords: data assimilation, global Bayesian optimization, parameter estimation, ensemble Kalman filtering

1. Introduction

Global Bayesian optimization (GBO) is an optimization method that is used widely in the tech-industry, e.g. for tuning parameters which define advertising schemes on web pages such as Yelp, or for optimizing server compiling flags at Facebook (Letham et al., 2019), or for tuning machine learning algorithms (Snoek et al., 2012). These engineering applications have in common that evaluations of the objective function (the function one needs to optimize) are 'expensive', either computationally, e.g. when tuning machine learning algorithms, or quite literally, e.g. when new advertisement arrangements are tested on a subset of website users.

The fact that GBO has been successful in such difficult problems suggests that it can find (near) optimal solutions after only a small number of evaluations of the objective function. The strategy with which GBO achieves a near optimal solution is constructing, refining and optimizing an emulator of the objective function (the emulator is typically a Gaussian process model). The power of GBO then stems from judiciously choosing *where* to evaluate the objective function to refine and improve the emulator. This is done by resolving a trade-off between exploring regions where the objective function is known to be large, and exploring unknown regions to encounter new, and possibly more relevant, extrema of the objective function. We note that the main computational cost of GBO arises from evaluations of the objective function (all other computations are manageable), and that no derivative or adjoints are required.

In short, GBO is an optimization method that can be used for the efficient numerical solution of optimization problems for which

i. derivatives and adjoints are difficult or impossible to compute;

*Corresponding author. e-mail: lunderman@math.arizona.edu

Citation: Tellus A: 2021, **73**, 1924952, https://doi.org/10.1080/16000870.2021.1924952

ii. evaluations of the objective function are expensive and a near optimal, or at least useful, solution can be found after only a few evaluations of the objective function.

Optimization problems with the above two characteristics occur frequently in the Earth sciences and it is natural to ask if and how one may be able to use GBO in Earth science problems. An example of the use of GBO in a geophysical problem is the recent work of Pirot et al. (2019), where GBO is used for underground contaminant source localization. A second example is the work of Abbas et al. (2016), which leverages GBO in the context of simultaneous state and parameter estimation problems.

In this paper, we continue this line of work and present numerical experiments in which we test the usefulness of GBO in three problems in ensemble data assimilation (DA). The goal in DA is to update forecasts of a numerical model by observations. A prominent example is (global) numerical weather prediction where model-based forecasts are updated in view of meteorological observations. The DA problem is usually formulated within a Bayesian framework, in which the forecast defines a prior, which is updated to a posterior distribution via a likelihood that takes the observations into account. Ensemble DA are a suite of numerical methods that use the Monte Carlo technique to perform this inference. Examples of ensemble DA methods include the ensemble Kalman filter (EnKF), see e.g. Evensen (2009a), and particle filters, see, e.g. Doucet et al. (2001) and Farchi and Bocquet (2018). Variational methods, which aim at finding the model state that maximizes the posterior distribution, are an alternative to ensemble DA (Talagrand and Courtier, 1987). Variational methods have been combined with the ensemble approach in hybrid-schemes (Lorenc, 2003; Bonavita et al., 2012).

The computational requirements of ensemble DA are connected to the required ensemble size: each ensemble member typically requires a simulation of the underlying computational model, which is expensive. For this reason, one must keep the ensemble size small. A small ensemble size, in turn, introduces sampling errors which can be reduced by 'localization' and 'inflation'. The essential idea of localization is to reduce sampling error by deleting spurious long-range correlations (Hamill et al., 2001; Houtekamer and Mitchell, 2001). Inflation is another technique used to compensate for the tendency of ensemble DA schemes to underestimate actual errors, see, e.g. Kotsuki et al. (2017) and Anderson and Anderson (1999). The basic idea in inflation is to artificially increase the variances and covariances in the ensemble. The various parameters that define the localization and inflation are 'tuned', which often means that one performs tests with a limited number of combinations of localization and inflation parameters, and then uses the ones that gave the best results (see, e.g. Lorenc (2017) [Section 3]). We refer to this technique of optimizing localization and inflation parameters as a 'grid-search' (because one searches for a good solution on a grid of parameters). Alternatives to a grid-search, including data-driven, adaptive and optimal localization methods, are discussed in Section 3.2.

The rest of this paper is dedicated to investigating the use of GBO within ensemble DA by describing how to use GBO in three specific problems:

1. tuning localization and inflation in ensemble DA;
2. solving DA problems where some or all model parameters are uncertain or completely unknown;
3. tuning localization and inflation, while simultaneously also estimating model parameters.

The first problem is motivated by the fact that tuning localization and inflation becomes increasingly difficult as new multi-scale localization schemes are put to use, see, e.g. Buehner (2012) and Buehner and Shlyaeva (2015). For multi scale localization methods, the number of parameters that need to be tuned may exceed what is feasible to do with simple grid-search techniques and GBO may represent a useful alternative. Data-driven or adaptive/optimal localization schemes have not yet been put forward for multi scale localization.

The second problem is motivated by the fact that, on short enough time intervals, linear or nearly linear ensemble DA methods (EnKF or hybrid methods) are successful because the state estimation is nearly linear on such short time scales, see e.g. Morzfeld and Hodyss (2019). Parameter estimation, however, is typically highly nonlinear, see, e.g. Vukicevic and Posselt (2008) and Posselt et al. (2012). Combining GBO with ensemble DA allows us to separate the simultaneous state/parameter estimation problem into its (approximately) linear and nonlinear components. Specifically, we use ensemble DA for the state estimation and, using GBO, create an additional outer loop to find optimal values for the model's parameters. The third problem is motivated by the fact that the localization and inflation parameters depend on the model parameters. For example, a larger inflation can compensate for errors in the model parameters (because a large inflation essentially means to increase model errors compared to observation errors). On the other hand, if one finds appropriate model parameters, large forecast errors can occur if the localization/inflation parameters are inappropriate.

We present numerical experiments with the usual Lorenz test problems (Lorenz, 1963, 1996) and with an EnKF (Evensen, 2009a) serving as an example of an ensemble DA system. Our study is a proof-of-concept, but more work is needed, in particular with more realistic or real models, to better understand the benefits and

limitations of the use of GBO within ensemble DA. It is clear, however, that the GBO framework is flexible and easy to use with any ensemble DA technique (variational or hybrid). The reason is GBO's 'offline' structure, i.e. the fact that GBO operates in an outer loop in which one calls an ensemble DA for objective function evaluations. Moreover, the overall computational cost of a GBO technique is determined by the computational cost of evaluations of the objective function (using an ensemble DA system on a set of observations). All other computations required (updating/optimizing the emulator and computing where to evaluate the objective function) are negligible.

Before proceeding, we summarize some notation we use throughout this paper. Lowercase bold characters represent vectors, lowercase non-bold characters are scalars, and capital bold characters represent matrices. The dimensions are denoted by N with an index that references the vector, e.g. N_a is the dimension of the vector $\mathbf{a}$. We also use a condensed indexing notation where $\boldsymbol{b}^{1:n}$ should be interpreted as the set $\{\boldsymbol{b}^1, ..., \boldsymbol{b}^n\}$. This indexing notation extends to scalar functions by interpreting $f(\boldsymbol{b}^{1:n})$ as the column vector $[f(\boldsymbol{b}^1), ..., f(\boldsymbol{b}^n)]^T$.

2. Global Bayesian optimization

In this section, we review GBO from a mathematical perspective. As indicated in the introduction, GBO is a derivative-free optimization method, which means that no gradients or adjoints are required. The optimization relies solely on evaluations of the objective function. The optimization of the objective function is achieved by building and refining an emulator of the objective function. Where the function evaluations take place is decided by resolving a trade-off between refining the emulator in regions where the objective function is known to be large, and exploring unknown regions.

Throughout this paper, we denote the objective function by $J(\boldsymbol{\phi})$, where $\boldsymbol{\phi}$ is typically vector valued, but the objective function itself is scalar. We assume that $J(\boldsymbol{\phi}) \leq 0$ for any $\boldsymbol{\phi}$ and, thus, we seek to *maximize* the objective function. In all cases we consider, the objective functions is connected to a nonlinear least squares fit to observations (see Equation (25)). GBO computes the maximizer $\hat{\boldsymbol{\phi}} = \arg\max(J)$ by iterating two steps. Step one builds the emulator based on all available function evaluations, and step two proposes a new parameter $\boldsymbol{\phi}^*$, which is determined by optimizing an 'acquisition function'. Optimization of the acquisition function resolves the trade-off between refining the emulator in regions where J is known to be large, and exploring the parameter space more broadly to find new and higher maxima. During the iteration, one learns the structure of the objective function and can make informed guesses about where the function is large (ideally maximally large). We now describe each of the key GBO ingredients: The emulator, which is usually a Gaussian process (GP) model, its update via regression, and the acquisition function. For simplicity, we present the case of a scalar parameter ϕ, but the extension to vector-valued parameters is straightforward. We refer to Frazier and Wang (2016) for more details about GBO.

