Assimilating Data into Models

Amarjit Budhiraja Eric Friedlander Colin Guider
Christopher KRT Jones John Maclean

October 6, 2017

1 Introduction

Data abound in studies of the environment, and their abundance is increasing at an extraordinary rate. The instruments we use to observe the environment, be it the atmosphere, ocean, land or ice, are constantly improving in their accuracy and efficiency. Moreover, they follow the technological trend of becoming less expensive as time moves on. It is tempting then to think we will eventually enjoy enough observational data that a complete picture of the world around us will be available and constitute a virtual replica from which we can conclude both how the environment works, and how it will change. But more data does not necessarily mean more understanding. It is hard from raw data alone to conclude how different effects are related. Correlation may be detected in data, but to establish causation will usually involve more experimentation than passive data can provide. Needed is the ability to vary conditions and see what results, but these experiments may not be feasible in environmental applications. For instance: imagine wanting to see what would happen if the world's oceans were to increase by $5°C$ in global averaged temperature. Luckily, we have models that allow us to experiment in such ways. These vary from stripped down models that focus on a small number of key effects in a particular environmental application to the few very large, physically inclusive and computationally expensive models, residing at dedicated centers around the world, that amount to computational replicas of the entire Earth system.

Scientific advances owe equally to models and data, and both will remain relevant and key to further understanding. Observations drive model development, and model development often drives data acquisition. It therefore is particularly prudent to have these two sides of the scientific coin work in concert. If we get information from data and from models, then should we not get the most complete picture from directly combining the informational content from each? This is a mathematical and statistical question: how to combine the output of model investigations and observational data. The area that is dedicated to studying and developing the best approaches to this issue is called *Data Assimilation (DA).* The image intentionally conveyed by the term is that a model is running and producing output, while the data are brought in to update and/or amend the output as it progresses, thus the data are being assimilated into the model.

The main impetus for the development of DA came in the 1980s from our need to predict the weather. At that time, models were coming into use in preparing daily weather forecasts [18]. From the seminal work of Lorenz two decades earlier [17], it was understood that an inherent sensitivity resided within the weather models that would lead to an inevitable discrepancy between model forecast and actual weather. The idea then arose (see [20]) that observational data could be used to correct the system state and keep the forecast on track. Thus was born the area of *Numerical Weather Prediction (NWP)* which underpins the great success of modern-day weather forecasts, whether they be of tomorrow's precipitation or the track of a category 5 hurricane.

Data assimilation methods are then aimed at giving a mathematical framework for suitably updating the system state predicted by a model through the use of observations of the state collected from sensory instruments. The main thrust of decades of research has been to develop mathematically justified and computationally tractable methods for DA that make the best use of the model and observations. A key point is that both models and data contain errors, and confidence will reside in each to varying degrees. The key characteristic of a DA scheme will then be how it resolves the issue of balancing the anticipated model error versus that residing in the observational data.

1.1 The core of a DA scheme

The heart of any DA scheme lies in the step at which data are incorporated into our (mathematical) description of the system state. This description will come from applying some model to the situation under study. In the NWP example, it might be the prediction of the weather based on estimates from 6 hours previously. We take this description to be a vector $x \in \mathbb{R}^M$, which could be thought of as physical variables, such as temperature, pressure and wind speed, listed at all of the grid points of our numerical scheme. The dimension M may be large and this presents an issue that we will discuss below. But, for understanding the logic of the DA step, the size of M is not relevant.

We start with our current "guess" of the system state, this is the current (mathematical) description of the system state. We denote this x^b where the superscript b stands for *background.* The observations are denoted y^o and are also assumed to be a vector but of possibly quite different dimension. We shall set $y^o \in \mathbb{R}^m$. In general then, $m \neq M$ and often, particularly in geophysical applications, $m \ll M$, which reflects the fact that the number of observations is often far less than the number of variables involved in describing the state of the system. Relating the observations to the system is a function $h : \mathbb{R}^M \to \mathbb{R}^m$, called the observation operator.

The core step is then to produce a new system state from the knowledge of x^b and y^o, which is called the analysis state and denoted by x^a. So the DA scheme amounts to a way in which we make the following transition concrete:

$$\left\{x^b, y^o\right\} \to x^a. \tag{1}$$

This chapter gives an overview of several different ways to realize this move from input information, coming from both model and data, to the best-informed estimate of the system state.

1.2 Model and observations

There is potential for confusion when the term model is used in a DA context. We usually think of a model as a mathematical representation of a physical situation. Such a model might be given by a differential equation or statistical process. But in DA, the term model is used in a related but technically different way. When a differential equation for a spatially dependent system is solved, the underlying physical space is covered by a grid and the physical variables are evaluated at the nodes of that grid. The computation then is aimed at calculating these physical variables at the grid points based on the underlying physical laws, which are manifest in the differential equations. The computation then "solves" those equations numerically to evaluate the system state. But in this context, the system state is thought of as the set of physical variable values at the grid points. The "model" is then thought of as the computational process that updates this state vector, and no longer the original mathematical model.

The observation operator is a critical element of the DA process and much can hide behind it. First, imagine that observations are of physical variables at grid points. In this case, the observation operator would just be a projection onto the components of the state vector that are being measured, and we could consider the observation space $\mathbb{R}^m$ to be a subspace of the state space $\mathbb{R}^M$. But, of course,

it is in general not so simple. The grid points are those determined by the numerical scheme being used in the model and there is no reason why observational measurements should be made at those points. We can, secondly, imagine the observations being of physical variables at points other than grid points. The corresponding "observational" values at observation locations can then be concluded by (linear) interpolation from model values nearby the observation location in question. In both of these cases, the observation operator would be linear, but there are more complicated relationships between observations and physical state variables that demand a more complex observation operator, which may be nonlinear and involve a model in its own right.

Data assimilation methods usually do not attempt to 'invert' the observation operator to map a data point in the observation space to a point in the state space. In other words, we work with the observations in their own space and not the full state space. This is because there are typically many less observations than there are model variables, and so much less error is introduced by converting model variables (e.g. temperature) at model grid points into observed variables (e.g. satellite radiances) at observation locations, than vice versa. See for instance [12] for an elaboration of this point.

1.3 Challenges of DA

Numerical Weather Prediction is used to make weather forecasts, typically ranging from 6 hours to 10 days after the initial state. The model for NWP is based on the primitive equations: conservation laws applied to the atmosphere, to which some approximations or simplifications must be made. This results in a system of seven partial differential equations, pertaining to velocity, density, pressure, temperature, and humidity.

As described above, these equations are not solved exactly but instead numerically approximated at discrete points in space. The computational model is obtained by discretizing (or "gridding up") the spatial domain in km or degrees in the longitude x and latitude y, and in km or pressure for the height z. For example, NOAA's Global Forecasting System uses a resolution of 0.5° horizontally and 20 layers vertically from the surface of the earth. Thus, there are $720 \times 360 \times 20 = 5,184,000$ grid points. Between seven and dozens of variables must be specified at each of these grid points to initialize or compute with this model, so that the numerical model has a dimension of $\mathcal{O}(10^8 - 10^9)$, which is the size M of the state space in the previous paragraphs.

Observations (data) are available from weather balloons, planes, satellites, etc. These observations are heavily concentrated over land (particularly America and Eurasia). Observations are usually not made at model grid points, and often are not of model variables but instead of related quantities. Such a situation leads to the type of complex observation operator mentioned in the previous subsection. A good example is that of satellite observations (these are often of radiances) where physical variables, such as temperature, have to be inferred from the direct observations. This inference is then encoded in the observation operator.

Although these observations may be plentiful and the resulting dimension m of the observation space may be high, they are likely to be many fewer than the number of dimensions in the state space, M. It may appear that m will catch up with M over time as we get more and more observations, but it is just as certain that M will increase accordingly as it largely depends on the current state of computing power.

1.4 Approaches to DA

There are three main approaches to DA, i.e., to realizing the analysis step (1). Each stems from a different historical tradition and we will explain them in that way here. But it should be noted, and will be seen throughout this chapter, that they are each very strongly related to each other. DA is

often presented with one of the approaches as the primary focus depending on the background of the authors. Our goal in this overview is to describe all of these different approaches and hopefully convey the strengths of the various viewpoints.

