ELEMENTARY 4D-VAR

DARC Technical Report No. 2

Ross N. Bannister

Data Assimilation Research Centre, University of Reading, UK

Last revised: 16th October 2001 / 29th August 2007

CONTENTS

1. WHAT IS "4D-VAR"?

All forecast models, whether they represent the state of the weather, the spread of a disease, or levels of economic activity, contain unknown parameters. These parameters may be the model's initial conditions, its boundary conditions, or other tunable parameters which have to be found for a realistic result. Four dimensional variational data assimilation, or "4d-Var", is a method of estimating this set of parameters by optimizing the fit between the solution of the model and a set of observations which the model is meant to predict. In this context, the procedure of adjusting the parameters until the model 'best predicts' the observables, is known as *optimization*. The "four dimensional" nature of 4d-Var reflects the fact that the observation set spans not only three dimensional space, but also a time domain.

For weather forecasting, the method of 4d-Var has been adopted by many numerical weather prediction (NWP) agencies as it is flexible enough to allow a range of atmospheric observations of many different types to be digested within a framework of a numerical model of the atmosphere. This has proved to be a valuable tool in estimating the initial conditions of a weather prediction

model, which are essential for a good forecast. In 4d-Var the method supersedes the simpler and less expensive "3d-Var", in which no proper account is made of the time of which an observation is made. Although the method of 4d-Var described in these notes is not restricted to any particular system, the application described here has a NWP model at its core, and the parameters to be determined are the model's initial conditions.

The optimization problem is quite demanding and all of the following issues must be considered:

- When determining the optimal solution (also known as the *analysis*, *i.e.* the best estimate of the initial conditions), account should be taken of the time evolution of the system. In the context of NWP this means that the best fit should be consistent not only with the observations themselves, but also with the governing equations (e.g. the laws of physics) in whatever form they are represented in the model.
- All measurements and model states have uncertainties and contain spatial correlations. The solution should reflect this, with largest consideration given to that knowledge of the system which is of highest accuracy. The term 'knowledge' in this context does not refer only to the observational data, but also to the information contained in model forecasts made previously (the *background state* - see below). The method should incorporate a representation of how the known errors of the background state and of the observations combine in the analysis.
- The procedure must cope with observations spread in both space and time, and the optimal solution should exist even in regions with no observational data.

All of the above aspects are automatically taken into account in the method of 4d-Var. Optimization, as we have defined it, should be regarded as a class of *inverse problem*. Indeed, we assume that the *forward problem* is soluble. That is, given a set of initial conditions, a set of *forward models* can be run to predict the observations. The most important forward model is the core NWP forecast model which gives the evolution in time of the model's state. Other forward models are important but secondary. These have as their input a model state (pertaining to a particular time), and predict the observables at that time. Examples of such predicted observations commonly used in an NWP context are of the meteorological variables (wind, temperature and humidity) interpolated from the model grid to the position of an instrument (in a field station, or on a sonde or aircraft), a layer mean mixing ratio (e.g. water vapour) as derived in a satellite profile, or a radiance measurement as observed by satellite. The predicted 'observations' may or may not match the actual observations; this depends on the choice of initial conditions (and on the suitability of the models). We are interested in solving the inverse problem which can be posed as follows: what set of initial conditions will seed the models to best predict the known observations?

Such inverse problems are in general much harder to solve than the forward analogue. Indeed, the inverse problem relies on the existence of the forward models themselves which are run many times in an iterative fashion to give the analysis (Fig. 1).

4d-Var is a powerful way of interfacing models and observations in this way. This is a difficult task as observations are generally not suitable for direct initialization of models. For the application of weather forecasting, it is sometimes confusing to understand our interest in initializing a model to run forward over time, some of which has already passed (after all, if observations have been made then the weather has happened). The answer is to develop a model state which is representative of the actual weather just before a forecast model is launched. This state is expected to yield the best possible forecast.

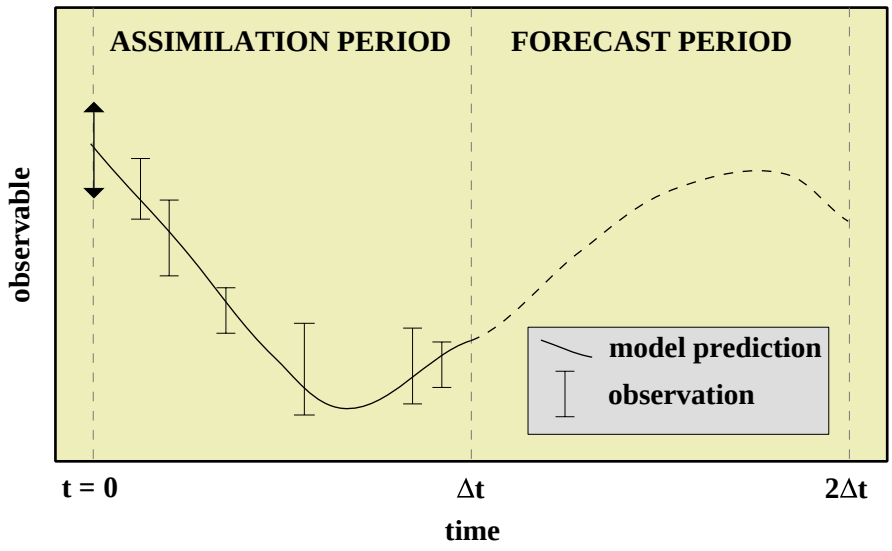


Figure 1: the model 'trajectory' - the model's prediction of an observable quantity (curve) - is shown which best fits the set of observations (bars). The height of the bars denote the degree of uncertainty in the measurements. The method of 4d-Var involves adjusting the model's initial conditions (bold arrow at $t = 0$) for this best fit.