2.1. Gaussian processes and their updates by regression

A Gaussian process is a collection of random variables such that any finite subset has a multivariate Gaussian distribution. A simple example of a GP is a multivariate Gaussian random variable (this corresponds to a finite indexing set). Allowing the indexing set to be (uncountably) infinite, one can extend the notion of Gaussian distributions to random functions. In practice, one always uses finite dimensional approximations of functions by discretizing on a grid, but the theory is easier to grasp when considering the infinite dimensional limit, which is independent of the discretization one may chose.

A GP is defined by its mean function and a kernel function, that describes the covariances within the function. A kernel $\mathcal{K}$ is a positive definite function which defines the spatial correlation between any two points within the domain. There are many examples of suitable kernel functions, see, e.g. Williams and Rasmussen (2006), and one popular choice is a Gaussian kernel with added white noise

$$\mathcal{K}(\phi, \phi') = \gamma^2 \cdot \exp\left(-\frac{|\phi - \phi'|^2}{2\ell^2}\right) + \kappa^2 \cdot \delta(\phi, \phi') \qquad (1)$$

where $\delta(\cdot,\cdot)$ is the Dirac delta function and where γ, κ and ℓ are positive parameters that define the kernel. We discuss how to find 'hyper-parameters' γ, κ and ℓ further below. To keep things simple, and since this study presents a proof-of-concept, we will use only the above kernel, but one can easily envision using other kernels, or even optimizing the kernel structure (see, e.g. Lunderman et al. (2018)).

One then emulates, or models, the objective function by a Gaussian process with the specified kernel:

$$\hat{J}(\phi) = \mathcal{GP}(\mu(\phi), \mathcal{K}(\phi, \phi')), \qquad (2)$$

and updates the GP using evaluations of the objective function J and regression. Suppose we have evaluated the objective function at N_G parameters $\phi^{1:N_G}$, then regression gives

$$\hat{J}(\phi|\phi^{1:N_G}) = \mathcal{GP}(\hat{\mu}(\phi), \hat{\mathcal{K}}(\phi, \phi')), \qquad (3)$$

where

$$\hat{\mu}(\phi) = \mu(\phi) + \mathcal{K}(\phi, \phi^{1:N_G})\mathcal{K}(\phi^{1:N_G}, \phi^{1:N_G})^{-1}\left[J(\phi^{1:N_G}) - \mu(\phi^{1:N_G})\right], \tag{4}$$

$$\hat{\mathcal{K}}(\phi, \phi') = \mathcal{K}(\phi, \phi) - \mathcal{K}(\phi, \phi^{1:N_G})\mathcal{K}(\phi^{1:N_G}, \phi^{1:N_G})^{-1}\mathcal{K}(\phi^{1:N_G}, \phi). \tag{5}$$

This (Bayesian) update can be implemented efficiently via linear algebra and since one can process each function evaluation sequentially, the linear algebra is relatively straightforward (see Rasmussen (2004) for more details).

The hyper-parameters (γ, κ and ℓ) can be obtained by maximizing the negative log marginal likelihood defined as

$$\begin{aligned}\mathcal{L}(\gamma, \kappa, \ell) &= -\log\ p(J(\phi^{1:N_G})|\ \gamma, \kappa, \ell) \\ &= \frac{1}{2}\left[J(\phi^{1:N_G})\right]^T \mathbf{K}^{-1}\left[J(\phi^{1:N_G})\right] + \frac{1}{2}\log|\mathbf{K}| \\ &\quad + \frac{N_G}{2}\log 2\pi,\end{aligned} \tag{6}$$

where $\mathbf{K} = \mathcal{K}(\phi^{1:N_G}, \phi^{1:N_G}; \gamma, \kappa, \ell)$ is an $N_G \times N_G$ matrix that discretizes the kernel (for a given set of γ, κ, ℓ) and where $|\mathbf{K}|$ is its determinant. It is advisable to re-do this optimization and adjust the hyper-parameters of the kernel *before* each update, since the function evaluations and updates may provide additional information about what appropriate hyper-parameters ought to be.

2.2. Acquisition function

We have described above how to emulate the objective function by a GP and how to update this GP in view evaluations of the objective function. What remains to do is to find a way to suggest new parameters at which to evaluate the objective function to improve its GP emulator. GBO finds new parameters for function evaluations via optimization of an acquisition function. An acquisition function, A, has the form

$$A(\phi) = \mathcal{A}(J^{\max}, \mu(\phi), \mathcal{K}(\phi, \phi')), \tag{7}$$

where $J^{\max} = \max_k\{J(\phi^k)\}$. The parameter ϕ^*, where the objective function is evaluated, is selected by maximizing the acquisition function over ϕ :

$$\phi^* = \arg\max_\phi A(J^{\max}, \mu(\phi), \mathcal{K}(\phi, \phi')). \tag{8}$$

In this work, we use Expected Improvement (EI) as the acquisition function because it is one of the simplest choices and it turned out to be sufficient in the problems we tried, but more sophisticated techniques, e.g. using knowledge-gradients, may be used in the future. EI is defined as

$$EI(\phi) = \mathbb{E}\left[(J(\phi) - J^{\max})^+\right], \tag{9}$$

where $(\cdot)^+ = \max\{\cdot, 0\}$ and $\mathbb{E}[\ \cdot\ ] = \mathbb{E}[\ \cdot\ |\ \phi^{1:N_G}, J(\phi^{1:N_G})]$ is the expectation conditioned on the known function realizations (see, e.g. Jones et al. (1998)). Under our assumptions, EI has the closed form representation

$$\begin{aligned}EI(\phi) &= (\mu(\phi) - J^{\max})\Phi\left(\frac{\mu(\phi) - J^{\max}}{\sigma(\phi)}\right) \\ &\quad + \sigma(\phi)\varphi\left(\frac{\mu(\phi) - J^{\max}}{\sigma(\phi)}\right),\end{aligned} \tag{10}$$

where $\Phi(\cdot)$ and $\varphi(\cdot)$ are the normal cumulative distribution function (cdf) and probability density function (pdf), and where $\sigma^2(\phi) = \mathcal{K}(\phi, \phi)$. Thus, maximizing EI is relatively straight forward.

Note that the maximizer of EI will have a large predicted value (relative to the known max $J^{\max}$) and a large amount of associated uncertainty (represented by $\sigma(\phi)$). Thus, using optimization of EI to propose new parameters for function evaluations resolves the trade-off between exploring the space where the objective is known to be large with exploring unkown and uncertain regions that may contain higher maxima (see also Frazier and Wang (2016), Section 3.4.1).

2.3. Summary of the GBO algorithm

The GBO algorithm can be summarized as follows. First, pick a GP model by specifying a kernel and associated hyper-parameters. Second, initialize the algorithm by picking a (small) number of points and evaluating the objective function at these points. For example, one can use a quasi-random Sobol sequence to obtain a few ϕ. One then optimizes the hyper-parameters of the kernel and uses regression to update the mean function and covariance kernel in view of the new function evaluations. The algorithms proceeds by iterating the following steps

1. Maximize expected improvement to obtain a ϕ^* and evaluate the objective function at ϕ^*.
2. Optimize the marginal log-likelihood to obtain appropriate hyper-parameters for the GP model's kernel in view of the new function evaluation.
3. Use regression to update the GP's mean function and covariance kernel model in view of all function evaluations done thus far.

This iteration proceeds until a stopping criterion is reached. In many engineering applications, the stopping criterion is simply 'we run GBO until we run out of time'. In the numerical examples below, we use a similar strategy and simply perform a pre-specified number of GBO steps. This corresponds to a priori specifying a computational budget and then accepting the results one obtains as appropriate, which is realistic. Other stopping criteria include monitoring on how the maximum function value evolves over the iterations and stopping after

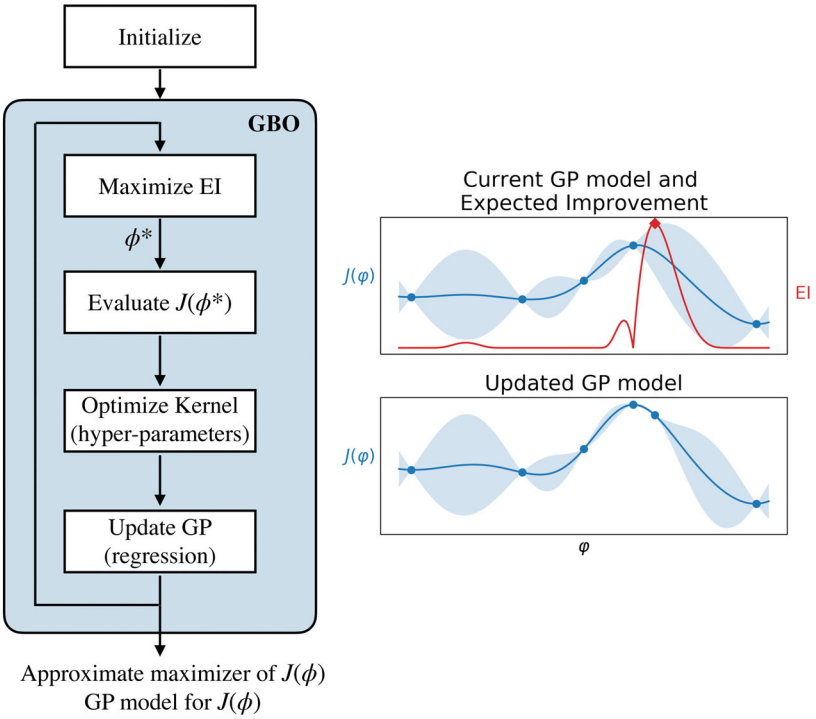

Fig. 1. Flowchart of the GBO algorithm. The two panels on the right illustrate (i) how maximization of expected improvement results in additional function evaluations that explore the space effectively; and (ii) the effects of a typical update of the GP mean and covariance based on a single, additional function evaluation.

reaching some threshold, or stopping when no improvement is made for a large number of iterations. Once the iteration is completed, the approximate maximizer of the objective function is

$$\hat{\phi} := \max_{\phi} J(\phi) \approx \max_{k} J(\phi^k). \tag{11}$$

The GBO algorithm is illustrated by a flowchart in Figure 1.