1.4.1 Variational approach

This is based on the formulation of a cost function that suitably penalizes deviations of a candidate state vector, x, from the model forecast and from the observations. This method comes from Optimization theory. We start with a background state x^b which is a first guess at the state of the system and observations y^o. For each of these a covariance matrix expresses the confidence we have in them: $\mathbf{B}$ for the background and $\mathbf{R}$ for the observational data. Each piece of information is then weighted with the inverses of their respective covariance matrices and the cost function is the sum of two quadratic terms.

$$J(x) = \frac{1}{2}\left[(x - x^b)^T \mathbf{B}^{-1}(x - x^b) + (y^o - h(x))^T \mathbf{R}^{-1}(y^o - h(x))\right] . \tag{2}$$

The analysis then corresponds to a state vector x^a that minimizes this cost function. If the observation operator is linear, this is the standard least-squares problem for which we have a unique minimizer. For nonlinear h a standard strategy is to linearize the observation operator in some appropriate fashion, which approximates the problem of minimizing (2) by a least-squares problem and provides a unique approximate minimizer. Note however that the cost function given by (2) in general may have multiple minima.

1.4.2 Kalman gain approach

This approach to DA produces the analysis x^a from a linear combination of the background and the observations. This can be expressed as

$$x^a = x^b + \mathbf{W}\left(y^o - h\left(x^b\right)\right), \tag{3}$$

where $\mathbf{W} \in \mathbb{R}^{M\times m}$ is a suitable weight matrix, called the *gain matrix*. It is calculated from the same covariance matrices $\mathbf{B}$ and $\mathbf{R}$ that appear in the cost function of the variational approach. The ubiquitous form of the gain matrix in DA is the *Kalman gain*,

$$\mathbf{W} = \mathbf{B}\mathbf{H}^T(\mathbf{R} + \mathbf{H}\mathbf{B}\mathbf{H}^T)^{-1} , \tag{4}$$

where $\mathbf{H}$ is a suitable linearization of h. This gain matrix was originally derived as part of the *Kalman-Bucy Filter* or *Kalman Filter*, which we will define later, and originated in control theory [11, 10]. However the Kalman gain appears in multiple other DA schemes, and we derive it here assuming that x^b and y^o are independent of each other and distributed normally with covariances given respectively by $\mathbf{B}$ and $\mathbf{R}$. In this simple context, as a DA scheme this is known as *Optimal Interpolation* (OI).

We posit a "true value of the system state" which we denote x^t (t should not be confused with time here!). The mean of the distributions of x^b and y^o are then taken to be x^t and $h(x^t)$ respectively. The OI scheme then requires that the Mean-Squared-Error (MSE) $E\left[(x^a - x^t)^T(x^a - x^t)\right]$ is minimized. Letting $\epsilon^a = x^a - x^t$ we have

$$\begin{aligned}\epsilon^a = x^a - x^t &= x^b + \mathbf{W}\left(y^o - h\left(x^b\right)\right) - x^t \\ &= (x^b - x^t) + \mathbf{W}y^o - \mathbf{W}h(x^b).\end{aligned}$$

A key step is to linearize h around x^b which yields, writing $\epsilon^b = x^b - x^t$ and $\epsilon^o = y^o - h(x^t)$,

$$\begin{aligned}\epsilon^a &\approx (x^b - x^t) + \mathbf{W}y^o - \mathbf{W}[h(x^t) + \mathbf{H}(x^b - x^t)] \\ &= \epsilon^b + \mathbf{W}\epsilon^o - \mathbf{W}\mathbf{H}\epsilon^b = \mathbf{W}\epsilon^o + (I - \mathbf{W}\,\mathbf{H})\epsilon^b, \end{aligned} \tag{5}$$

where $\mathbf{H}$ is now defined explicitly to be the linearization of h at x^b. It can be shown using elementary matrix calculus that the Kalman gain (4) minimizes the MSE $E\left[(\epsilon^a)^T\epsilon^a\right]$, if it is expressed using (5).

To connect with the variational approach, we will see that if h is replaced by the linearization $\mathbf{H}$ in both (2) and (3) then the minimizer of the cost function $J(x)$ in (2) is given by x^a from (3) with $\mathbf{W}$ given by the Kalman gain (4).

1.4.3 Probabilistic approach

The key difference in the probabilistic or Bayesian approach to DA is that instead of updating point estimates of the state or $x^b \mapsto x^a$, it is full probability distributions that are instead updated. Each probability density function (pdf) is written as $p()$, and must be interpreted by looking at the variables it is conditioned upon. For example, $p(x)$ represents the likelihood function of the state with no knowledge of observations: this is usually called the *prior* or *forecast* distribution. The *posterior* or *analysis* distribution is found by conditioning the forecast distribution on the observation, forming $p(x|y^o)$, and the question of data assimilation is how to find, or approximate, this posterior density. By an application of Bayes' formula, one can write the posterior density as

$$p(x|y^o) \propto p(y^o|x) \times p(x), \tag{6}$$

where $p(y^o|x)$ is the conditional density, or *likelihood*, of the observation given that the state is x. In principle, this reduces the task of finding the posterior density into two tasks: find or approximate the prior or forecast density $p(x)$, and the likelihood $p(y^o|x)$.

The probabilistic approach can be related to both of the above approaches. First, it is directly related to the variational method as follows: Under a Gaussian assumption on the prior $p(x)$,

$$p(x) \propto \exp\left\{-(x - x^b)^T\mathbf{B}^{-1}(x - x^b)\right\},$$

and with the likelihood of the observations given by

$$p(y^o|x) \propto \exp\left\{-(y^o - h(x))^T\mathbf{R}^{-1}(y^o - h(x))\right\}\ ,$$

it is easy to write the Bayesian posterior in terms of the variational cost function (2), with

$$p(x|y^o) \propto \exp\left\{-J(x)\right\}. \tag{7}$$

As in the variational section more can be said if the observation operator is linear, so that $h(x) = \mathbf{H}x$. In this case the likelihood $p(y^o|x)$ is Gaussian, and so the posterior $p(x|y^o)$ is Gaussian as well. The global minimizer of the cost function $J(x)$ is then the mode (and mean) of $p(x|y^o)$. The connection to the Kalman gain approach, as before, is that the mean of $p(x|y^o)$ is given by (3) with $\mathbf{W}$ from (4). Note that (7) can be made into a general relationship between variational and Bayesian approaches for arbitrary prior and likelihood densities by writing the variational cost function $J(x)$ as a combination of the logarithms of these densities.

1.5 Perspectives on DA

The different ways of looking at a data assimilation problem are evident from the previous section. It is worth pointing out that some of the difference comes from the fundamentally different perspectives of statisticians and applied mathematicians. The applied mathematician will tend to look for a single answer to the question: "what is the best estimate of the system state?" The underlying view here is that there is some "truth" about the system out there and the task is to approximate that as closely as possible. This perspective is reflected in the formulation described above in Section 1.4.2. The same view underlies the variational method. Mathematically, this view encounters problems when the observation operator is nonlinear and the cost function (2) is not necessarily quadratic. The cost function may then have multiple local minima and even the global minimum may not be unique (although this is a non-generic situation.) Efforts then to find a minimum can take the person implementing the DA scheme down blind alleys. Nevertheless, this view is powerful and compelling and has driven much of the historical development of DA.

The statistician will take a different view by seeking to determine how likely the system is to be in a certain state. The viewpoint here is that even if there is a "true state of the system," we cannot know it and can only assign probabilities to prospective states. The idea of DA, from this perspective, is then to combine all the information, coming from model runs and observations, to formulate the "best" probability density function for the state of the system. This is clearly encoded in Bayes' Formula (6). This shifts the problem from estimating a particular state to computing a probability distribution.

It might be expected that one or other of these viewpoints would have won out, but both remain important and influential. The statistician's approach, encoded in Bayes' Formula, is now taken as providing the framework for DA [16, 19] but the variational and Kalman gain methods remain dominant as operational problem solvers. The reason for this equivocation lies in the nature of the problems that DA is asked to address. As discussed above, the geophysical applications, such as weather and climate prediction, that have driven DA development have very high dimensional state spaces. The methods that have been developed for approximating the pdfs involved in Bayes' Formula, such as particle filters, work very well in low dimensions but do not scale to high dimensions. In particular, they do not require any assumptions of Gaussianity, nor any need to linearize the observation operator. This makes them very appealing, especially since they approximate the full pdf and not just the mean (or mode), but if they cannot deal with high dimensions, then other methods will still be needed.

To summarize this point, there is an inherent tension in current DA research between nonlinearity and dimension. The statistical methods do very well in dealing with nonlinearity in low dimensions, but some kind of linearization and the methods of Kalman Filtering and/or the variational approach are needed to deal with high dimensional problems.