Although the principle product of data assimilation is to obtain well initialized forecasts, other important 'spin-offs' have emerged. Global and complete model analyses (four dimensional datasets of meteorological variables) are routinely produced from historical data to aid the science of climate. Well known examples include the European Reanalysis datasets of the European Centre for Medium Range Weather Forecasts (ERA-15 or ERA-40) and the global dataset of the National Center for Environmental Protection. The technique also provides a valuable quantification of model performance and a measure of observation quality.

2. THE COST FUNCTION

The approach of this inverse modelling technique is to find the initial conditions of a model, $\boldsymbol{x}(0)$, such as to minimize some scalar quantity, J. J is known as the *cost* or *penalty* function. $J[\boldsymbol{x}]$ is a functional of the state vector $\boldsymbol{x}$, and is defined to be a global measure of the simultaneous misfit between $\boldsymbol{x}$, the current guess of the true atmospheric state, and two independent 'versions' of the atmospheric representation. One of these versions is that according to the observations themselves, and the other is taken from a model forecast. The latest observations (for a manageable time slice of 6-12 hours) are contained in the vector $\boldsymbol{y}$. Using observations alone for the purpose of initializing a model is inadequate as measurements are not evenly spread over the atmosphere. Furthermore the number of degrees of freedom in the model (the size of the vector space, N, describing $\boldsymbol{x}$) far outweighs the number of reliable observations taken within the time period, and different observations do not necessarily provide independent pieces of information. To help overcome this problem *a priori* information is needed. This comes from a previous model forecast, and is the second version of the atmospheric state used in the cost function. This state is called the *background*, $\boldsymbol{x}_B$, valid at time $t = 0$, serving as an information 'filler' for data voids.

We can now proceed to define a form of J which simultaneously penalizes a bad fit between (i) the model state $\boldsymbol{x}$ and the background, and the (ii) model state and the predicted observations [1]. The usual form of J reflects these two contributions (J_B and J_O respectively):

$$J[\boldsymbol{x}] = \frac{1}{2}(\boldsymbol{x}_B - \boldsymbol{x})^T \mathbf{B}^{-1}(\boldsymbol{x}_B - \boldsymbol{x}) + \frac{1}{2}\sum_{t=0}^{\Delta t}(\boldsymbol{y}(t) - \boldsymbol{H}_t^o[\boldsymbol{x}(t)])^T \mathbf{E}^{-1}(\boldsymbol{y}(t) - \boldsymbol{H}_t^o[\boldsymbol{x}(t)])$$

$$= J_B + J_O, \qquad (1)$$

where here and in the remainder of this report, the model state, $\boldsymbol{x}$, applies to the initial time $t = 0$ (defined to be at the start of the current assimilation cycle) unless explicitly labelled otherwise. Other symbols are the following: $\mathbf{B}$ is the *background error covariance matrix*, $\boldsymbol{y}(t)$ is the set of observations made at time t, $\boldsymbol{H}_t^o[\boldsymbol{x}(t)]$ is the part of the forward model which predicts the observations based on the model's current state, and $\mathbf{E}$ is the *observational error covariance matrix*. The vector $\boldsymbol{y}(t) - \boldsymbol{H}_t^o[\boldsymbol{x}(t)]$ is known as a *residual*.

We have made various assumptions when writing down this cost function. Firstly, we have assumed that the errors are unbiased and Gaussian [1]. If the former condition is not met, and the biases are known, observations can be corrected in the preprocessing performed on the raw observations when compiling $\boldsymbol{y}$. Another assumption is that the model is 'perfect' over the assimilation period. This leads to the so-called *strong constraint* formalism as used in Eq. (1). We could write an alternative cost function with a third term which is the additional constraint which

penalises model error. This would be an example of the weak constraint formalism [2].

The compactness of the definition of J in Eq. (1) is good for brevity, but the apparent simplicity is somewhat misleading, as the underlying operators are complicated and difficult to deal with. Before we explain why, let us describe each term. The form of each term is, assuming any non-linearity in the forward models to be weak, roughly quadratic in $\boldsymbol{x}$. For this reason, the process of minimising J is a generalisation of the *method of least squares*. A cost function which is exactly quadratic is guaranteed to possess a global minimum.

2.1 The background penalty

The state vectors $\boldsymbol{x}$ and $\boldsymbol{x}_B$ exist in the model space and consist of N elements. The meaning of each element depends upon the type of model and how it represents the meteorological fields. In a grid-point model, e.g., the vectors will contain information pertaining to the values of each field (e.g. u, v, θ, p and q to use standard meteorological symbols for zonal wind, meridional wind, potential temperature, pressure and water vapour mixing ratio respectively) at each grid position. In a spectral model, similar quantities will be stored for the weights of each spectral component at each level. Either way, a huge amount of information is stored in these vectors.