3. Review of ensemble data assimilation

The goal in data assimilation is to estimate the state of a numerical model at 'time' k, where k is an integer indexing a discrete time variable. The state estimate is based on previous observations, collected up to time $k-1$, a numerical model which we write as $\mathcal{M}$, and a new observation collected at time k. We assume that the model state and observation satisfy

$$\mathbf{y}_k = \mathbf{H}\mathbf{x}_k + \zeta_k, \tag{12}$$

where $\zeta_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ are independent identically distributed (iid) random variables and and $\mathbf{R}$ is the observation error covariance matrix; $\mathbf{H}$ is a matrix that maps the state to the observations, where the number of observations is usually much less than the number of state variables. We write the underlying numerical model as

$$\mathbf{x}_k = \mathcal{M}(\boldsymbol{\theta}; \mathbf{x}_{k-1}), \tag{13}$$

where $\boldsymbol{\theta}$ are model parameters. In 'pure' state estimation problems, the model parameters are assumed to be known, but one can extend the framework to account for partially or entirely unknown model parameters (see below). Making use of the Bayesian framework, the relevant posterior distribution for state estimation is

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) \propto p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1}). \tag{14}$$

It is now clear that data assimilation is a sequential process, in which state estimates are updated as observations become available, see, e.g. Carrassi et al. (2018) for more details and explanations.

3.1. Ensemble Kalman filter

The ensemble Kalman filter (EnKF) is an ensemble DA method that approximates the posterior distribution by using a Monte Carlo approximation within the Kalman filter formalism (Evensen (2009a)). Specifically, let $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{N_e}\}$ be an ensemble of size N_e, approximately distributed according to the distribution $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$. The model can be applied to each ensemble member to generate a *forecast ensemble*

$$\mathbf{x}_j^f = \mathcal{M}(\boldsymbol{\theta}; \mathbf{x}_j), \quad j = 1, ..., N_e. \tag{15}$$

The covariance matrix computed from the forecast ensemble, $\mathbf{P}^f$, is called the 'forecast covariance'. The forecast covariance and the observation $\mathbf{y}_k$ are used to generate an *analysis* ensemble

$$\mathbf{x}_j^a = \mathbf{x}_j^f + \mathbf{K}(\mathbf{y} - (\mathbf{H}\mathbf{x}_j^f + \boldsymbol{\zeta}_j)) \quad j = 1, ..., N_e, \tag{16}$$

where $\boldsymbol{\zeta}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ are iid random variables, and where

$$\mathbf{K} = \mathbf{P}^f\mathbf{H}^T(\mathbf{H}\mathbf{P}^f\mathbf{H}^T + \mathbf{R})^{-1}, \tag{17}$$

is the Kalman gain. The analysis ensemble is approximately distributed according to the posterior distribution $p(\mathbf{x}_k|\mathbf{y}_{1:k})$. When a next observation becomes available, the entire process is repeated.

The EnKF described here is referred to as the 'stochastic' implementation of the EnKF, and other implementations are also heavily used and referred to as ensemble transform filters, see, e.g. Hunt et al. (2007). The GBO techniques we describe below, pertain to all implementations of the EnKF, but we consider only the stochastic implementation as an example, and to illustrate our ideas.

3.2. Localization and inflation

When applying the EnKF, one must keep computational requirements in mind, especially if the model is complex and computationally intensive to run. In this case, a large component of the computational cost of an EnKF is generating the forecast ensemble by applying the model to each ensemble member. To keep the computational requirements low, the ensemble size should be chosen as small as possible and is typically on the order of hundreds, often several orders of magnitude smaller than the dimension of the state or the observations. Localization and inflation are two techniques that enable using the EnKF with modest ensemble sizes, and it is widely accepted that localization and inflation are critical components of an EnKF, see, e.g. Hamill et al. (2009).

A small ensemble size causes errors in the sample-based estimates of the forecast covariance matrix, which often results in unrealistic long-range correlations. The basic idea of *localization* is to reduce spurious correlations, see, e.g. Hamill et al. (2001) and Houtekamer and Mitchell (2001). There are several localization techniques one can use, see, e.g. Shlyaeva et al. (2019), but here we focus on localization of the forecast covariance by a Schur-, or element-wise, product. One first determines a symmetric positive definite (SPD) localization matrix $\mathbf{L}$ with the property that correlations of any pair of state variables that are 'far' from each other are dampened. A simple example is the matrix $\mathbf{L}$ whose elements $\mathbf{L}_{i,j}$, are given by

$$\mathbf{L}_{i,j} = \exp\left(-\frac{1}{2}\left(\frac{z_{i,j}}{L}\right)^2\right) \tag{18}$$

where $z_{i,j}$ is the distance between the i^{th} and j^{th} states and where L is a length scale that determines how quickly correlations decay. The localized forecast covariance is defined by

$$\mathbf{P}^f_{\text{loc}} = \mathbf{L}^{\circ}\mathbf{P}^f, \tag{19}$$

where the open circle, $^{\circ}$, denotes an element-wise product of $\mathbf{L}$ and $\mathbf{P}^f$. The localized forecast covariance $\mathbf{P}^f_{\text{loc}}$ then replaces the forecast covariance within the EnKF algorithm.

Inflation is another method that is used to make EnKF applicable and accurate by keeping the ensemble size small. More specifically, inflation is used to prevent underestimation of variances and covariances due to a small ensemble size, which may cause the EnKF to generate analysis ensembles whose covariance underestimates actual errors (Anderson and Anderson, 1999). In this work, we use *multiplicative* inflation as an example of one of many inflation techniques (see, e.g. Luo and Hoteit (2011) for an alternative). As above, we denote the forecast ensemble by $\mathbf{x}_1^f, ..., \mathbf{x}_{N_e}^f$ and write the forecast mean as $\bar{\mathbf{x}}^f = (1/N_e)\sum_{j=1}^{N_e}\mathbf{x}_j^f$. The *inflated* ensemble, which has a larger associated covariance matrix than the 'original' ensemble if $\alpha > 1$, is given by

$$\mathbf{x}^f_{j,\text{infl}} = \bar{\mathbf{x}}^f + \sqrt{\alpha}(\mathbf{x}_j^f - \bar{\mathbf{x}}^f). \tag{20}$$

The inflated ensemble $\mathbf{x}^f_{\text{infl}}$ is then updated via (16) within the EnKF algorithm, using the localized forecast covariance matrix when computing the Kalman gain.

We note that localization and inflation require that one specifies parameters that define the degree of localization (the length-scale L) and inflation (the inflation parameter α). Moreover, localization and inflation are usually intertwined: Fixing the inflation and optimizing the localization (or vice versa), does not yield results that are as good as a joint optimization over localization and inflation parameters. Perhaps the simplest technique to find appropriate localization/inflation parameters is to perform a 'grid-search', where one tries a number of pairs of localization/inflation parameters (often informed by the physics of the problem) and then uses the one that gives the best results, e.g. those that minimize EnKF forecast errors (see, e.g. Nerger (2015) and Lorenc (2017)). A grid-search requires that one performs a data assimilation experiment for each pair of localization/inflation parameters and, therefore, can accrue a large computational cost. In fact, the computational requirements of a grid-search may become excessive when multi-scale

localization techniques, which require several parameters to define the localization, are used (Buehner, 2012; Buehner and Shlyaeva, 2015).

Besides a grid search, one can also use a data-driven approach to obtain a suitable localization (De La Chevrotière and Harlim, 2017). The basic idea is to first construct a model for the localization of the forecast covariance during an 'offline training' procedure, and to then deploy this localization in the ensemble DA. The training involves performing a DA for a specified number of training observations. A data-driven approach is thus similar in spirit to the grid-search because a set of observations is required and processed during training. Similarly, one can also use machine learning to find appropriate localizations Moosavi et al. (2019). Empirical localization functions (Lei et al., 2015), can also be use and are constructed, empirically, from a training set of obserations.