1.6 Chapter Overview

In the following sections we elaborate on fundamental DA schemes from the variational, Kalman gain and Bayesian approaches. The notation used will reflect the approach taken, as described in the sections above, but may also change to reflect the dependence of the DA problem on time.

Section 2 is devoted to the variational approach introduced in 1.4.1. The notation of 'background' x^b is used throughout this section to refer to the model variables before the assimilation step. We introduced this notation in 1.4.1 and its use reflects a historical use in NWP. The 'background' was frequently a guess for the state produced from historical data, e.g. a climatological mean.
We describe the 3D-Var scheme in Section 2.1, that is concerned with estimating the state given observations at the same fixed time.

In Section 2.2 we present a second variational method, 4D-Var, that assimilates observations from multiple time instants simultaneously in order to estimate the state at an initial time, and consequently the notation for observations will change to include time subscripts. We comment that the problem of estimating the state given observations from a later time is known as *smoothing*, and that 4D-Var is the only smoothing method that we consider.

Section 3 is concerned with the Bayesian approach described in 1.4.3. This involves a switch in notation as we transfer from the applied mathematics perspective to the statistical perspective. In prior sections capitals typically denote matrices, or the large model dimension M. In this section, capitals will denote random variables. Moreover, all the methods described in this section will be *sequential*. This describes a particular recursive dependence of the DA problem on time, in which we would like to alternate between using the model to propagate the state forward in time, and using the observations at that time to improve or correct the forecast. Consequently the notation for both model and observations will include a time index.
In Section 3.2 we derive the celebrated Kalman Filter from the Bayesian approach. The Kalman Filter is a method of the form (3)–(4) in which we also track the confidence in the analysis. As described at the end of Section 1.4.3, there is a close connection between the deterministic and the Bayesian approaches, and in this section we present, in the context of the Kalman filter, a translation between the notation and perspectives of the Bayesian and the Kalman gain approach. In the same section we discuss extensions of the Kalman Filter suitable for (weakly) nonlinear models or observation operators, and show a connection between the Kalman Filter and 4D-Var schemes.
In Section 3.3 we describe the particle filter, another DA method that arises from the Bayesian formulation. This method does not need, nor use, linearity in the model or observations, nor Gaussianity in the prior or posterior pdf.

Section 4 is concerned with modifications of the above methods that are made in practice to mitigate the weak points of each method. This is an extensive topic and the focus of much research; we do not attempt a comprehensive review of the literature but instead present some of the key techniques.

2 Variational Methods

The two fundamental variational methods are usually referred to as 3D-Var and 4D-Var. As described above, 3D-Var is a method that finds a minimizer of the cost function (2), while 4D-Var is concerned with an extension of the 3D-Var problem in which the observations are taken at different time points and consequently minimizes a different, but related, cost function.

2.1 3D-Var

Consider the cost function (2), which seeks to balance the observation increment $y^o - h(x)$ with the deviation of the analysis from the background $x - x^b$, where the terms are weighted by the inverse of their respective covariance matrices, and which we rewrite here for convenience:

$$J(x) = \frac{1}{2}\left[(x - x^b)^T \mathbf{B}^{-1}(x - x^b) + (y^o - h(x))^T \mathbf{R}^{-1}(y^o - h(x))\right] .$$

Thus the 'more uncertain' term is given 'lesser weight' in this cost. By linearizing h around x^b, and defining $d = y^o - h(x)$, one obtains

$$J(x) \approx \frac{1}{2}\left[(x-x^b)^T\mathbf{B}^{-1}(x-x^b) + (y^o - h(x^b) - \mathbf{H}_{x^b}(x-x^b))^T\mathbf{R}^{-1}(y^o - h(x^b) - \mathbf{H}_{x^b}(x-x^b))\right]$$
$$= \frac{1}{2}\left[(x-x^b)^T\mathbf{B}^{-1}(x-x^b) + (d - \mathbf{H}_{x^b}(x-x^b))^T\mathbf{R}^{-1}(d - \mathbf{H}_{x^b}(x-x^b))\right]. \quad (8)$$

The analysis x^a is defined to be the minimizer of the above approximation of the cost function, which, using elementary vector calculus, yields

$$x^a = x^b + \mathbf{W}d\,, \quad (9)$$
$$\mathbf{W} = \left(\mathbf{B}^{-1} + \mathbf{H}_{x^b}^T\mathbf{R}^{-1}\mathbf{H}_{x^b}\right)^{-1}\mathbf{H}_{x^b}^T\mathbf{R}^{-1}. \quad (10)$$

It is easy to check that $\mathbf{W}$ defined by (10) is same as the gain matrix defined in (4), so that the minimizer of (2) is the same as the analysis given by (3). In applications in NWP, however, the matrix inversion of $\mathbf{B}^{-1} + \mathbf{H}_{x^b}^T\mathbf{R}^{-1}\mathbf{H}_{x^b}$ required to calculate $\mathbf{W}$ in either (4) or (10) is often too expensive to compute. This prevents the direct implementation of Kalman gain approaches to DA in NWP. The advantage of 3D-Var is that the variational description (2) allows one to avoid the exact calculation of (9) by instead finding an approximation by minimization algorithms. Moreover such algorithms usually result in a more accurate estimates of x^a since they will not just use the linearization of the h function around the background x^b, but recursively linearise h around successive guesses for the analysis. Recall that in order to compute $\mathbf{W}$ in the OI scheme h was linearized around x^b, however in 3D-Var one can improve the point about which linearization takes place using an iterative method. This is discussed in the following section.

2.1.1 Incremental Method for 3D-Var

The 3D-Var analysis can be approximated without requiring the evaluation of (10). The algorithm consists, first, of an outer loop that creates an approximation of the cost function with the observation operator linearized around a guess for the analysis, and an inner loop in which the cost function (8) is minimized. Below is a step-by-step description of the algorithm.

1. Set $x^{(1)} := x^b$ and $i, = 1$.
2. (Outer Loop) Define i-th approximation of the gradient
$$\nabla J(x) \approx \nabla J_i(x) = \mathbf{B}^{-1}(x-x^b) - \mathbf{H}_{x^{(i)}}^T\mathbf{R}^{-1}(d - \mathbf{H}_{x^{(i)}}(x-x^b))$$
by linearizing h around $x^{(i)}$, the current best guess for x^a.
3. If the gradient is "close enough" to zero then accept $x^{(i)}$ as x^a. Namely, if $|\nabla J_i(x^{(i)})| < \epsilon$ where ϵ is some predefined threshold then stop and take the final analysis as $x^a = x^{(i)}$. Otherwise:
    4. (Inner Loop) Set $\eta^{(0)} = x^{(i)}$ and call a minimization subroutine of the form $\eta^{(j+1)} = \eta^{(j)} + \alpha_k f(\nabla J_i(\eta^{(j)}))$. Methods for choosing α_k and f are described elsewhere but a simple example is choosing a fixed $\alpha_k := \alpha$ and $f(x) = -x$.
    5. When a minimum, $\eta^{(J)}$, has been found, set $x^{(i+1)} = \eta^{(J)}$, increment i by 1 and repeat steps 2-5.

In addition, there exist a variety of computation techniques (PSAS, preconditioning, etc.) that the user can use to reduce the number of iterations needed for the algorithm to converge. The improved accuracy of the algorithm stems from the ability to relinearize h in step 4. The algorithm in general will not converge without additional conditions on h, however experimental results have shown that the incremental method can achieve much more accurate estimates than OI without a significant increase in computational complexity.

2.2 4D-Var

Implicit in the previous variational methods has been that all components of the observation vector y^o were collected at the same time instant. In this section we give an overview of the 4D-Var scheme, that assimilates observations at multiple time instants by minimizing a global cost function.

Suppose we want to assimilate observations collected at times $\{t_n\}_{n=1}^N$ over some time interval. Denote these observations by $\{y_n\}_{n=1}^N$. The 4D-Var method minimizes a cost function analogous to (8), but that uses a numerical integrator to compare state estimates to observations at multiple time steps. We distinguish between two formulations of 4D-Var, strong and weak constraint.

2.2.1 Strong constraint 4D-Var

We now assume the numerical model to be perfect. This implies that given that the state at time t_0 is x, one can obtain a perfect state value $m_n(x)$ at every time instant t_n through a nonlinear model integrator m_n. The goal of the DA scheme is to find the initial condition that best matches the background x^b and all the observations. The background is viewed as some imprecise 'first guess' of the state at time instant t_0. The model in this formulation is a strong constraint because the cost function does not allow predicted state values to deviate from those obtained from a forward integration of the model.