Even greater is the amount of information contained in **B** which consists of N^2 elements. The diagonal elements of **B** are the *variances* of the components of the background state (proportional to the square of the uncertainty in $\boldsymbol{x}_B$, provided that there are no biases). Due to the structure of J_B, components of $\boldsymbol{x}_B$ with large uncertainty will contribute little influence in the minimization. The off-diagonal elements are a measure of the correlation between different components and contain information regarding how the variational scheme couples different elements of the state vector. Strictly, **B** will contain covariances not only between the same quantity (e.g. θ) at different positions, but also covariances across different quantities (the latter elements are known as *multivariate* covariances). It is very important to have a reasonable knowledge of the elements of **B** as it contains crucial statistical information summarizing the behaviour of the system. Fundamentally, it is found by evaluating the matrix,

$$\mathbf{B} = (\boldsymbol{x}_B - \boldsymbol{x}_T)(\boldsymbol{x}_B - \boldsymbol{x}_T)^T, \tag{2}$$

where $\boldsymbol{x}_T$ is the true state of the atmosphere. This matrix is far too large to calculate, use, or even store in any practical way (let alone invert as required in the cost function). Besides, the true state of the atmosphere in Eq. (2) is unknown. Attempts have been made to represent **B** with far fewer pieces of information than N^2 and by using approximations to the truth. In weather forecasting, it is usual to transform from the meteorological variables (as listed above) to an

alternative set which are not strongly correlated (ie transforming to a set of *univariate* variables) and to treat vertical and horizontal covariances separately. For the purposes of this report we will not worry about this and presume that we know how to act with the inverse $\mathbf{B}^{-1}$ (**B** itself must be non-singular).

2.2 The observation penalty

$\boldsymbol{H}_t^o$ represents the observational forward model which appears in the observation term. It has already been mentioned in section 1. It acts directly on the model state $\boldsymbol{x}(t)$. The time evolution operator, $\boldsymbol{M}_t$, is required to produce this state given $\boldsymbol{x}$. This is done via the sequence,

$$\boldsymbol{x}(t) = \boldsymbol{M}_t \boldsymbol{M}_{t-\delta t} \ldots \boldsymbol{M}_{\delta t} \boldsymbol{x}. \tag{3}$$

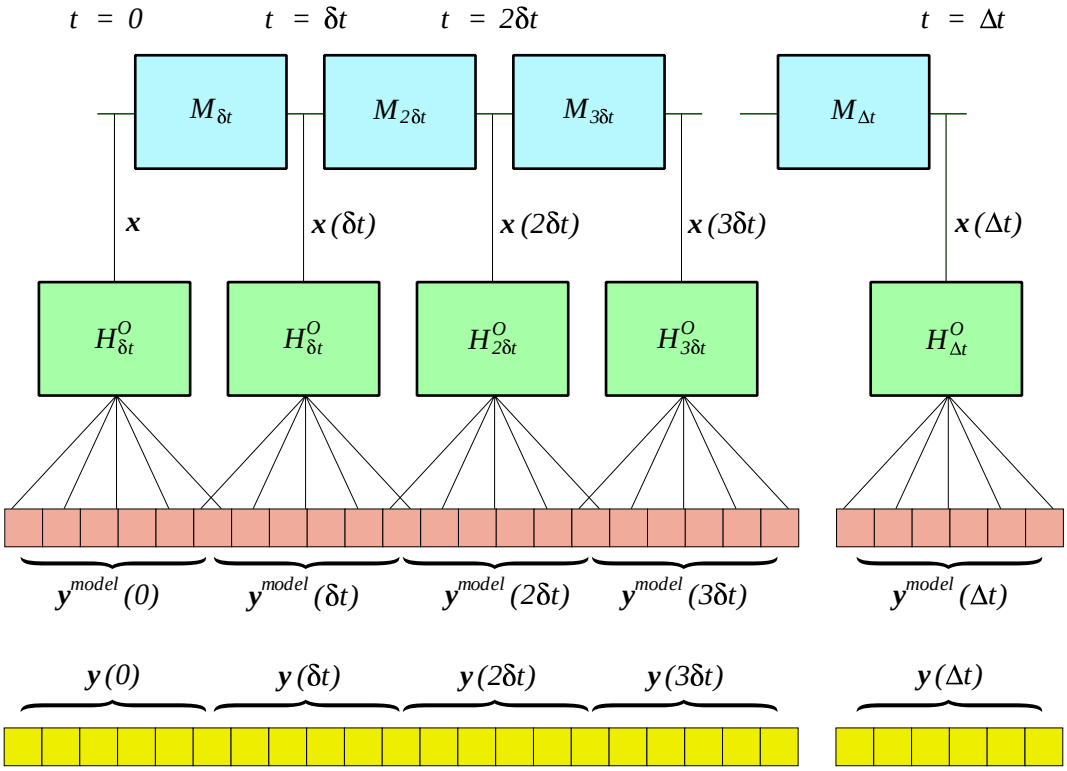


Figure 2: schematic illustration of the purpose of the forward models. The time evolution operators, $\boldsymbol{M}_t$ (blue boxes) move the model state forward in time, and the observation operators (green boxes), $\boldsymbol{H}_t^o$, use each model state to predict the observations close to their proper times. $\boldsymbol{M}_t$ could be a NWP forecast model (often linearized) and $\boldsymbol{H}_t^o$ is itself subdivided into parts pertaining to the different observation types. The mathematical form of the combined operator is given in Eq. (4). The model-predicted observations form the pink vector which are compared with the actual observations stored in the yellow vector.