The computational cost of a grid-search or data-driven method can be high, because a large number of DA are performed during the grid-search/training phase. Alternatively, one can use adaptive localization schemes where the localization is adapted in view of the observations (and assumed errors). Examples of adaptive localization techniques include Anderson (2012), Zhen and Zhang (2014), Anderson (2016) and Popov and Sandu (2019). The computational cost of an adaptive localization is low and we emphasize that no training is required when using an adaptive strategy. Optimal localization methods are similar in spirit to adaptive techniques, in that these methods can be employed directly (without using any training or optimization). In an optimal localization scheme, the localization is chosen to optimize a specified set of criteria. The optimality is based on mathematical assumptions, e.g. about sampling- or model-, or observation errors, and the validity of these assumptions is difficult to ensure in realistic DA problems. Examples of optimal localization methods include Ménétrier et al. (2015), Flowerdew (2015) and Destouches et al. (2020). The tuning of localization 'on-the-fly' (no training required) using information from past observations is discussed in Flowerdew (2017). Finally we note that one can also adaptively chose the inflation, see, e.g. Li et al. (2009) and El Gharamti et al. (2019).

3.3. Augmented state EnKF for simultaneous state and parameter estimation

The EnKF framework can be extended to simultaneous state and parameter estimation, where one estimates the state of the model, $\mathbf{x}$, and unknown model parameters $\boldsymbol{\theta}$, see e.g. Evensen (2009b), Bocquet and Sakov (2013) and Ruckstuhl and Janjić (2018). This technique, often called the *augmented state EnKF* method is as follows. One defines an 'augmented state'

$$\tilde{\mathbf{x}} = (\mathbf{x}^T, \boldsymbol{\theta}^T)^T \tag{21}$$

which contains the state and parameters. One then also augments the model by simply writing

$$\tilde{\mathcal{M}}(\tilde{\mathbf{x}}) = \begin{pmatrix} \mathcal{M}(\boldsymbol{\theta}) \\ \boldsymbol{\theta} \end{pmatrix}. \tag{22}$$

Thus, the augmented model evolves the state $\mathbf{x}$ via the numerical model $\mathcal{M}$ as before, and the model parameters $\boldsymbol{\theta}$ remain constant ($\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1}$). Since the relationship between parameters and observations is unknown, the observation matrix $\tilde{\mathbf{H}}$, is augmented with a zero matrix:

$$\tilde{\mathbf{H}} = (\mathbf{H} \quad \mathbf{0}). \tag{23}$$

The EnKF can now be used as described above and it generates estimates (in the form of ensembles) for the state $\mathbf{x}$ and the parameters $\boldsymbol{\theta}$. To localize an augmented state EnKF, one can, for example, augment the localization matrix by

$$\tilde{\mathbf{L}} = \begin{bmatrix} \mathbf{L} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix} \tag{24}$$

where $\mathbf{L}$ is the same localization matrix as before and where $\mathbf{B} = \mathbf{1}_{N_x \times N_\theta}$ (an $N_x \times N_\theta$ matrix whose elements are all equal to one) and $\mathbf{C} = \mathbf{I}_{N_\theta}$. More generally, one can consider localization matrices that dampen correlations between model states and model parameters in more sophisticated ways, but the tuning (or adapting, or optimally choosing) the augmented localization matrix can become cumbersome. Below, we will use the augmented state EnKF (with localization as above) as a benchmark method to test and verify the results we obtain via GBO applied to simultaneous state and parameter estimation problems.

Other numerical methods for simultaneous state and parameter estimation are often based on Monte Carlo (MC) sampling and particle filters. Particle filters are fully nonlinear/non-Gaussian state estimation algorithms. Examples of techniques that combine particle filters with Markov chain Monte Carlo (MCMC) for simultaneous state and parameter estimation include particle Markov chain Monte Carlo (PMCMC) (Andrieu et al., 2010) and SMC^2 (Chopin et al., 2013). One can envision using these algorithms within a GBO framework, but we leave such ideas for future work.

4. The use of GBO in ensemble DA

We use GBO to estimate model parameters, or to tune localization and inflation, or to do both. The basic idea is to couple GBO to an EnKF (or another ensemble DA

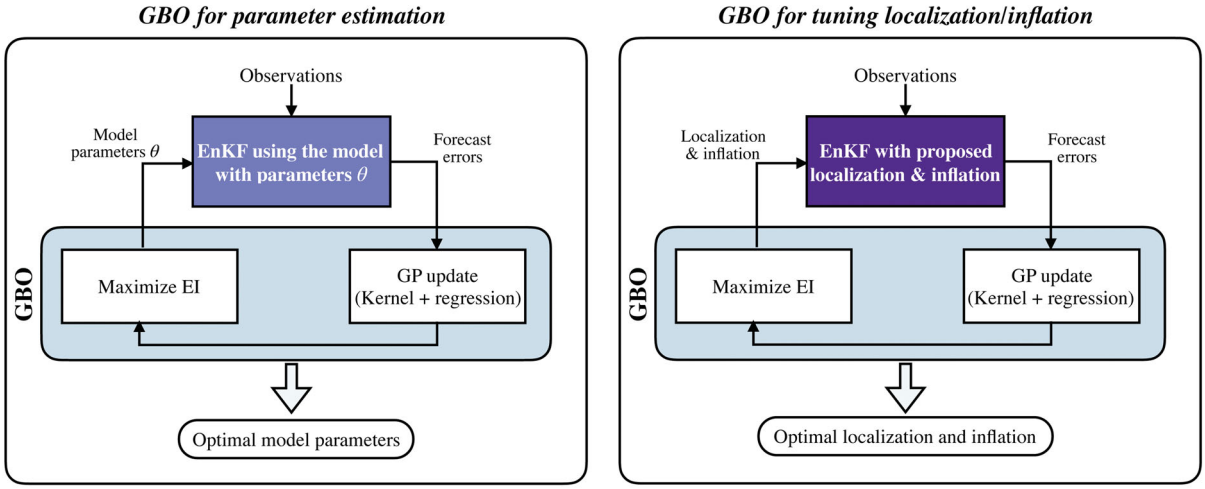

Fig. 2. Flowchart of the GBO used for parameter estimation (left) and tuning of localization/inflation (right). Note that the algorithms are essentially the same for both tasks, and differ only in the parameters being handed back and forth between the ensemble DA (EnKF) and the GBO iterations. We highlight this difference by using different colors for the parameter estimation (blue) and the tuning of localization/inflation (purple).

system) by defining the objective function by the forecast error of an EnKF, averaged over a pre-specified time-interval. More specifically, the objective function is defined as

$$J(\boldsymbol{\phi}) = -\frac{1}{N_T - N_B}\sum_{j=N_B}^{N_T}\left|\left|\mathbf{y}_j - \mathbf{H}\bar{\mathbf{x}}_j^f(\boldsymbol{\phi})\right|\right|_2^2, \tag{25}$$

where $||\cdot||_2$ is the 2-norm, $\mathbf{y}_j$ is the observation at time j, and $\bar{\mathbf{x}}_j^f$ is the ensemble average of the forecast ensemble at time j (the forecast ensemble at time j is the analysis ensemble at time $j-1$ propagated to time j via the model). We emphasize that the observations $\mathbf{y}_j$ have already been collected and serve as a 'training' set of observations. Using the training observations $\mathbf{y}_j$, GBO finds appropriate parameters and once training is complete, these parameters are deployed in an ensemble DA system. Moreover, the total number of training observations is N_T, but we disregard the first $N_B \geq 1$ observations when defining the objective function to account for a possible spin-up period of the EnKF, during which errors may be large because of effects of the initial ensemble.

In the definition of the objective function in (25), we use $\boldsymbol{\phi}$ to denote the parameters we are interested in, which may be the model parameters $\boldsymbol{\theta}$, the parameters that define the localization/inflation, or both (see below). Thus, the objective function we use within GBO is the same for all three tasks (parameter estimation, tuning of localization/inflation, or both), but the 'input' parameters, i.e. the variables we optimize over, vary for the three tasks. This is illustrated in Figure 2, where we show (simplified) flowcharts for using GBO for model parameter estimation and for tuning localization and inflation (see Figure 1 for a more detailed flowchart of the various steps in GBO). The third case, in which we estimate model parameters and localization and inflation, is self-explanatory upon inspection of these two flowcharts.

With the objective function defined in terms of the EnKF forecast errors, GBO operates as described above and finds a maximizer of the objective function by evaluating it a few times and by updating and revising the underlying emulator (the GP model of the objective function) in view of the function evaluations. Evaluation of the objective function requires that we apply the EnKF (with fixed ensemble size) to a set of N_T observations, distributed on the time interval $1, ..., N_T$. Evaluating the objective function is, thus, computationally expensive. In fact all other computations are negligible in comparison to the evaluation of the cost function. This is exactly the scenario GBO is designed for – optimizing a cost function that is expensive (computationally) to evaluate.

One advantage of GBO, that helps with dealing with this computational challenge, is its flexibility. Recall that the GBO update (expected improvement maximization, Kernel optimization, and regression) can be done *independently* of the EnKF analysis and the update does not require any more runs with the model. Thus, one can use an existing DA framework to compute the various forecast errors and then hand these errors over to a GBO code to perform the GBO update, which in turn provides

a new set of parameters to perform the DA with. One can in fact use different computers for the GBO update (which runs on a personal computer or workstation) and the DA (might require high performance computing resources). Moreover, by design, GBO should only run for a few function evaluations so that the required computations remain manageable.