The initial condition uniquely determines the model trajectory and thus all the analysis values. Once again we denote by $\mathbf{B}$ the background covariance matrix, which captures the uncertainty associated with the forecast at time t_0 and denote by $\mathbf{R}_n$ the observation covariance matrix at time t_n which quantifies the uncertainty associated with the observation at time t_n. The cost function is then given as follows

$$J(x) = \frac{1}{2}\left[(x-x^b)^T\mathbf{B}^{-1}(x-x^b) - \sum_{n=1}^{N}\left(y_n - h\left(m_n(x)\right)\right)^T\mathbf{R}_n^{-1}\left(y_n - h\left(m_n(x)\right)\right)\right]. \tag{11}$$

Intuitively, by minimizing the above cost function, we are selecting the x at time t_0 that is not too far from the background and also not too far from the observations when x is "pushed forward" to the corresponding time and transformed through the observation function h.

Minimization of J is similar to the method used for 3D-Var but more approximations need to be made to deal with the nonlinearily of the forecast operator m_n. Most of the computational effort is in suitably approximating the gradient of J. In order to accomplish this we separate the components of J and write $J(x) = J^b(x) + J^o(x)$ where

$$J^b(x) = \frac{1}{2}\left[(x-x^b)^T\mathbf{B}^{-1}(x-x^b)\right]$$

and

$$J^o(x) = \frac{1}{2}\left[\sum_{n=1}^{N}(y_n - h(x_n))^T\mathbf{R}_n^{-1}(y_n - h(x_n))\right]. \tag{12}$$

Note that $\nabla J = \nabla J^b + \nabla J^o$. Clearly

$$\nabla J^b(x) = \mathbf{B}^{-1}(x-x^b).$$

Suppose starting from the forecast x^b at time t_0 the state $x_n^b = m_n(x^b)$ at time instant n is computed. The state update function m_n can be written as $m_{n-1,n} \circ m_{n-2,n-1} \cdots \circ m_{0,1}$ where $m_{j-1,j}$ is the

function that propagates the state at time instant $j-1$ to j. We denote by $\mathbf{M}[t_{n-1}, t_n]$ the linearization of $m_{n-1,n}$ around x^b_{n-1}. Using these matrices the gradient ∇J^o can be approximated as

$$\nabla J^o(x) \approx \sum_{n=1}^{N} \mathbf{M}^T[t_1, t_0] \cdots \mathbf{M}^T[t_n, t_{n-1}] \mathbf{H}_n^T d_n$$

where $d_n = \mathbf{R}_n^{-1} y_n - h(m_n(x))$ and $\mathbf{H}_n$ is the gradient of h at $x^b_n = m_n(x^b)$. Therefore, every iteration of the minimization algorithm requires two steps. First, the model must be integrated forward using a candidate analysis state x in order to calculate d_n. Then a backward pass must be made to calculate the gradient. This requires evaluating $m_{n,n+1}(x^b_n)$ for all n.

2.2.2 Incremental Method for 4D-Var

As with 3D-Var, one may find it impossible to calculate the minimizer of $J(x)$; we present here an approach that approximates the analysis, and does not use x^b_n as the point of linearization but rather improves the linearization points recursively through a gradient descent algorithm. Specifically, the algorithm works as follows.

1. (Outer loop) Choose an starting point $x_0^{(1)}$ (normally x^b is chosen) and set $i = 1$.
2. Integrate the model forward to calculate

   $$x_n^{(i)} = m_{n-1,n}(x_{n-1}^{(i)})$$

   These values are used as points about which linearization is done to calculate $\mathbf{M}[t_{n+1}, t_n]$ and $\mathbf{H}_n$.
3. Using the $x_n^{(i)}$'s obtained from the forward integration calculate the weighted observation increments $d_n = \mathbf{R}_n^{-1}(h(x_n^{(i)}) - y_n)$.
4. Using these d_n's calculate the gradient

   $$\nabla J_i\left(\{x_0^{(i)}, \ldots, x_n^{(i)}\}\right) = \mathbf{B}^{-1}(x - x^b) + \sum_{n=1}^{N} \mathbf{M}^T[t_1, t_0] \cdots \mathbf{M}^T[t_n, t_{n-1}] \mathbf{H}_n^T d_n$$

5. If $\left|\nabla J_i\left(\{x_0^{(i)}, \ldots, x_n^{(i)}\}\right)\right| < \epsilon$, where ϵ is a predetermined threshold, then accept each $x_n^{(i)}$ as x_n^a. Otherwise:
   6. (Inner loop) Set $\eta^{(0)} = x_0^{(i)}$ and call a minimization subroutine of the form $\eta^{(j+1)} = \eta^{(j)} + \alpha_k \nabla J(\eta^{(j)})$.
   7. When a minimum, $\eta^{(J)}$, has been found, set $x_0^{(i+1)} = \eta^{(J)}$, increment i by 1 and repeat steps 2-7.

2.2.3 Weak Constraint 4D-Var

The perfect model assumption can be relaxed and this is known as weak constraint 4D-Var. Instead of selecting an optimal initial condition, the algorithm produces a set of state values $\{x_0, x_1, \ldots, x_N\}$ that approximate the state at each time from t_0 to t_N. The cost function (8) gains an extra term that measures the difference between the state estimate x_n and the push-forward of the model at

the previous time step, $m_{n-1,n}(x_{n-1})$. Consequently the state estimates x_n are only approximately consistent with the model. The cost function is now written as

$$J(x_0,\dots,x_N) = \frac{1}{2}\Bigg((x_0 - x^b)^T \mathbf{B}^{-1}(x_0 - x^b) + \sum_{n=1}^{N} \left(x_n - m_{n-1,n}(x_{n-1})\right)^T \mathbf{Q}^{-1} \left(x_n - m_{n-1,n}(x_{n-1})\right) + \sum_{n=1}^{N} (y_n - h(x_n))^T \mathbf{R}_n^{-1} (y_n - h(x_n)) \Bigg),$$

where $\mathbf{Q}$ is the model error covariance.

The above cost function also has a statistical interpretation. Suppose that $X_0 \sim \mathcal{N}(x^b, \mathbf{B})$ and

$$X_{n+1} = m_{n-1,n}(X_n) + \epsilon_n^b,\ Y_n = h(X_n) + \epsilon_n^o,$$

where $\epsilon_n^b \sim \mathcal{N}(0, \mathbf{Q})$, $\epsilon_n^o \sim \mathcal{N}(0, \mathbf{R}_n)$ are mutually independent (and independent of X_0) Normal random variables. Then $J(x_0,\dots,x_N)$ equals, up to a constant, the negative log of the conditional density of $(X_0,\dots X_N)$(evaluated at $(x_0,\dots,x_N)$) given $(Y_1,\dots Y_N) = (y_1,\dots y_N)$. Thus minimizing the cost function corresponds to maximizing this log-likelihood.

3 Bayesian formulation and sequential methods

We now reformulate the problem of Data Assimilation in statistical terms, changing the notation we have previously established in order to match that used by statisticians. We adopt the Bayesian approach, in which Bayes' law (6) transitions prior beliefs about the model variables into a posterior density via the likelihood function. The analogue of the prior density was previously the background state or first guess; one way to convert these deterministic quantities into a prior is to model the prior as Gaussian, with mean given by the background and variance given by the level of confidence in the background, that is $\mathbf{B}$ in (2), (4). We model the true state sequence and the observation sequence as a collection of random variables $\{(X_n, Y_n)\}_{n=0}^N$ that describe a two component (state, observation) Markov chain. Here X_n represents the true state of the system at time instant t_n and Y_n the observation collected at the same time instant. We will use lower case x_n, y_n to refer to realizations of these random variables. Note that an observation that has been made will always be a realization of the random variable, and so will always be written in lower case.
The underlying state process $\{X_n\}_{n=0}^N$ is unobservable, so all inference must be based on $\{Y_n\}_{n=0}^N$.

The use of time-dependent state and observation variables suggests one should employ a sequential Data Assimilation algorithm to recursively estimate the state given ever-updating observations. We now describe how this can be done using a Bayesian approach.

Let us describe the complete sets of state variables and observations from t_0 to t_n by $X_{0:n} = \{X_0, X_1, \dots, X_n\}$, $Y_{1:n} = \{Y_1, \dots, Y_n\}$, and similarly define the realizations of those random variables by $x_{0:n}$, and $y_{1:n}$. Then Bayes' rule says that

$$p(x_{0:n}|y_{1:n}) \propto p(y_{1:n}|x_{0:n}) p(x_{0:n}) \,. \tag{13}$$

Analogously to Section 1.4.3, $p(x_{0:n}|y_{1:n})$ is the posterior density, $p(y_{1:n}|x_{0:n})$ is the likelihood of observing $y_{1:n}$, and $p(x_{0:n})$ is the prior or forecast density.