Due to the complexity of the $\boldsymbol{M}_t$ and $\boldsymbol{H}_t^o$ forward models, J_O is the more complicated of the two penalty terms. Given the initial conditions, $\boldsymbol{x}$, the model's prediction of the observations at time t

is,

$$\boldsymbol{y}^{model}(t) \;=\; \boldsymbol{H}_t^o[\boldsymbol{x}(t)] \;=\; \boldsymbol{H}_t^o[\boldsymbol{M}_t\boldsymbol{M}_{t-\delta t}\ldots\,\boldsymbol{M}_{\delta t}\boldsymbol{x}]\,, \tag{4}$$

which is illustrated in Fig. 2 for all times in the time window. The lengths of each observation vector pertaining to the each time can be different in each case to cope with the asynoptic nature of the available observations.

Both the time evolution and observation types of forward model are generally non-linear, although observation operators for some observations involve only linear interpolation. The allowance of non-linear forward models in 4d-Var means, importantly, that information directly from satellite radiances (which are a complicated non-linear function of the temperature, pressure and chemical constituents along the path of the radiation into the instrument) can, in principle, be assimilated directly into the system. The implementation of such operators has had a striking positive impact on the success of weather forecasts [3].

Errors in the observations are represented by the diagonal elements of **E**. Observations are usually independent pieces of information and are thus uncorrelated. For this reason, **E** is often taken to be diagonal and is then easy to invert. Some observations though will be highly pre-processed (e.g. satellite retrieval profiles) and strictly are correlated.

3. MINIMIZING THE COST FUNCTION

The whole process of finding the initial conditions, $\boldsymbol{x}$ can be summarised as the task of finding the set of N elements of $\boldsymbol{x}$ which minimise the value of J. In the limit of linear operators, the definition of J ensures that it is real and positive semi-definite ($J \geqslant 0$). At first, we can only guess what $\boldsymbol{x}$ is and its value is refined iteratively by a *descent algorithm* [4] which we shall not discuss here. The first guess of $\boldsymbol{x}$ is often taken to be the background, $\boldsymbol{x}_B$, as it is readily available.

For the descent algorithm to work, it does not actually need to evaluate the value of J itself (although it is a useful diagnostic). Only its gradient with respect to the initial conditions need be found. This is,

$$\nabla_x J \;\equiv\; \left(\frac{\mathrm{d}J}{\mathrm{d}\boldsymbol{x}}\right)^T, \tag{5}$$

which will be used, e.g., by a *steepest descent* algorithm to find how $\boldsymbol{x}$ should be modified to reduce the value of J. At the minimum, J is stationary ($\nabla_x J = 0$). The derivative of Eq. (5) is a column vector containing N components, the ith one being,

$$(\nabla_x J)_i \equiv \frac{\partial J}{\partial x^i}, \tag{6}$$

x^i being the the ith component of $\boldsymbol{x}$. The reason for the presence of the transpose instruction in Eq. (5) is to make the differential operator $\mathrm{d}J/\mathrm{d}\boldsymbol{x}$ (which is by convention a row vector) into a column as required. Although this point seems unimportant at this stage, it is crucial to define our terms properly now as such details will become significant later, especially when we discuss the *adjoint method*.

In order to see how the gradient will help us minimize J, consider the cost function expanded as a Taylor series about the guess state generalised to the many variables contained in the vector $\boldsymbol{x}$. To second order:

$$J[\boldsymbol{x} = \boldsymbol{x}_G + \delta\boldsymbol{x}] = J[\boldsymbol{x}_G] + \left.\frac{\mathrm{d}J}{\mathrm{d}\boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_G} \delta\boldsymbol{x} + \frac{1}{2}\delta\boldsymbol{x}^T \left.\left(\left(\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{x}}\right)^T \frac{\mathrm{d}J}{\mathrm{d}\boldsymbol{x}}\right)\right|_{\boldsymbol{x}=\boldsymbol{x}_G} \delta\boldsymbol{x}, \tag{7}$$

where $\boldsymbol{x_G}$ is a guess of the analysis (in the first instance this could be taken as $\boldsymbol{x_B}$). Contained in the last term of Eq. (7) is the Hessian matrix written in compact notation. Expanded, the Hessian is,

$$\mathrm{Hess} = \left(\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{x}}\right)^T \frac{\mathrm{d}J}{\mathrm{d}\boldsymbol{x}} = \begin{pmatrix} \partial^2 J/\partial(x^1)^2 & \partial^2 J/\partial x^1 \partial x^2 & \cdots & \partial^2 J/\partial x^1 \partial x^N \\ \partial^2 J/\partial x^2 \partial x^1 & \partial^2 J/\partial(x^2)^2 & \cdots & \partial^2 J/\partial x^2 \partial x^N \\ \cdots & \cdots & \cdots & \cdots \\ \partial^2 J/\partial x^N \partial x^1 & \partial^2 J/\partial x^N \partial x^2 & \cdots & \partial^2 J/\partial(x^N)^2 \end{pmatrix}. \tag{8}$$