4.1. Parameter estimation

When using GBO to estimate model parameters, one in fact performs a simultaneous state and parameter estimation for the training set of observations: The states are estimated by the EnKF, using the model parameters as determined by the GBO. Perhaps the main advantage of using GBO for the estimation of model parameters is its flexibility in terms of how the various computational tasks are performed. This flexibility allows us to effectively separate the linear/Gaussian characteristics of many state estimation problems, from the highly non-linear and non-Gaussian characteristics of parameter estimation problems, see, e.g. Vukicevic and Posselt (2008). As indicated in the introduction, on the typically short time intervals between successive observations, state estimation is a nearly linear problem, so that linear, or nearly linear, ensemble DA methods (such as the EnKF) can work well, see e.g. Morzfeld and Hodyss (2019). The highly nonlinear parameter estimation, however, is solved by an 'outer loop' of the GBO optimization, which does not rely on underlying assumptions of linearity or Gaussianity. One should carefully use the terminology here, because a Gaussian process model for the objective function, does not imply that this function is (nearly) Gaussian.

In applications with a large state dimension, the DA system should include a localization and inflation, even if the number of model parameters may be small. In this case, one can use adaptive or optimal techniques when implementing the DA system (since localization/inflation parameters are intertwined with model parameters, see also below). The flexibility of GBO is advantageous here because the objective function can be defined by forecast errors of an ensemble DA system that makes use of (adaptive) localization and inflation.

Finally we emphasize that while we only consider an EnKF for the state estimation in this work, it is straightforward to modify the framework we discuss to be used with other ensemble DA techniques, which include other versions of the EnKF, but also hybrid-variational/ensemble methods or even particle filters. Here, again, the flexibility of the GBO technique is important, because GBO is largely agnostic to how the state estimation is performed.

4.2. Tuning localization and inflation

If the model parameters are (assumed to be) known, one can use GBO to determine optimal, or at least appropriate, configurations of localization and inflation. The reason is that the localization and inflation has a large effect on the forecast errors in the EnKF and, for that reason, the GBO objective function is sensitive to the localization/inflation, so that GBO can be used to search for appropriate parameters that define the localization and inflation. Using GBO in this context is similar in spirit to a data-driven technique for localization, or finding empirical localization functions, since it requires repeatedly performing a DA on a training set of observations (each evaluation of the objective function in GBO requires a DA run). The computational cost of the training can be large, and is clearly larger than deploying adaptive or optimal localization strategyies which do not require any training. An advantage of using GBO could be that it couples the search for optimal localization *and* inflation parameters (the optimization is done jointly over both parameters), while emprical localization functions or data-driven techniques leave inflation aside.

We anticipate that GBO may become useful in the future, when localization techniques move to being multi scale or involve a large number of localization scales that may vary throughout the domain. The reason is that, in this case, many parameters are needed to specify the multi scale or spatially varying localization. Even for simple localization schemes with only a single parameter, one may find that GBO brings about significant computational savings over the a grid-search (which is still the standard in particular for finding inflation). The reason is that the GBO refines a model for the objective function as this function is evaluated. Thus, the search for optimal localization/inflation parameters is not 'blind', but takes the function being optimized into account directly (but perhaps less directly than gradient-based optimization methods). Further below, we illustrate computational savings of GBO-based tuning of localization and inflation over a grid-search approach in simple numerical examples.

4.3. Estimating parameters while tuning localization and inflation

If the model parameters are unknown, then the degree of localization/inflation needed depends on what model parameters are used. For example, if the model is poor (bad choice of model parameters), then the required inflation is typically large. On the other hand, even an appropriate choice of the model parameters may lead to large forecast errors if the localization/inflation are not

appropriate. It is thus natural, and in fact advisable, to simultaneously tune localization and inflation, while searching for model parameters. The GBO framework naturally allows for this option by making the parameters one optimizes be the composition of the model parameters and of the parameters that define the localization and inflation.

4.4. Algorithmic details

Recall that the three main steps in GBO are:

i. maximize expected improvement;
ii. optimize the kernel (hyper-parameters)
iii. perform a regression.

Step (iii) is straightforward linear algebra (see Equations (4) and (5)), and steps (i) and (ii) are intertwined by the choice of the Kernel function. In this paper, we focus on the family of Kernels defined in (1). In particular, the reason for adding 'white noise' ($\kappa^2 \cdot \delta(\phi, \phi')$) to the Gaussian Kernel is our implementation of the EnKF. Recall that we use a stochastic implementation of the EnKF, which means that the forecast errors depend (mildly) on the stochastic perturbations used within the EnKF. This in turn implies that the objective function (forecast errors of EnKF) is a random function and the addition of the white noise kernel helps with ameliorating the effects of the stochasticity of the objective function (see also Rasmussen (2004)).

One can, of course, consider more general families of Kernels (see, e.g. Rasmussen (2004)), or one can optimize over the structure and composition of the kernel (Lunderman et al., 2018), but we are content with using a single and relatively simple kernel because this choice seemed sufficient in the numerical examples we tried for this proof-of-concept study. More realistic or real applications, however, may benefit from considering a broader family of kernels that is more capable of emulating more complex functional dependencies.

With the kernel family fixed, one can derive an analytic expression of the expected improvement (see Equation (10)). Step (i), i.e. maximization of EI, is thus relatively straightforward and can be done, e.g. by using a derivative-free BFGS method (Byrd et al., 1995), (or one can compute gradients of EI and use another optimization method). To avoid being trapped in local extrema, we perform a whole suite of BFGS optimizations, each initialized by a different starting value, and then pick the parameter $\boldsymbol{\phi}$ that leads to the highest local maximum. Again, this simple approach was sufficient in all examples we tried, but more sophisticated techniques may lead to better results.

Step (ii), i.e. optimizing the kernel, requires optimization of the negative log marginal likelihood, which also depends on the kernel family. With our choice of kernel family, one can derive the negative log marginal likelihood, $\mathcal{L}(\ell)$, similar to the expression given in (6). Recall that we parametrize the kernel family by a length scale ℓ, and the parameters γ and κ. Throughout we fix $\kappa = 10^{-3}$, because only the ratio of γ and κ is important. We thus optimize the kernel over two parameters (ℓ and γ) and we implement this optimization via a BFGS technique (using the same code as when optimizing EI).

Finally, we use a space-filling, quasi-random Sobol sequences to initialize the GBO (Joe and Kuo, 2008). This means that we specify a (small) number of parameters $\boldsymbol{\phi}$, evaluate the objective function at these points, and then build the initial GP model based on these function evaluations. We use Sobol sequences because these are a common choice, but other technique for finding an initial set of parameters can of course also be used.

In the numerical examples below, we stop the GBO iterations after a specified number of iterations. This means that we a priori decide on a 'computational budget' we wish to invest and accept the (approximate) solution after this number of iterations. It is also possible to stop the iteration using other criteria, e.g. when only little progress is made or when a desired error criterium is met. Such strategies may reveal the same, or a similar, solution to the one we obtain with a specified number of iterations, but after fewer iterations. We are, however, not so concerned about the required computations (because these computations occur during a training phase) and think of a pre-specified computational budget as being a realistic case in many applications.

5. Numerical examples

We present numerical examples to illustrate the use of GBO in ensemble DA. The examples are 'synthetic', in the sense that we use a numerical model to generate noisy observations, which are then used in GBO to estimate model parameters, tune localization and inflation, or both. We use the classical Lorenz models (Lorenz, 1963, 1996) with classical parameters (see below). We simulate each model using a 4^{th} order Runge-Kutta method with time step $\Delta t = 0.01$ for $T = 400$ time units. Observations are collected at regular time intervals $\Delta T = 0.1$:

$$\mathbf{y}_k = \mathbf{x}_k + \boldsymbol{\zeta}_k, \quad \boldsymbol{\zeta}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \text{ iid.} \tag{26}$$

This corresponds to setting $\mathbf{H} = \mathbf{I}$ and $\mathbf{R} = \mathbf{I}$ in our earlier review of ensemble DA. The procedure results in 4000 observations that we use in our GBO algorithms. We stop the GBO after completing 100 iterations, and also report results after the first 10, 25, 50 and 75 iterations to track how the iterations improve the estimates. We then study the GBO results by computing the time-

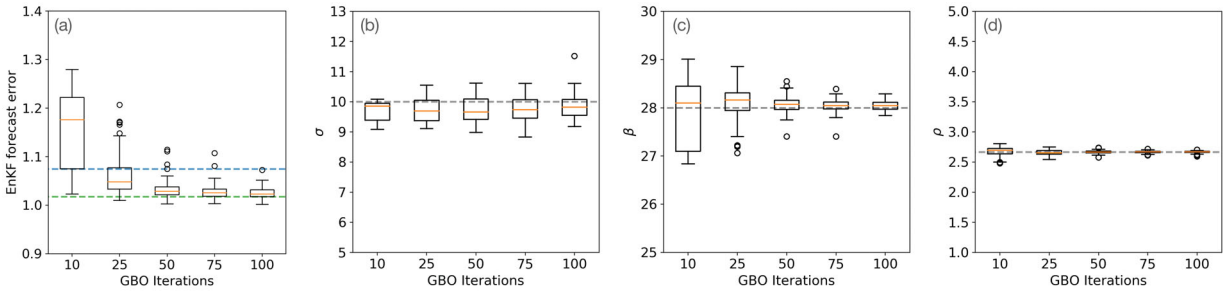

Fig. 3. Summary of results for the parameter estimation of Lorenz '63. (a) Box-and-whisker plot of time averaged forecast RMSE. Dashed blue: RMSE of the augmented state EnKF. Dashed green: RMSE of an EnKF using the 'true' model parameters. Statistical variation of RMSE for the EnKF or augmented state EnKF is too small to be visible on the scale of the plot. (b)–(d) Box-and-whisker plots of the model parameters σ, β and ρ. The limits of the y-axes in (b)–(d) are set to reflect the assumed bounds on the model parameters. In each box-and-whisker plot, the orange line is the median, the box spans the first and third quartiles, the whiskers are the minimum/maximum and dots represent outliers (if applicable).

average of the forecast root mean square error (RMSE), where the forecast RMSE is defined by

$$\mathrm{RMSE}_j = \frac{1}{\sqrt{N_y}} \left|\left| \mathbf{y}_j - \bar{\mathbf{x}}_j^f \right|\right|_2. \tag{27}$$

Here $\bar{\mathbf{x}}_j$ is the EnKF forecast mean (see above) and $N_y = N_x$ is the number of observations, which in all examples is equal to the number of state variables of the model.