Working with (13) would require updating the joint distribution on $X_{0:n}$ in order to assimilate the observation y_n, for each n. One would also have to calculate the joint likelihood of every observation, $Y_{1:n}$. Rather than working with these joint densities, we now formulate the *filtering* problem, in which

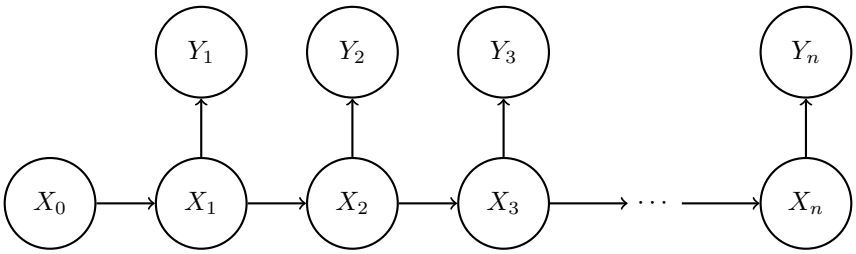

Figure 1: Dependency Structure for Nonlinear State Space Model

assumptions on the structure of the model and observations allow us to rewrite (13) so that we can recursively update the pdf for the state X_n at time t_n, instead of updating the joint density $X_{0:n}$. We then consider the Kalman Filter and Particle Filter, so-called because they solve the filtering problem.

3.1 Filtering

We assume that the distribution of Y_n given $\{Y_k\}_{k=0}^{n-1}$ and X_n depends only on X_n, and the distribution of X_n given X_{n-1} and $\{Y_k\}_{k=0}^{n-1}$ depends only on X_{n-1}. This dependency structure is shown in Figure 1. Given these assumptions, (13) can be rewritten as

$$p(x_n|y_{1:n}) = p(y_n|x_n)p(x_n|y_{1:n-1}), \tag{14}$$

where the forecast density is $p(x_n|y_{1:n-1})$ and the posterior density is $p(x_n|y_{1:n})$. The question of computing the posterior density $p(x_n|y_{1:n})$ is the filtering problem, and methods that accomplish this are called filtering methods or filters. To compute the forecast density $p(x_n|y_{1:n-1})$, one must integrate the posterior density $p(x_{n-1}|y_{1:n-1})$ from the previous time step, as follows:

$$p(x_n|y_{1:n-1}) = \int p(x_n|x_{n-1})p(x_{n-1}|y_{1:n-1})dx_{n-1}. \tag{15}$$

This is the probabilistic analogue of a forecast model in the case of noiseless state dynamics. The posterior density is then given by $p(x_n|y_{1:n}) = p(y_n|x_n)\, p(x_n|y_{1:n-1})$. Note that point estimates of the analysis x_n^a can be given by the mean (or other measures of centrality such as the median or the mode) of the posterior density $p(x_n|y_{1:n})$.

In general closed form expressions for the forecast and posterior distributions are not readily available for typical physical applications due to nonlinearity of state space models and observation functions. However there is one important setting where closed form expressions can be given, namely that of the classical *Kalman filter*.

3.2 The Kalman Filter

Suppose that the state-observation Markov process has the dependence structure in Section 3.1 and is described through the following linear model with Gaussian errors

$$X_n \sim \mathcal{N}\left(\mathbf{M}_n\, x_{n-1},\, \mathbf{Q}_n\right)\ , \tag{16}$$

where X_n, $x_n \in \mathbb{R}^M$ and $\mathbf{M}_n$, $\mathbf{Q}_n \in \mathbb{R}^{M\times M}$, and suppose that observations are sampled independently and with Gaussian errors from the linear observation operator $\mathbf{H}_n$, with

$$Y_n \sim \mathcal{N}\left(\mathbf{H}_n\, x_n,\, \mathbf{R}_n\right)\ , \tag{17}$$

where $Y_n \in \mathbb{R}^m$, $\mathbf{H}_n \in \mathbb{R}^{m\times M}$, and $\mathbf{R}_n \in \mathbb{R}^{m\times m}$. Namely, the conditional distribution of X_n given $(X_{0:n-1}, Y_{0:n-1})$ is Normal with conditional mean $\mathbf{M}_n X_{n-1}$ and conditional variance $\mathbf{Q}_n$ and conditional distribution of Y_n given $(X_{0:n}, Y_{0:n-1})$ is Normal with conditional mean $\mathbf{H}_n X_n$ and conditional variance $\mathbf{R}_n$. We assume without loss of generality that $X_0 \sim N(0, \mathbf{B})$.

It is easy to check that the linear, Gaussian form of the model and observations imply that the prior, likelihood and posterior pdf in (14) are all Gaussian. In this case, each distribution can be specified by giving its mean and covariance. We define the mean of the prior density $p(x_n|y_{1:n-1})$ to be $x_{n|n-1}$, and its covariance to be $\mathbf{P}_{n|n-1}$; similarly for the posterior density $p(x_n|y_{1:n})$, we define the mean to be $x_{n|n}$ and the covariance to be $\mathbf{P}_{n|n}$. This notation refers back to the Bayesian formulation of the filter; the first subscript gives the present time step, and the second subscript gives the time step of the last observation that we condition on. Using this notation in the model (16), we see that the forecast step in the analysis cycle consists of updating the mean and covariance

$$x_{n|n-1} = \mathbf{M}_n \, x_{n-1|n-1} \ , \tag{18}$$

$$\mathbf{P}_{n|n-1} = \mathbf{M}_n \, \mathbf{P}_{n-1|n-1} \, \mathbf{M}_n^T + \mathbf{Q}_n \ . \tag{19}$$

Before we construct the mean and covariance of the posterior density, let us pause to establish the connection between the variables used above and the Kalman gain approach of Section 1.4.3.
The mean of the prior density can be interpreted as the background x^b. In fact the only difference is that the notation used in this section allows for multiple assimilation steps, taken sequentially, while the notation of the background x^b in Section 1.4.2 assumes we are assimilating once at a fixed time. This has a standard name in the Kalman gain approach, one uses the term *forecast* for the sequential prediction of the model state. One can define $x_n^f := x_{n|n-1}$ and $\mathbf{B}_n := \mathbf{P}_{n|n-1}$ as time-varying analogues of the background x^b and confidence in the background, $\mathbf{B}$, respectively. Observe that $\mathbf{B}$ needed to be known a priori for OI and 3D-Var, while the Kalman Filter calculates the (optimal) value of $\mathbf{B}$ at each time step.

Returning to the derivation, one can use (16)–(19) and Bayes' rule (6) to determine that the posterior density is also Gaussian, with mean and covariance given by

$$x_{n|n} = \mathbf{P}_{n|n} \left(\mathbf{H}_n^T \, \mathbf{R}_t^{-1} \, y_n + \mathbf{P}_{n|n-1}^{-1} \, x_{n|n-1} \right) \ ,$$

$$\mathbf{P}_{n|n} = \left(\mathbf{H}_n^T \, \mathbf{R}_t^{-1} \mathbf{H}_n + \mathbf{P}_{n|n-1}^{-1} \right)^{-1} \ .$$

These can be rewritten as

$$x_{n|n} = x_{n|n-1} + \mathbf{K}_n \left(y_n - \mathbf{H}_n x_{n|n-1} \right) \ , \tag{20}$$

$$\mathbf{P}_{n|n} = \left(I - \mathbf{K}_n \, \mathbf{H}_n \right) \mathbf{P}_{n|n-1} \ , \tag{21}$$

where the matrix $\mathbf{K}_n$ is given as

$$\mathbf{K}_n = \mathbf{P}_{n|n-1} \, \mathbf{H}_n^T \left(\mathbf{H}_n^T \, \mathbf{P}_{n|n-1} \, \mathbf{H}_n + \mathbf{R}_t \right)^{-1} \ . \tag{22}$$

Equations (18) – (22) fully describe the analysis cycle for the Kalman filter at time instant t.

We now complete the connection between the Bayesian formulation of the Kalman Filter presented here and the Kalman gain approach of Section 1.4.2. If we change notation as described above, replacing $x_{n|n-1}$ with the forecast x_n^f and $\mathbf{P}_{n|n-1}$ with the confidence $\mathbf{B}_n$, then (20) is clearly (3) (but with a linearized observation operator) and (22) is clearly the Kalman gain (4). Apart from the sequential nature of the forecast and of the confidence in the forecast, the only difference is that one now also

obtains a confidence level in the analysis, $\mathbf{P}_{n|n}$. It is easy to check (see for instance [22]) that for the linear model (16) and data collection (17) the Kalman Filter provides the minimum MSE estimate of the true system state, by calculating the optimal confidence levels in the forecast and analysis at each time step.