We require the increment, $\delta\boldsymbol{x}$, which should be added to $\boldsymbol{x_G}$ to make J stationary. Differentiating J in Eq. (7) with respect to $\delta\boldsymbol{x}$, and setting to zero for a stationary value gives (this procedure is similar to that made explicitly to the more complicated cost function as shown in section 3.1),

$$\nabla_{\boldsymbol{x}} J = \nabla_{\boldsymbol{x}} J\big|_{\boldsymbol{x}=\boldsymbol{x}_G} + \mathrm{Hess}\big|_{\boldsymbol{x}=\boldsymbol{x}_G} \delta\boldsymbol{x} = 0. \tag{9}$$

Rearranged for $\delta\boldsymbol{x}$, Eq. (9) gives,

$$\delta\boldsymbol{x} = -(\mathrm{Hess}\big|_{\boldsymbol{x}=\boldsymbol{x}_G})^{-1} \nabla_{\boldsymbol{x}} J\big|_{\boldsymbol{x}=\boldsymbol{x}_G}, \tag{10}$$

which shows that the required increment is related to the gradient in a linear sense. This result is nothing more than a generalisation of the Newton-Raphson method, here applied to finding roots of the gradient of a multi-dimensional function. Note that the inverse Hessian has, in general, the effect of altering the direction and length of the gradient vector.

As an aside, this stresses the importance of *preconditioning*, which is the process of choosing new variables (call the state vector represented in these new variables $\boldsymbol{v}$, related to $\boldsymbol{x}$ via a transformation) which are uncorrelated and which have unit variance. For perfectly preconditioned variables, the Hessian measured with respect to changes in the components of $\boldsymbol{v}$ would be the unit matrix and, in $\boldsymbol{v}$-space,

$$\delta \boldsymbol{v} \;=\; -\nabla_{\boldsymbol{v}} J\big|_{\boldsymbol{v}=\boldsymbol{v}_G}. \tag{11}$$

In the remainder of this report we shall return to $\boldsymbol{x}$-space as we are more interested here in the method of 4d-var, rather than in preconditioning techniques.

3.1 The gradient of J

We proceed to calculate the gradient of J. In order to do this explicitly from first principles, Eq. (1) can be expanded from its usual 'matrix form' into 'series form',

$$J[\boldsymbol{x}] \;=\; \frac{1}{2}\sum_{ij}(x_B^i - x^i)(B^{-1})_{ij}(x_B^j - x^j) \;+\; \frac{1}{2}\sum_t\sum_{mn}(y^m(t) - H_t^{o,m}[\boldsymbol{x}(t)])(E^{-1})_{mn}(y^n(t) - H_t^{o,n}[\boldsymbol{x}(t)]), \tag{12}$$

where the indices i, j run over model components, m, n run over observation components, and t runs over time steps (each of length δt) in the time window. The gradient with respect to the individual component, x^k, is (making use of the product rule),

$$\frac{\partial J}{\partial x^k} \;=\; -\frac{1}{2}\sum_{ij}\Big((x_B^i - x^i)(B^{-1})_{ij}\,\delta_{jk} \;+\; \delta_{ik}(B^{-1})_{ij}(x_B^j - x^j)\Big) \;-\; \frac{1}{2}\sum_t\sum_{mn}\left((y^m(t) - H_t^{o,m}[\boldsymbol{x}(t)])(E^{-1})_{mn}\frac{\partial H_t^{o,n}[\boldsymbol{x}(t)]}{\partial x^k} \;+\; \frac{\partial H_t^{o,m}[\boldsymbol{x}(t)]}{\partial x^k}(E^{-1})_{mn}(y^n(t) - H_t^{o,n}[\boldsymbol{x}(t)])\right), \tag{13}$$

where x^k is the kth component of the initial state of the model (in the state of its current guess). This expression can be tidied up by making use of covariance matrix symmetry,

$$(B^{-1})_{ij} \;=\; (B^{-1})_{ji} \qquad , \qquad (E^{-1})_{mn} \;=\; (E^{-1})_{nm}, \tag{14}$$

$$\therefore\; \frac{\partial J}{\partial x^k} \;=\; -\sum_i (B^{-1})_{ki}(x_B^i - x^i) \;-\; \sum_t\sum_{mn}\frac{\partial H_t^{o,m}[\boldsymbol{x}(t)]}{\partial x^k}(E^{-1})_{mn}(y^n(t) - H_t^{o,n}[\boldsymbol{x}(t)]). \tag{15}$$

All components of $\nabla_x J$ can be assembled into a column vector. This is compactly written by reverting to matrix form,

$$\nabla_x J \;=\; -\mathbf{B}^{-1}(\boldsymbol{x}_B - \boldsymbol{x}) \;-\; \sum_t\left(\frac{\mathrm{d}\boldsymbol{H}_t^o[\boldsymbol{x}(t)]}{\mathrm{d}\boldsymbol{x}}\right)^T \mathbf{E}^{-1}(\boldsymbol{y}(t) - \boldsymbol{H}_t^o[\boldsymbol{x}(t)]). \tag{16}$$