The results one obtains, both in terms of the time-averaged forecast RMSE and the estimates of the parameters the GBO generates, depend to some extend on the noise that defines the observations (see Equation (26)). We address this issue by running each numerical experiment 50 times and using different random perturbations of the observations in each experiment. We then report the results of these experiments by showing 'box-and-whisker' plots of the time averaged forecast RMSE. Each plot thus shows the minimum, the maximum, the sample median, the first and third quartiles, and outliers.

5.1. Parameter estimation for Lorenz '63

We consider the Lorenz '63 model (Lorenz, 1963),

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \sigma(y-x), \quad \frac{\mathrm{d}y}{\mathrm{d}t} = x(\rho-z)-y, \quad \frac{\mathrm{d}z}{\mathrm{d}t} = xy-\beta z, \tag{28}$$

where $\sigma = 10$, $\beta = 28$ and $\rho = 8/3$, and demonstrate how to use GBO to estimate these model parameters from observations. Localization and inflation are not tuned by GBO. In fact, we do not use any localization in this example and use a multiplicative inflation of $\alpha = 1.025$, based on preliminary numerical tests.

In the parameter estimation, we assume the following bounds for the model parameters: $\sigma \in [5, 13]$, $\beta \in [25, 30]$ and $\rho \in [1, 5]$. We initialize the GBO algorithm with $N_G = 4$ parameter samples which we obtain a Sobol sequence over the hyper-cube, defined by the parameter bounds. We then perform 100 iterations of GBO. The EnKF is run with an ensemble size $N_e = 25$. Figure 3 summarizes our numerical results. Figure 3(a) shows how forecast RMSE decreases as we increase the number of iterations in the GBO. The reduction in forecast RMSE is reduced due to improved parameter estimates, which are shown in Figure 3(b)–(d). For reference, we also use an augmented state EnKF and an EnKF that uses a model with the 'true' model parameters. We note that the GBO quickly reduces forecast RMSE and finds appropriate parameter estimates which are near the true parameters. After about 50 iterations, the results do not change much, indicating that GBO has converged, and forecast errors are comparable to those of an EnKF that makes use of the 'true' model. After about 25 iterations, the GBO yields forecast errors that are comparable to those of an augmented state EnKF. In summary, we find that the GBO can reliably estimate the parameters of the Lorenz '63 model given a set of observations. This is perhaps not so surprising, since this problem is relatively simple. Nonetheless, the positive results are encouraging and we think that a simple example helps with understanding the algorithm and how it functions in an ensemble DA and parameter estimation context.

5.2. Tuning inflation and localization in Lorenz '96

We consider the Lorenz '96 model (Lorenz, 1996)

$$\frac{\mathrm{d}x_k}{\mathrm{d}t} = -x_{k-1}(x_{k-2}-x_{k+1})-x_k + F, \tag{29}$$

where $k = 1, \ldots, 40$, $F = 8$ is a forcing term, and where we assume a periodic domain so that $x_{-1} = x_{39}, x_0 = x_{40}$ and $x_{41} = x_1$. We assume that the forcing term $F = 8$ is known and wish to tune the localization and inflation of an EnKF (ensemble size $N_e = 40$), applied to the Lorenz '96

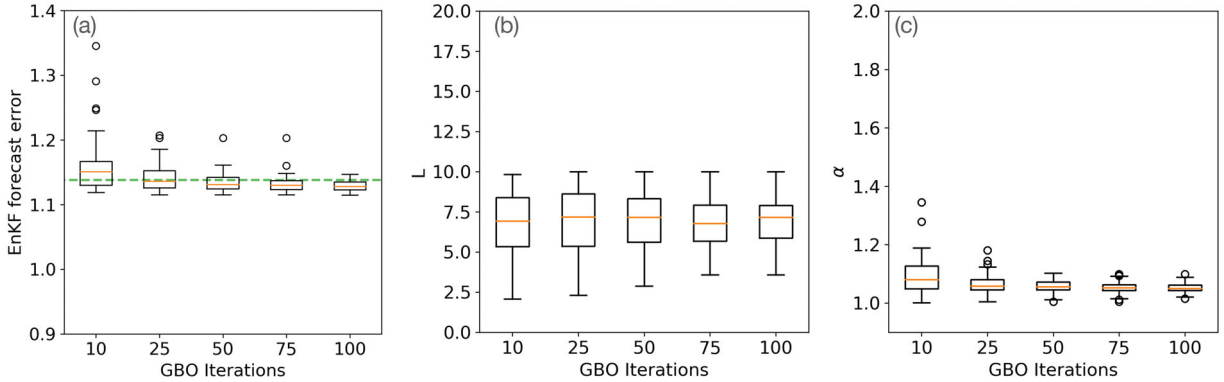

Fig. 4. Summary of results for tuning localization and inflation in Lorenz '96. (a) Box-and-whisker plot of time averaged forecast RMSE. Dashed green: time-averaged forecast RMSE of an EnKF with localization/inflation tuned by a grid-search (statistical variation of RMSE is too small to be visible on the scale of the plot). (b) and (c) show box-and-whisker plots of localization (L) and inflation (α) parameters as a function of GBO iterations. The limits of the y-axes in (b) and (c) are set to reflect the assumed bounds on the localization/inflation parameters. In each box-and-whisker plot, the orange line is the median, the box spans the first and third quartiles, the whiskers are the minimum/maximum and dots represent outliers (if applicable).

model. We constrain our search for localization and inflation using the bounds $L \in (0, 20]$ for the localization length-scale and $\alpha \in [0.9, 2]$ for the multiplicative inflation. The GBO is initialized with $N_G = 2$ parameter samples and, as before, we perform 100 iterations. The results are summarized in Figure 4. We note that, as before, GBO converges quickly and reduces forecast RMSE after only a few steps (see Figure 4(a)). The inflation parameter α is determined to be slightly larger than one, and the variation about the mean across the 50 experiments is small (see Figure 4(c)). The localization parameter (L), shows more variation over the 50 experiment, see Figure 4(b)), which is due to the fact that the cost function is relatively 'flat' (various localization parameters lead to similar forecast errors). This 'flatness' of the cost function may be problem dependent.

We compare the results of GBO with an EnKF where we tune localization and inflation using the grid-search approach. Specifically, we use a uniform 23×20 grid over the square $[0.9, 2] \times (0, 20]$, defined by the bounds for the localization and inflation parameters. Thus, this grid-search requires 460 EnKF runs, but we obtain a similar forecast RMSE (see Figure 4(a)) after 25 GBO iterations, which requires a total of 27 EnKF runs (two for initialization and 25 iterations). Beyond 25 iterations, the localization and inflation tuned by GBO results in smaller forecast errors than what can be obtained by a grid-search. In summary, even for a 'simple' localization and inflation scheme with only two parameters, the GBO approach can lead to significant computational savings when compared to a grid-search technique.

Our study does not address possible computational savings compared to techniques when localization is tuned via a data-driven localization or empirical localization functions. It is clear, however, that the training phase of GBO (or other, similar techniques) is larger than deploying an adaptive/optimal localization or inflation strategy (which require no tuning). Our main goal is to show a proof-of-concept that GBO can be helpful when tuning localization and inflation. More specific ideas, such as choosing localization (inflation) adaptively while using GBO for the inflation (localization) is left for future studies in the context of specific scientific problems. We further expect that computational savings brought about by using GBO may become more important if the tuning of localization and inflation needs to be repeated, e.g. in view of new observations, in view of changes to the resolution of the model, or other significant modifications of the DA system. In fact, the use of GBO may enable a more frequent re-tuning of the localization and inflation, which is currently avoided (in operational centers) due to a large computational cost.