3.2.1 Extensions

The simplicity and tractability of the Kalman filter has led to its use even for settings where the model conditions for its validity are not satisfied. One common setting is where the linear functions $\mathbf{M}_n x$ and $\mathbf{H}_n x$ are replaced by general nonlinear functions $m_n(x)$ and $h_n(x)$. In such a setting an adaptation of the Kalman filter, usually referred to as the *extended Kalman filter* , approximates the densities in the analysis cycle (14)–(15) by Gaussian approximations obtained by replacing $\mathbf{M}_n$ and $\mathbf{H}_n$ in (18) – (22) with the linearizations of m_n and h_n about appropriate points. We refer the reader to [9] for details.

Another approach that attempts to bypass the linearization of m_n is the so-called *ensemble Kalman filter*. Here one uses the full nonlinear state equation

$$x_n = m_n(x_{n-1}) + \tau_n \tag{23}$$

to simulate state values that are used to approximate the forecast density, and the nonlinear observation function h_n. More precisely, having obtained a Gaussian approximation for the posterior distribution at step $n-1$, one takes L samples from this distribution labeled as $\{X^i_{n-1}\}_{i=1}^L$. Using this and L samples $\{\tau^i_n\}$ of the noise in the state model, one uses the nonlinear state equation (23) to produce X^i_n according to

$$X^i_n = m_n(X^i_{n-1}) + \tau^i_n \,, \tag{24}$$

where $i = \{1, \cdots, L\}$. The samples approximate the forecast distribution at time n as a Gaussian density p^f_n with mean

$$x_{n|n-1} = \frac{1}{L}\sum_{i=1}^{L} X^i_n \,, \tag{25}$$

and covariance given by the covariance of the ensemble, with the (i, j)-th entry given by

$$\left(\mathbf{P}_{n|n-1}\right)_{ij} = \frac{1}{L-1}\left(X^i_n - x_{n|n-1}\right)\left(X^j_n - x_{n|n-1}\right)^T \,. \tag{26}$$

This forecast distribution is used to produce a Gaussian posterior density by linearizing the observation function h_n and either using a modified form of equations (20)–(21) in which the observations are perturbed slightly for each ensemble member [4], or using the class of Ensemble Square Root Filters. We refer the reader to [7] for details.

3.2.2 Equivalence of 4D-Var and KF

Recall that strong constraint 4D-Var, formulated in Section 2.2.1, assumes that the model is perfect and attempts to find the optimal initial condition $x^a(t_0)$ which is the best match to both the background x^b and a set of observations $\{y_0, y_1, \ldots, y_N\}$ at times t_0, t_1, …, t_N. This initial condition then uniquely determines the analysis states $x^a(t_n)$, obtained by integrating $x^a(t_0)$ with the model.
In this Section we argue that the analysis state $x^a(t_N)$ obtained from 4D-Var at the final observation time exactly matches the mean of the final analysis state $x_{N|N}$ obtained from a Kalman filter with the

same initial condition and observations, provided the model is perfect and linear and the observations are linear.

Recall that the Kalman Filter provides a recursive solution to the *filtering* problem, i.e. it computes $p\left(x(t_n)|y_0,\ldots,y_n\right)$, the conditional distribution of the state at the current time given all observations up to the current time. The analysis state $x^a(t_0)$ in 4D-Var computes the mode of $p\left(x(t_0)|y_0,\ldots,y_N\right)$, namely the conditional distribution of the state at the initial time given all future observations (and this is one example of a *smoothing* problem, where the state is conditioned on observations at a later time). We will denote the *pushforward* of this density under the n time step map M_n as $M_n * p\left(x(t_0)\right) = p\left(x(t_n)\right)$ (cf. [15], Ch.1).

The equivalence of 4D-Var and the KF under the restrictive assumptions above can now be seen as follows. In the case of a perfect, deterministic model, the push-forward of the smoothing density, i.e. $M_N * p(x_0|y_0,\ldots,y_N)$, is the filtering density at the final observation time, $p(x(t_N)|y_0,\ldots,y_N)$. This is due to the fact that state dynamics is noiseless (see for example [15], Section 2.5). As can be seen from Section 3.2, in the case of a perfect linear model with a linear observation operator, the final analysis step $x^a(t_N)$ of the Kalman filter is the mode of the filtering distribution (note that for a Gaussian posterior, the mean and mode are identical). On the other hand, in the case of a perfect linear model with linear observation operator, the analysis $x^a(t_0)$ of 4D-Var is the mode of the smoothing distribution at the time t_0. From this and the linearity of M_n it follows that $x^a(t_N)$ is the mode of $p(x(t_N)|y_0,\ldots,y_N)$, proving the claimed equivalence. To be clear, this statement does not apply to the iterative approach to 4D-Var we described in Section 2.2, because that method approximates the analysis x^a. Instead, here $x^a(t_0)$ refers to the unique minimizer of (11), which by definition is the mode of the smoothing distribution.

3.3 Particle Filters

The methods discussed in all previous sections rely at some point on linearity of the observation operator and/or of the model, or have used a linearization approximation. In this section we give a brief overview of a collection of particle based schemes that are quite flexible, do not rely on linearization of the dynamics and are well suited for parallel computing architecture. These methods have a long history; we refer the reader to [6] for a comprehensive review of this area. The basic idea is to replace the high dimensional integrals in (14)-(15) with suitable Monte-Carlo sample averages. Instead of using probability densities to describe the distributions, here we will use discrete probability measures supported on finitely many points. These points and their weights will evolve in time to give the forecast measure Π_n^f and posterior measure Π_n for different values of t. One can compute the analysis state x_n^a by computing an integral with respect to the measure Π_n.

3.3.1 A basic particle filter

Suppose that Π_{n-1} is given as a discrete probability measure supported on points $x_{n-1}^1,\ldots x_{n-1}^L$ and with corresponding weights $p_{n-1}^1,\ldots p_{n-1}^L$. Here L represents the number of particles that are used to approximate the distribution Π_{n-1}. The two key steps in the analysis cycle are as follows:
Prediction step. Propagate each of the particles $x_{n-1}^i \mapsto \hat{x}_n^i$ using the nonlinear state dynamics (23). This requires simulating L noise random variables τ_n^i, $i = 1,\ldots L$. Given such random variables, $\hat{x}_n^i$ are defined as

$$\hat{x}_n^i = m_n(x_{n-1}^i) + \tau_n^i.$$

This gives the forecast probability distribution Π_n^f as a discrete probability measure concentrated on L points $\{\hat{x}_n^i\}_{i=1}^L$ with weights $\{p_{n-1}^i\}_{i=1}^L$.

Filtering step. Update the weights $\{p^i_{n-1}\}_{i=1}^L$ using the observation Y_n by setting $p^i_n = cp^i_{n-1}R(\hat{x}^i_t, Y_n)$, where

$$R(x,y) \doteq \exp\left\{-\frac{1}{2}(Y_n - h(\hat{x}^i_n))^T \mathbf{R}_t^{-1}(Y_n - h(\hat{x}^i_n))\right\}. \tag{27}$$

The posterior distribution Π_n is then defined as the discrete measure with support points $\{x^i_n = \hat{x}^i_n\}_{i=1}^L$ and weights $\{p^i_n\}$.

Although this scheme is easy to implement, it suffers from severe degeneracy, especially in high dimensions. The main difficulty is that after a few time steps all the weights tend to concentrate on a very few particles which drastically reduces the effective sample size. A common remedy for this paucity of significant particles is to occasionally re-sample in order to refresh the particle cloud.

3.3.2 Particle filter with resampling

The main idea here is to periodically resample with replacement from the discrete distribution Π_n to obtain a uniform distribution of weights. Of course, resampling adds extra noise to the approximating scheme so it is important not to resample too frequently. Fix a resampling lag parameter $\alpha \in \mathbb{N}$. This parameter specifies the number of time steps between successive resampling steps. Suppose that Π_{n-1} is given as a discrete probability measure supported on points $x^1_{n-1}, \ldots x^L_{n-1}$ and corresponding weights $p^1_{n-1}, \ldots p^L_{n-1}$. Suppose first that n/α is not an integer. In this case the posterior distribution Π_n is given exactly as before in Section 3.3.1. If n/α is an integer we further modify the above discrete probability measure Π_n as follows. Take a random sample of size L from the discrete distribution $\{(x^1_n, p^1_n), \ldots, (x^L_n, p^L_n)\}$. Relabel the new points as $(x^1_n, \ldots, x^L_n)$. The posterior distribution Π_n is then given as the discrete distribution: $\{(x^1_n, p^1_n), \ldots, (x^L_n, p^L_n)\}$ where all the p^i_n are set equal to $1/L$.