A point on linear algebra: the object $\mathrm{d}\boldsymbol{H}_t^o[\boldsymbol{x}_t]/\mathrm{d}\boldsymbol{x}$ in Eq. (16) is a matrix, known as a *Jacobian*, and its elements are,

$$\left(\frac{\mathrm{d}\boldsymbol{H}_t^o[\boldsymbol{x}(t)]}{\mathrm{d}\boldsymbol{x}}\right)_{mk} \;=\; \frac{\partial H_t^{o,m}[\boldsymbol{x}(t)]}{\partial x^k}, \tag{17}$$

which is the sensitivity of the mth model-predicted observation at time t due to changes in the kth component of the model's initial condition vector. It has been transposed in Eq. (16) to work with the matrix notation.

Forward models which predict the subset of observations at a time t operate on the model state at that time, yet the sensitivities highlighted in Eq. (17) are made with respect to the *initial* state of the model. A more natural Jacobian to calculate would be one like,

$$\left(\frac{\mathrm{d}\boldsymbol{H}_t^o[\boldsymbol{x}(t)]}{\mathrm{d}\boldsymbol{x}(t)}\right)^T \qquad \text{rather than} \qquad \left(\frac{\mathrm{d}\boldsymbol{H}_t^o[\boldsymbol{x}(t)]}{\mathrm{d}\boldsymbol{x}}\right)^T, \tag{18}$$

where the derivatives on the left hand side are made with respect to the model state at the same time that the observation forward model is relevant to. We can facilitate this change by using the generalised chain rule [5]. Noting the relationship between $\boldsymbol{x}(t)$ and $\boldsymbol{x}$ (Eq. (3)), then the chain rule for derivatives gives,

$$\left(\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{x}}\right)^T = (\mathbf{M}_t\mathbf{M}_{t-\delta t}\ldots\,\mathbf{M}_{\delta t})^T\left(\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{x}(t)}\right)^T, \tag{19}$$

where the non-linear model operators (in bold italic font in Eq. (3)) have been linearised (bold Roman font in Eq. (19), e.g. element i,j of matrix $\mathbf{M}_t$ is $(\mathbf{M}_t)_{ij} = \partial \boldsymbol{x}_i(t)/\partial \boldsymbol{x}_j(t-\delta t)$). Inserting this transformation into Eq. (16) yields,

$$\nabla_x J \approx -\mathbf{B}^{-1}(\boldsymbol{x}_B - \boldsymbol{x}) - \sum_t (\mathbf{M}_t\mathbf{M}_{t-\delta t}\ldots\,\mathbf{M}_{\delta t})^T\left(\frac{\mathrm{d}\boldsymbol{H}_t^o[\boldsymbol{x}(t)]}{\mathrm{d}\boldsymbol{x}(t)}\right)^T\mathbf{E}^{-1}(\boldsymbol{y}(t) - \boldsymbol{H}_t^o[\boldsymbol{x}(t)]). \tag{20}$$

In Eq. (20) we have linearized the adjoint time evolution operators (the time evolution operator implied in the innovation vector remains non-linear) as it is unclear how to find the adjoint of non-linear terms (this results in the gradient calculation being approximate). In this context, these operators are *perturbation forecast* operators. The differential $\mathrm{d}\boldsymbol{H}_t^o[\boldsymbol{x}(t)]/\mathrm{d}\boldsymbol{x}(t)$ is effectively a linearization of the forward observation operator. We let,

$$\frac{\mathrm{d}\boldsymbol{H}_t^o[\boldsymbol{x}(t)]}{\mathrm{d}\boldsymbol{x}(t)} = \mathbf{H}_t^o, \tag{21}$$

distinguishing again non-linear bold italic operators from matrix bold Roman operators. This makes the expression for the gradient into,

$$\nabla_x J \approx -\mathbf{B}^{-1}(\boldsymbol{x}_B - \boldsymbol{x}) - \sum_t \mathbf{M}_{\delta t}^T\ldots\,\mathbf{M}_{t-\delta t}^T\mathbf{M}_t^T\mathbf{H}_t^{oT}\mathbf{E}^{-1}(\boldsymbol{y}(t) - \boldsymbol{H}_t^o[\boldsymbol{x}(t)]). \tag{22}$$

In Eq. (22) we have also applied the transpose action to each member in the sequence of time evolution operators of Eq. (20) separately (remember that this involves reversing their order).

3.2 Evaluating the gradient of J

We can evaluate the gradient of J with respect to the initial conditions in one of three ways.