5.3. Parameter estimation while tuning localization and inflation in Lorenz '96

We again consider the Lorenz '96 model, but now estimate the forcing term F, while simultaneously tuning the localization and inflation. Our results are summarized in Figure 5. We note that the GBO iterations reduce forecast RMSE (see Figure 5(a)) and that after about 50 iterations, GBO performs as well as an augmented state EnKF, for which we tuned the localization and inflation

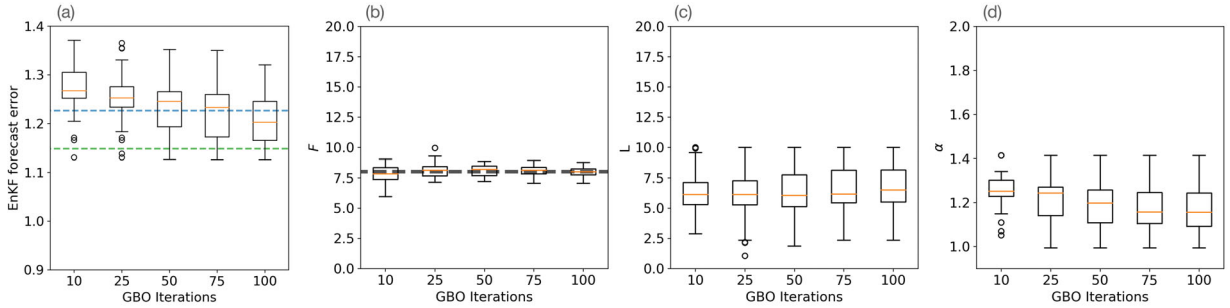

Fig. 5. Summary of results for tuning localization and inflation while estimating a model parameter in Lorenz '96. (a) Box-and-whisker plot of time averaged forecast RMSE. Dashed blue: RMSE of the augmented state EnKF. Dashed green: RMSE of an EnKF using the 'true' model parameters. Statistical variation of RMSE for the EnKF or augmented state EnKF is too small to be visible on the scale of the plot. (b)–(d) show box-and-whisker plots of the forcing F, localization (L) and inflation (α) parameters as a function of GBO iterations. The limits of the y-axes in (b)–(d) are set to reflect the assumed bounds on the model and localization/inflation parameters. In each box-and-whisker plot, the orange line is the median, the box spans the first and third quartiles, the whiskers are the minimum/maximum and dots represent outliers (if applicable).

parameters via a grid-search. We note that the tuning of the augmented state EnKF relied on 460 runs with the augmented state EnKF (we used the same grid for the localization and inflation as in the previous example). The GBO thus leads to a significant reduction in the computational requirements when compared to the augmented state EnKF. We also compare the forecast RMSE of GBO to the forecast RMSE of a tuned EnKF which uses a model with the correct forcing term ($F = 8$), and note that the GBO leads to a forcing term and associated localization/inflation that make the RMSE of the 'uncertain' model comparable to that of the 'true' model. All EnKFs use an ensemble size $N_e = 40$. Figures 5(b)–(d) illustrate how the parameters (model parameter F and localization/inflation parameters L and α) vary with the number of GBO iterations. As before, we note a large variation of the localization parameter over the 50 experiments, which, as before, is due to the fact that the objective function is relatively flat (several localization parameters lead to similar forecast RMSE). We further note that the forcing parameter F is correctly and accurately identified after about 10–25 iterations, so that GBO spends most of the iterations beyond 25 to fine-tune the localization and inflation.

In summary, we could demonstrate on a simple example how GBO can be used for parameter estimation while simultaneously tuning the localization and inflation. The GBO approach leads to significant computational savings when compared to an augmented state EnKF, with its localization/inflation tuned via a grid-search. We note that our numerical experiments are not exhaustive: One can easily come up with schemes where one uses adaptive localization or inflation while estimating model parameters or various other combinations (e.g. using only adaptive localization, tuning inflation and model parameters, etc.). As indicated before, our numerical experiments should be viewed as a proof-of-concept, perhaps useful for assessing the usefulness of GBO in important tasks in ensemble DA.

6. Conclusions

Global Bayesian optimization (GBO) is a derivative-free optimization technique that is used to optimize objective functions that are expensive to evaluate (computationally or otherwise). The basic idea of GBO is to construct and revise an emulator of the objective function and to use this emulator for the function's optimization. GBO is widely used in the tech-industry and in engineering, and, to a lesser extent, in the Earth sciences. The use of GBO for parameter and state estimation problems was pioneered by Abbas et al. (2016) and we continue this line of work by discussing the use of GBO in ensemble data assimilation (DA), where the goal is to estimate a model's state based on noisy observations.

We specifically discussed three related problems in ensemble DA and how these can be tackled by GBO:

i. parameter estimation problems in which some, or all, of the parameters that define the model used in the ensemble DA are uncertain or unknown;
ii. tuning of the ensemble DA (localization and inflation);
iii. estimation of model parameters while simultaneously tuning the ensemble DA.

We believe that problem (ii) is particularly relevant in atmospheric sciences, especially when multi-scale aspects of the ensemble DA are more directly accounted for. We used GBO in all three problems by defining the objective function in terms of forecast errors of the ensemble DA. In this way, we couple the GBO to an ensemble DA

framework indirectly by the definition of the objective function. This formulation makes the computations flexible and one can couple an existing GBO code (which runs on a personal computer or workstation) to an existing ensemble DA code (which may require high-performance computing), which makes GBO easy to use in complex and realistic applications. Moreover, the GBO is agnostic to which ensemble DA method is used and one can easily couple GBO to ensemble Kalman filters (EnKF), or ensemble-variational methods.

The GBO has proved effective for all three tasks in simple numerical experiments with Lorenz models (L63 and L95). We anticipate that using GBO for the tuning of localization and inflation may become important when the community moves to multi-scale techniques (which are needed in view of an ever increasing resolution of the underlying numerical models). For parameter estimation, the flexibility of GBO is advantageous and essentially decouples the linear aspects of the problem (the state estimation) from its nonlinear and non-Gaussian aspects (the parameter estimation). We used simple numerical examples with the classical Lorenz models (Lorenz, 1963, 1996) to illustrate our ideas and to demonstrate the computational savings one can achieve by using GBO. We hope that this work sparks interest in the use of GBO in ensemble DA and that GBO-based methods will be applied to more realistic, or real, problems in the future.

Acknowledgements

We thank Deryl Kleist (NOAA/NWS/NCEP/EMC) for interesting discussion of some of the ideas in this paper. A portion of this research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

SL and DJP were supported by JPL's Strategic University Research Partnerships (SURP) program. MM was supported in part by the US Office of Naval Research (ONR) grant N00014-21-1-2309.

References

Abbas, M., Ilin, A., Solonen, A., Hakkarainen, J., Oja, E. and co-authors. 2016. Empirical evaluation of Bayesian optimization in parametric tuning of chaotic systems. *Int. J. Uncertainty Quantif. 6*, 467–485. doi:10.1615/Int.J.UncertaintyQuantification.2016016645

Anderson, J. 2012. Localization and sampling error correction in ensemble Kalman filter data assimilation. *Mon. Weather Rev. 140*, 2359–2371. doi:10.1175/MWR-D-11-00013.1

Anderson, J. 2016. Reducing correlation sampling error in ensemble Kalman filter data assimilation. *Mon. Weather Review 144*, 913–925. doi:10.1175/MWR-D-15-0052.1

Anderson, J. and Anderson, S. 1999. A Monte Carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts. *Mon. Wea. Rev. 127*, 2741–2758. doi:10.1175/1520-0493(1999)127<2741:AMCIOT>2.0.CO;2

Andrieu, C., Doucet, A. and Holenstein, R. 2010. Particle Markov chain Monte Carlo methods. *J. R. Stat. Soc. Ser. B (Stat. Methodol.) 72*, 269–342. doi:10.1111/j.1467-9868.2009.00736.x

Bocquet, M. and Sakov, P. 2013. Joint state and parameter estimation with an iterative ensemble Kalman smoother. *Nonlin. Processes Geophys. 20*, 803–818. doi:10.5194/npg-20-803-2013

Bonavita, M., Isaksen, L. and Hólm, E. 2012. On the use of EDA background-error variances in the ECMWF 4D-Var. *Q. J. R. Meteorol. Soc. 138*, 1540–1559. doi:10.1002/qj.1899

Buehner, M. 2012. Evaluation of a spatial/spectral covariance localization approach for atmospheric data assimilation. *Mon. Wea. Rev. 140*, 617–636. doi:10.1175/MWR-D-10-05052.1

Buehner, M. and Shlyaeva, A. 2015. Scale-dependent background-error covariance localisation. *Tellus A: Dyn. Meteorol. Oceanogr.* 67, 28027. doi:10.3402/tellusa.v67.28027

Byrd, R. H., Lu, P., Nocedal, J. and Zhu, C. 1995. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput. 16*, 1190–1208. doi:10.1137/0916069

Carrassi, A., Bocquet, M., Bertino, L. and Evensen, G. 2018. Data assimilation in the geosciences: An overview of methods, issues, and perspectives. *Wiley Interdiscip. Rev. Clim. Change 9*, e535. doi:10.1002/wcc.535

Chopin, N., Jacob, P. E. and Papaspiliopoulos, O. 2013. SMC2: An efficient algorithm for sequential analysis of state space models. *J. R. Stat. Soc. Ser. B (Stat. Methodol.) 75*, 397–426. doi:10.1111/j.1467-9868.2012.01046.x

De La Chevrotière, M. and Harlim, J. 2017. A data-driven method for improving the correlation estimation in serial ensemble Kalman filters. *Mon. Wea. Rev. 145*, 985–1001. doi:10.1175/MWR-D-16-0109.1

Destouches, M., Montmerle, T., Michel, Y. and Ménétrier, B. 2020. Estimating optimal localization for sampled background-error covariances of hydrometeor variables. *Q J. R. Meteorol. Soc. 147*(734), 74–93.