3.3.3 Variance reduction: Deterministic allocation and residual resampling

As noted earlier, one drawback of the above algorithm is that it unnecessarily introduces extra variability in the algorithm due to random sampling. This limitation motivates the study of various variance reduction schemes. We describe below one such commonly used scheme. Another variance reduction scheme is given in the next subsection. Let α be as in the last subsection. The key difference here is that at a resampling step, i.e. when n/α is an integer, instead of random sampling with replacement from $(x^1_n, p^1_n), \ldots, (x^L_n, p^L_n)$ we do a partial deterministic allocation as follows. Let $k_i = \lfloor Lp^i_n \rfloor$ and branch (i.e. duplicate) x^i_n into k_i particles. This yields $\sum_{i=1}^L k_i$ particles. Set $L_r \doteq L - \sum_{i=1}^L k_i$ and $w^i_n = Lp^i_n - k_i$. Now resample L_r particles using random sampling from the distribution $\{(x^1_n, cw^1_n), \ldots, (x^L_n, cw^L_n)\}$, where c is a normalizing constant. This gives a total of L particles which are relabeled as $x^1_n, \ldots, x^L_n$. Finally, as before, the posterior distribution Π_n is given as the discrete distribution: $\{(x^1_n, p^1_n), \ldots, (x^L_n, p^L_n)\}$ where all the p^i_n are set equal to $1/L$.

3.3.4 Branching particle filter

In this scheme(cf. [5]) the number of particles is allowed to change at each time step. Let α be as before. The main steps are as follows. Suppose that Π_{n-1} is given as a discrete measure on L_{n-1} points x^i_{n-1}, $i = 1, \ldots, L_{n-1}$ with equal weights (namely $1/L_{n-1}$). Propagate each of the particles $x^i_{n-1} \mapsto \hat{x}^i_n$ using the nonlinear state dynamics (23) as in Section 3.3.1. The main difference from Sections 3.3.1 – 3.3.3 is that instead of reweighing the particles as in the above algorithms we "branch and kill particles". More precisely, if n/α is not an integer, the particle $\hat{x}^i_n$ branches into a random number (denoted by ζ^i_n) of particles. The distribution of ζ^i_n is given as

$$P(\zeta^i_n = \lfloor \gamma^i_n \rfloor + 1) = \gamma^i_n - \lfloor \gamma^i_n \rfloor = 1 - P(\zeta^i_n = \lfloor \gamma^i_n \rfloor),$$

where

$$\gamma_n^i = \frac{L_{n-1}\beta_n^i}{\sum_{i=1}^{L_{n-1}} \beta_n^i},\ \beta_n^i = R(\hat{x}_n^i, Y_n).$$

This results in a total of say L_n particles denoted as $\{x_n^i\}_{i=1}^{L_n}$. The posterior distribution Π_n is the discrete distribution: $\{(x_n^1, p_n^1), \ldots, (x_n^{L_n}, p_n^{L_n})\}$ where all the p_n^i are set equal to $1/L_n$. In order to manage the explosion or decay of the number of particles L_n we add a resampling step to restore the number of particles to L at every α time steps, which is carried out as in Section 3.3.2 or 3.3.3.

3.3.5 Regularized particle filters

One common difficulty with particle filters is that they suffer from the lack of diversity among particles. This problem can be particularly severe in settings where the noise in state dynamics is degenerate. In order to treat this difficulty one usually considers regularized particle filters(cf. [6, Chapter 12]) which corresponds to replacing the sampling from the discrete distribution $\{(x_n^1, p_n^1), \ldots, (x_n^L, p_n^L)\}$ by that from an absolutely continuous approximation. The key idea is to use kernel density smoothers. The two basic versions of such regularized particle filters correspond to regularization at the prediction step and regularization at the filtering step. We only describe the latter, details of the first scheme can be found in [6, Chapter 12].

A regularization kernel K is a symmetric probability function on $\mathbb{R}^k$ satisfying $\int_{\mathbb{R}^k} xK(x)dx = 0$ and $\int_{\mathbb{R}^k} \|x\|^2 K(x)dx < \infty$. For a smoothing parameter $\gamma \in (0, \infty)$, referred to as the bandwidth, denote $K_\gamma(x) \doteq \frac{1}{\gamma^K} K(x/\gamma)$. The two commonly used kernels are the Gaussian kernel and the Epanechnikov kernel (cf. [6]). The basic algorithm is as follows. We consider the regularization of the algorithm in Section 3.3.2. For simplicity we take the lag parameter $\alpha = 1$. Suppose that Π_{n-1} is given as a discrete measure on L points x_{n-1}^i, $i = 1, \ldots, L$ with equal weights $1/L$. Define $\Pi_{n-1}^f = \{(\hat{x}_n^1, 1/L), \ldots, (\hat{x}_n^L, 1/L)\}$ as in Section 3.3.1. Update the weights $\hat{p}_n^i = 1/L \mapsto p_n^i$ using the observation Y_n by setting $p_n^i \doteq c\hat{p}_n^i R(\hat{x}_n^i, Y_n)$, where c is the normalization constant. Draw $\{\tilde{x}_n^i, i = 1, ..., L\}$ from the discrete distribution $\{(x_n^1, p_n^1), \ldots, (x_n^L, p_n^L)\}$ and generate $\{\epsilon^i\}_{i=1}^L$, i.i.d. from the kernel K. Define (abusing notation) $x_n^i \doteq \tilde{x}_n^i + \gamma\epsilon^i$. The posterior distribution Π_n is then the discrete distribution: $\{(x_n^1, p_n^1), \ldots, (x_n^L, p_n^L)\}$ where all the p_n^i are set equal to $1/L$. Such a random jiggling of the particle locations can be particularly important if the stochastic dynamical system governing the state evolution is very sensitive to the initial condition.

4 Implementation of DA methods

This section is concerned with the following practical question: given a particular DA problem, consisting of a numerical model, which may be low or high dimensional, linear or weakly or strongly nonlinear, and data, how does one go about selecting a DA scheme, what are the common pitfalls of each DA scheme, and how are these pitfalls typically mitigated?

A particular DA problem may be suitable for several DA schemes, or only one, or indeed none at all, and the suitability of a particular scheme may not be obvious from the approach it is formulated under. For instance Optimal Interpolation, which we developed in Section 1.4.2 as a simple implementation of the Kalman gain approach, was developed as a method for DA in Numerical Weather Prediction (NWP), and was only later associated with the Kalman gain approach, which originated in control theory. One might ask then why the Kalman Filter, that is the optimal scheme formulated under the Kalman gain approach, was never used in NWP, given that the linearization of the model and observation operator needed for the Kalman Filter are used in NWP. In fact, the Kalman Filter has never been used for NWP, and NWP centers adopted the variational scheme 3D-Var and some later moved to using 4D-Var.

This historical quirk in the development of DA schemes is due to the scaling of the computational cost of each scheme with the dimension of the model. As outlined in Section 1.3, numerical models for NWP may have a dimension of order 10^9. We mention in Section 2.1 that the direct computation of a Kalman gain type matrix is too expensive for a model of this high dimension, and this rules out for the time being the use of the Kalman Filter in NWP; the variational schemes 3D-Var and 4D-Var employ the incremental approximations of Sections 2.1.1 and 2.2.2 to avoid computing the gain matrix. The ensemble Kalman filter is, similarly, significantly computationally cheaper in the analysis step than directly evaluating the Kalman gain, though at the cost of requiring an ensemble of runs of the numerical model.
There is another large gap in performance between the Kalman gain approach and the Particle Filter. In particular, without exploiting a special structure in the model or advanced filtering schemes, use of the Particle Filter will require the model dimension to be at most 10–20. This restriction is also due to the computational cost of the Particle Filter; one finds that the number of particles required in the Particle Filter scales exponentially with the effective dimension of the model, and this exorbitant scaling in computational cost quickly becomes unreasonable.