1. Evaluate finite differences for the gradient of J without using the results shown above.

For component i, the gradient is given as Eq. (23) below for a centred difference scheme, which must be performed for each i. Although this approach is useful to understand the meaning of the gradient, it is too costly in practice, requiring $2N$ evaluations of the cost function. Typically for a global forecast model, $N \sim 10^6$ (or greater), and each evaluation would require a forward integration of the model over the time period $0 \rightarrow \Delta t$, which is extremely prohibitive.

$$\frac{\partial J}{\partial x^i} \approx \frac{J(x^1, \ldots, x^i + \delta x, \ldots, x^N) - J(x^1, \ldots, x^i - \delta x, \ldots, x^N)}{2\delta x}. \qquad (23)$$

2. Evaluate Eq. (22) directly. This would involve, at each timestep, acting on $[\boldsymbol{y}(t) - \boldsymbol{H}_t^o[\boldsymbol{x}(t)]$ with the sequence of operators $\mathbf{M}_{\delta t}^T \ldots \mathbf{M}_{t-\delta t}^T \mathbf{M}_t^T \mathbf{H}_t^{oT} \mathbf{E}^{-1}$. Although better than finite differencing, evaluating the time series in this way is still inefficient as there is a lot of duplication as the same time evolution operators are used in many of the terms in the time summation.
3. Use the adjoint technique [6] (section 3.3). This is at the heart of 4d-Var and is an efficient means of calculating the gradient. The method does not explicitly store or use matrices. Instead, an operator acts by applying a set of rules (in a program, this would best be done by a subroutine). The absence of matrices makes the transpose (or *adjoint*) operator difficult to deal with which means that a set of code, pertaining to the adjoint operator, needs to be implemented separately. This is possible if the 'normal' operator is linear, or has been linearized, and there are mechanical methods of doing this given the normal operator [7].

3.3 The adjoint method

We identify an efficiency in evaluating Eq. (22) if we write out the sum in the following way,

$$\begin{array}{l|l}
t = \Delta t & -\boldsymbol{M}_{\delta t}^T \quad \boldsymbol{M}_{2\delta t}^T \quad \boldsymbol{M}_{\Delta t-\delta t}^T \quad \boldsymbol{M}_{\Delta t}^T \quad \boldsymbol{H}_{\Delta t}^{oT} \mathbf{E}^{-1} \boldsymbol{r}(\Delta t) + \\
t = \Delta t - \delta t & -\boldsymbol{M}_{\delta t}^T \quad \boldsymbol{M}_{2\delta t}^T \quad \boldsymbol{M}_{\Delta t-\delta t}^T \quad \boldsymbol{H}_{\Delta t-\delta t}^{oT} \mathbf{E}^{-1} \boldsymbol{r}(\Delta t - \delta t) + \\
 & \qquad\qquad\qquad\qquad\qquad\qquad \boldsymbol{H}_{\Delta t-2\delta t}^{oT} \mathbf{E}^{-1} \boldsymbol{r}(\Delta t - 2\delta t) + \\
 & \qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
t = 2\delta t & -\boldsymbol{M}_{\delta t}^T \quad \boldsymbol{M}_{2\delta t}^T \\
t = \delta t & -\boldsymbol{M}_{\delta t}^T \quad \boldsymbol{H}_{\delta t}^{oT} \mathbf{E}^{-1} \boldsymbol{r}(\delta t) + \\
t = 0 & -\boldsymbol{H}_0^{oT} \mathbf{E}^{-1} \boldsymbol{r}(0) + \\
b/g & -\mathbf{B}^{-1}(\boldsymbol{x}_B - \boldsymbol{x})
\end{array}$$

where,

$$\boldsymbol{r}(t) \equiv \boldsymbol{y}(t) - \boldsymbol{H}_t^o(\boldsymbol{x}(t)), \tag{24}$$

is called the *residual*, and the background term is for time $t = 0$. Each row in the above sum is a term in the summation of Eq. (22), with the time corresponding to each given on the left hand side. The rows are all added together and the coloured columns serve only to highlight the fact that specific time evolution operators are repeated in different rows (operators of the same colour background are the same). Written in this way, it is possible to see that we can evaluate the series in an alternative order by factorizing common operators according to the following prescription.

1. Perform an integration from $t = 0$ to $t = \Delta t$ using the forward (and non-linear) version of the forecast model. As the initial conditions, $\boldsymbol{x}$, are as yet undetermined, use a guess. This will be refined as a result of calculating the gradient of J. At each time step, store the model state $\boldsymbol{x}(t)$. This stage is that shown in Fig. 2.
2. Start the next stage of the algorithm at $t = \Delta t$ and evaluate the vector $\mathbf{H}_{\Delta t}^{o\,T}\mathbf{E}^{-1}\boldsymbol{r}(\Delta t)$ (call this $\lambda(\Delta t)$ - this appears on the extreme right hand side of the table above). Vectors of this kind (which will also be calculated at earlier times below) are known as *adjoint variables*.
3. Integrate the adjoint backwards in time by δt. This is done by replacing t with $t-\delta t$ and then calculating the term of Eq. (20) below (using the information, $\boldsymbol{x}(t)$, stored from step 1 to calculate $\boldsymbol{r}(t)$). Equation (20) may be proved by mathematical induction working from $t = \Delta t$ to $t = 0$. This action combines residual information from time t and adjoint information from time $t + \delta t$ to form an adjoint state at time t. This step comes from the above table: the first term of Eq. (20) appears below each coloured column, the adjoint time evolution operator, $\mathbf{M}_{t+\delta t}^T$, is that which appears inside the filled column above, and the adjoint variable, $\lambda(t + \delta t)$ represents everything to the right of the column.