Doucet, A., Freitas, N. d., & Gordon, N. (Eds.). 2001. *Sequential Monte Carlo Methods in Practice*. New York, New York: Springer.

El Gharamti, M., Raeder, L., Anderson, J. and Wang, X. 2019. Comparing adaptive prior and posterior inflation for ensemble filters using an atmospheric general circulation model. *Mon. Weather Rev. 147*, 2535–2553. doi:10.1175/MWR-D-18-0389.1

Evensen, G. 2009a. *Data Assimilation: The Ensemble Kalman Filter*. Springer Science & Business Media, Dordrecht, Heidelberg, London, New York.

Evensen, G. 2009b. The ensemble Kalman filter for combined state and parameter estimation. *IEEE Control Syst. 29*, 83–104. doi:10.1109/MCS.2009.932223

Farchi, A. and Bocquet, M. 2018. Review article: Comparison of local particle filters and new implementations. *Nonlinear Processes Geophys. Discuss,* 25(4), 765–807.

Flowerdew, J. 2015. Towards a theory of optimal localisation. *Tellus A: Dyn. Meteorol. Oceanogr. 67*, 25257. doi:10.3402/tellusa.v67.25257

Flowerdew, J. 2017. Initial trials of convective-scale data assimilation with a cheaply tunable ensemble filter. *Q. J. R. Meteorol. Soc. 143*, 1670–1684. doi:10.1002/qj.3038

Frazier, P. I. and Wang, J. 2016. Bayesian optimization for materials design. In: Lookman T., Alexander F., Rajan K. (eds) *Information Science for Materials Discovery and Design*. Springer, vol 225. Springer, Cham.

Hamill, T., Whitaker, J. S., Anderson, J. and Snyder, C. 2009. Comments on "Sigma-point Kalman filter data assimilation methods for strongly nonlinear systems". *J. Atmos. Sci. 66*, 3498–3500. doi:10.1175/2009JAS3245.1

Hamill, T., Whitaker, J. and Snyder, C. 2001. Distance-dependent filtering of background error covariance estimates in an ensemble Kalman filter. *Mon. Wea. Rev. 129*, 2776–2790. doi:10.1175/1520-0493(2001)129<2776:DDFOBE>2.0.CO;2

Houtekamer, P. and Mitchell, H. 2001. A sequential ensemble Kalman filter for atmospheric data assimilation. *Mon. Wea. Rev. 129*, 123–137. doi:10.1175/1520-0493(2001)129<0123:ASEKFF>2.0.CO;2

Hunt, B., Kostelich, E. and Szunyogh, I. 2007. Efficient data assimilation for spatiotemporal chaos: A local ensemble transform Kalman filter. *Phys. D. 230*, 112–126. doi:10.1016/j.physd.2006.11.008

Joe, S. and Kuo, F. Y. 2008. Constructing Sobol sequences with better two-dimensional projections. *SIAM J. Sci. Comput. 30*, 2635–2654. doi:10.1137/070709359

Jones, D. R., Schonlau, M. and Welch, W. J. 1998. Efficient global optimization of expensive black-box functions. *J. Global Optim. 13*, 455–492. doi:10.1023/A:1008306431147

Kotsuki, S., Ota, Y. and Miyoshi, T. 2017. Adaptive covariance relaxation methods for ensemble data assimilation: Experiments in the real atmosphere. *Q. J. R. Meteorol. Soc. 143*, 2001–2015. doi:10.1002/qj.3060

Lei, L., Anderson, J. and Romine, G. 2015. Empirical localization functions for ensemble kalman filter data assimilation in regions with and without precipitation. *Mon. Weather Rev. 143*, 3664–3679. doi:10.1175/MWR-D-14-00415.1

Letham, B., Karrer, B., Ottoni, G. and Bakshy, E. 2019. Constrained Bayesian optimization with noisy experiments. *Bayesian Anal. 14*, 495–519.

Li, H., Kalnay, E. and Miyoshi, T. 2009. Simultaneous estimation of covariance inflation and observation errors within an ensemble Kalman filter. *Q. J. R. Meteorol. Soc. 135*, 523–533. doi:10.1002/qj.371

Lorenc, A. 2003. The potential of the ensemble Kalman filter for NWP – A comparison with 4D-Var. *Q. J. R. Meteorol. Soc. 129*, 3183–3203. doi:10.1256/qj.02.132

Lorenc, A. 2017. Improving ensemble covariances in hybrid variational data assimilation without increasing ensemble size. *Q. J. R. Meteorol. Soc. 143*, 1062–1072. doi:10.1002/qj.2990

Lorenz, E. N. 1963. Deterministic nonperiodic flow. *J. Atmos. Sci. 20*, 130–141. doi:10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2

Lorenz, E. N. 1996. Predictability: A problem partly solved. In: *Proceedings Seminar on Predictability*, Vol. 1. Reading, United Kingdom.

Lunderman, S., Fioroni, G., McCormick, R., Nimlos, M., Rahimi, M. and co-authors. 2018. Screening fuels for autoignition with small-volume experiments and gaussian process classification. *Energy Fuels 32*, 9581–9591. doi:10.1021/acs.energyfuels.8b02112

Luo, X. and Hoteit, I. 2011. Robust ensemble filtering and its relation to covariance inflation in the ensemble Kalman filter. *Mon. Weather Rev. 139*, 3938–3953. doi:10.1175/MWR-D-10-05068.1

Ménétrier, B., Montmerle, T., Michel, Y. and Berre, L. 2015. Linear filtering of sample covariances for ensemble-based data assimilation. Part I: Optimality criteria and application to variance filtering and covariance localization. *Monthly Weather Review 143*, 1622–1643. doi:10.1175/MWR-D-14-00157.1

Moosavi, A., Attia, A. and Sandu, A. 2019. Tuning covariance localization using machine learning. In: *Computational Science – ICCS 2019* (eds. J. M. F. Rodrigues, P. J. S. Cardoso, J. Monteiro, R. Lam, V. V. Krzhizhanovskaya and co-authors.). Springer International Publishing, Cham, pp. 199–212.

Morzfeld, M. and Hodyss, D. 2019. Gaussian approximations in filters and smoothers for data assimilation. *Tellus A: Dyn. Meteorol. Oceanogr. 71*, 1600344. doi:10.1080/16000870.2019.1600344

Nerger, L. 2015. On serial observation processing in localized ensemble Kalman filters. *Mon. Weather Rev. 143*, 1554–1567. doi:10.1175/MWR-D-14-00182.1

Pirot, G., Krityakierne, T., Ginsbourger, D. and Renard, P. 2019. Contaminant source localization via Bayesian global optimization. *Hydrol. Earth Syst. Sci. 23*, 351–369.

Popov, A. and Sandu, A. 2019. A Bayesian approach to multivariate adaptive localization in ensemble-based data assimilation with time-dependent extensions. *Nonlin. Processes Geophys. 26*, 109–122. doi:10.5194/npg-26-109-2019

Posselt, D., van den Heever, S., Stephens, G. L. and Igel, M. 2012. Changes in the interaction between tropical convection, radiation and the large scale circulation in a warming environment. *J. Clim. 35*, 557–571.

Rasmussen, C. E. 2004. Gaussian processes in machine learning. In: Bousquet O., von Luxburg U., Rätsch G. (eds) *Advanced Lectures on Machine Learning*. Springer, Berlin, Heidelberg. p. 193.

Ruckstuhl, Y. and Janjić, T. 2018. Parameter and state estimation with ensemble Kalman filter based algorithms for convective-scale applications. *Q. J. R. Meteorol. Soc. 144*, 826–841. doi:10.1002/qj.3257

Shlyaeva, A., Whitaker, J. and Snyder, C. 2019. Model-space localization in serial ensemble filters. *J. Adv. Model. Earth Syst. 11*, 1627–1636. doi:10.1029/2018MS001514

Snoek, J., Larochelle, H. and Adams, R. P. 2012. Practical Bayesian optimization of machine learning algorithms. In: Bartlett, P. (ed) *Advances in Neural Information Processing Systems*, pp. 2951–2959. Red Hook, NY: Curran Associates Inc.

Talagrand, O. and Courtier, P. 1987. Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory. *Q. J. R. Meteorol. Soc. 113*, 1311–1328. doi:10.1002/qj.49711347812

Vukicevic, T. and Posselt, D. 2008. Analysis of the impact of model nonlinearities in inverse problem solving. *J. Atmos. Sci. 65*, 2803–2823. doi:10.1175/2008JAS2534.1

Williams, C. K. and Rasmussen, C. E. 2006. *Gaussian Processes for Machine Learning*. Vol. 2, No. 3. MIT Press, Cambridge, MA.

Zhen, Y. and Zhang, F. 2014. A probabilistic approach to adaptive covariance localization for serial ensemble square root filters. *Mon. Weather Rev. 142*, 4499–4518. doi:10.1175/MWR-D-13-00390.1