The above paragraph may seem to imply that there is no advantage to the Kalman or Particle Filter. In practice, though, this is not the case. The (non)linearity of the model is another major factor in the selection of a DA scheme. A crucial point is that, while the schemes orginally developed under the variational or Kalman gain approach can be reformulated under the Bayesian framework, it is not possible to reformulate the particle filter under the variational or Kalman gain approach. For example, the Kalman Filter requires that the model and observations be linear with Gaussian errors, and so the posterior pdf is always Gaussian; the ensemble Kalman Filter does not require that the model be linear, but attempts to fit the best Gaussian posterior pdf to the forecast ensemble. By contrast, the particle filter does not require that the model be linear and does not require that the posterior pdf be Gaussian. In consequence, the ability of 3D-Var, 4D-Var, the ensemble Kalman Filter and Particle Filter to handle nonlinearity (or nonGaussianity) is roughly inverse to their computational cost.

One therefore must decide to include or eliminate DA schemes from consideration by comparing the model dimension, the nonlinearity of the model, and what is known or expected about the structure of the posterior pdf.

As a closing remark for this section, we note that the capabilities and in particular the drawbacks of each DA scheme are very much a topic for active research. For instance, recent developments in the ensemble Kalman Filter are competitive with 4D-Var in some situations, and the question of the relative drawbacks of each method is quite complicated; see [13, 23] for a review and numerical implementation. Meanwhile operational NWP centers have spearheaded the development of hybrid variational schemes. These schemes blend the fixed background covariance used in standard 3D-Var or 4D-Var with an evolving covariance matrix, estimated from an ensemble in a manner like the ensemble Kalman Filter, in a manner designed to yield the benefits of both schemes [2, 3, 14].

4.1 Common modifications for DA schemes

It is common in the Data Assimilation literature to employ some strategies in the application of a DA scheme to mitigate the known biases or flaws in that scheme. These strategies do not change the scheme to such an extent that the scheme is known under a different name, but are typically mentioned in concert with the scheme.

4.1.1 Localization

A problem known as 'spurious correlations' occurs when state variables are updated by any DA scheme using distant observations. For instance, the local weather in North Carolina should not affect the weather in California. However, it is sometimes the case that nonzero entries in the forecast covariance matrix will create a (spurious) correlation between distant sites, due to a coincidental pattern in the forecasts or to noise in the numerical model. The strategy employed to mitigate this problem is to *localize* the background or forecast covariance matrix (respectively $\mathbf{B}$ and $\mathbf{P}_{n|n-1}$), which ensures observations can only affect state variables within some selected distance. This can be effected by for instance taking the Schur product of the background or forecast covariance matrix with a matrix that is 0 on every entry that links distant locations, and 1 everywhere else, or by using some other cut-off function. For instance, if there is one state variable measured at locations $\{1, 2, 3\}$ and we want to localize so that only adjacent locations are updated by observations, then we would rewrite the forecast covariance matrix as

$$\mathbf{P}_{n|n-1} \mapsto \mathbf{P}_{n|n-1} \circ \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} ,$$

where $\circ$ denotes element-wise multiplication.

4.1.2 Inflation

This section is concerned only with ensemble methods, particularly the ensemble Kalman Filter. One would like for the data assimilation problem to correctly account for the uncertainty or errors in the forecast, and this is done via the forecast covariance matrix. However, the forecast covariance matrix (26) is rank deficient, as the number L of ensemble members is typically much less than the dimension M of the numerical model, and consequently some directions in model space in which the forecast is uncertain may not be represented in the forecast covariance matrix. Furthermore, the forecast covariance matrix usually will not take into account the presence of model error, which is a feature in any application of DA. For these reasons, the ensemble Kalman Filter will typically under-estimate the forecast covariance matrix. The strategy adopted to remedy this defect is to artificially increase the forecast covariance. There are broadly two ways this may be done: multiplicative covariance inflation [1], which involves choosing a number $\delta > 0$ and replacing the forecast covariance matrix $\mathbf{P}_{n|n-1}$ according to

$$\mathbf{P}_{n|n-1} \mapsto (1 + \delta)\mathbf{P}_{n|n-1} ,$$

and additive inflation [8], in which one adds a small amount of variance along the diagonal, replacing

$$\mathbf{P}_{n|n-1} \mapsto \mathbf{P}_{n|n-1} + \delta \mathbf{I} .$$

Both methods have advantages. Multiplicative inflation preserves the rank and range of the forecast covariance matrix; that is, relationships between variables do not change. Additive inflation adds a little significance to each variable in the forecast, and so will prevent the covariance matrix from collapsing over a few dominant modes, but changes the relationships between variables. These inflation schemes can be tuned to relax the ensemble in a way that matches the prior, or the observations; see for instance [21]. Either or both methods can be combined with localization.

References

[1] Jeffrey L Anderson and Stephen L Anderson. A monte carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts. *Monthly Weather Review*, 127(12):2741–2758, 1999.

[2] Mark Buehner, PL Houtekamer, Cecilien Charette, Herschel L Mitchell, and Bin He. Intercomparison of variational data assimilation and the ensemble kalman filter for global deterministic nwp. part i: Description and single-observation experiments. *Monthly Weather Review*, 138(5):1550–1566, 2010.

[3] Mark Buehner, PL Houtekamer, Cecilien Charette, Herschel L Mitchell, and Bin He. Intercomparison of variational data assimilation and the ensemble kalman filter for global deterministic nwp. part ii: One-month experiments with real observations. *Monthly Weather Review*, 138(5):1567–1586, 2010.

[4] Gerrit Burgers, Peter Jan van Leeuwen, and Geir Evensen. Analysis scheme in the ensemble kalman filter. *Monthly weather review*, 126(6):1719–1724, 1998.

[5] Dan Crisan, Pierre Del Moral, and Terry Lyons. *Discrete filtering using branching and interacting particle systems*. Citeseer, 1998.

[6] Arnaud Doucet, Nando De Freitas, and Neil Gordon. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pages 3–14. Springer, 2001.

[7] Geir Evensen. *Data assimilation: the ensemble Kalman filter*. Springer Science & Business Media, 2009.

[8] Thomas M Hamill and Jeffrey S Whitaker. Accounting for the error due to unresolved scales in ensemble data assimilation: A comparison of different approaches. *Monthly weather review*, 133(11):3132–3147, 2005.

[9] Andrew H Jazwinski. *Stochastic processes and filtering theory*. Courier Corporation, 2007.

[10] Rudolph E Kalman and Richard S Bucy. New results in linear filtering and prediction theory. *Journal of basic engineering*, 83(1):95–108, 1961.

[11] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[12] Eugenia Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge university press, 2003.

[13] Eugenia Kalnay, Hong Li, Takemasa Miyoshi, SHU-CHIH YANG, and JOAQUIM BALLABRERA-POY. 4-d-var or ensemble kalman filter? *Tellus A*, 59(5):758–773, 2007.

[14] David D Kuhl, Thomas E Rosmond, Craig H Bishop, Justin McLay, and Nancy L Baker. Comparison of hybrid ensemble/4dvar and 4dvar within the navdas-ar data assimilation framework. *Monthly Weather Review*, 141(8):2740–2758, 2013.

[15] Kody Law, Andrew Stuart, and Zygalakis Konstantinos. *Data Assimilation: A Mathematical Introduction*, volume 62. Springer Texts in Applied Mathematics, 2015.

[16] Kody Law, Andrew Stuart, and Konstantinos Zygalakis. *Data assimilation: a mathematical introduction*, volume 62. Springer, 2015.

[17] Edward N Lorenz. Deterministic nonperiodic flow. *Journal of the atmospheric sciences*, 20(2):130–141, 1963.

[18] Peter Lynch. *The emergence of numerical weather prediction: Richardson's dream*. Cambridge University Press, 2006.

[19] Sebastian Reich and Colin Cotter. *Probabilistic forecasting and Bayesian data assimilation*. Cambridge University Press, 2015.

[20] Ian Roulstone and John Norbury. *Invisible in the Storm: the role of mathematics in understanding weather*. Princeton University Press, 2013.

[21] Jeffrey S Whitaker and Thomas M Hamill. Evaluating methods to account for system errors in ensemble data assimilation. *Monthly Weather Review*, 140(9):3078–3089, 2012.

[22] Christopher K Wikle and L Mark Berliner. A bayesian tutorial for data assimilation. *Physica D: Nonlinear Phenomena*, 230(1):1–16, 2007.

[23] Shu-Chih Yang, Matteo Corazza, Alberto Carrassi, Eugenia Kalnay, and Takemasa Miyoshi. Comparison of local ensemble transform kalman filter, 3dvar, and 4dvar in a quasigeostrophic model. *Monthly Weather Review*, 137(2):693–709, 2009.