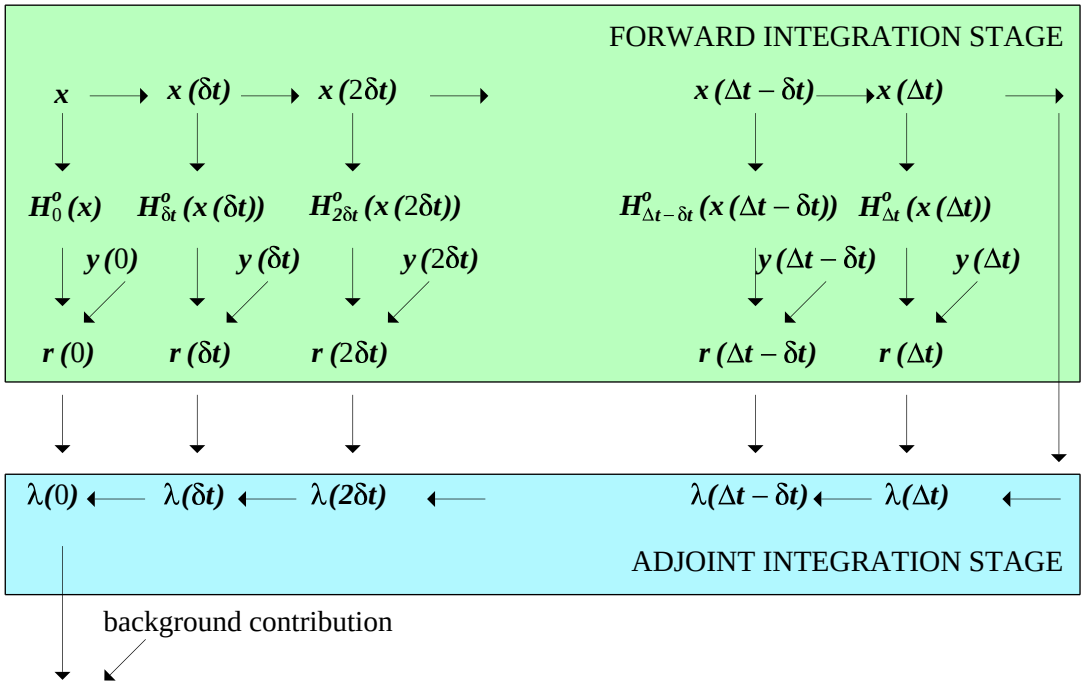
$$\lambda(t) = \mathbf{H}_t^{o\,T}\mathbf{E}^{-1}\boldsymbol{r}(t) + \mathbf{M}_{t+\delta t}^T\lambda(t+\delta t). \tag{25}$$

4. Go back to step 3 until t=0. At $t = 0$, $\lambda(0)$ is minus the gradient of J_O with respect to $\boldsymbol{x}$.
5. Calculate the full gradient by including the background (the full gradient is then used in the descent algorithm to refine our guess),

$$\nabla_x J \approx -\mathbf{B}^{-1}(\boldsymbol{x}_B - \boldsymbol{x}) - \lambda(0). \tag{26}$$

This method is called the *adjoint method* due to the use of adjoint variables, λ, in step 3 (Eq. (25)) and the continuous operation with adjoint (transpose) operators. It is the efficiency of the adjoint technique which has meant that 4d-Var is practical.

Applied in this form, the method has two stages (Fig. 3): (i) a forward integration of the model variables (step 1) followed by (ii) a backward integration of the adjoint variables by the adjoint model (steps 2 to 4). The final step (step 5) just adds on the background contribution. Once the gradient is calculated, the descent algorithm is applied and finds a refined initial condition, $\boldsymbol{x}$, with which the whole process is repeated. The optimal state is where $\nabla_x J = 0$ (by suitable convergence) which means that the cost function has been minimized.



$$\nabla_x J \approx -\mathbf{B}^{-1}(\boldsymbol{x}_B - \boldsymbol{x}) - \lambda(0)$$

Figure 3: schematic illustration of the adjoint 4d-Var algorithm. The forward integration (green) is essentially the same process as that represented in Fig. 2, but with the additional residual information calculated. The integration of the adjoint variable, λ, is done backwards in time (blue box). In this Fig., $\boldsymbol{x}(t)$ is the model state at time t, $\boldsymbol{H}_t^o$ is the observational forward operator, $\boldsymbol{y}(t)$ is the set of observations at time t, $\boldsymbol{r}(t)$ is the residual (Eq. 19), and $\lambda(t)$ is the adjoint at time t. The vector $\lambda(0)$ is the gradient (with respect to $\boldsymbol{x}$) of the observational component of the cost function (Eq. (1)) which is combined with the background gradient to give the total gradient.

4. GLOSSARY OF TERMS

3d-Var Variational data assimilation in which, during a particular assimilation period, no account is taken of the time that observations are taken. The three dimensions are spatial. Although adjoint operators are used, there is, unlike 4d-var, no integration of the adjoint variables back in time.

4d-Var Variational data assimilation within three space dimensions plus one time

dimension. The times that observation are made is resolved by the assimilation system. The adjoint method is at the heart of the technique.

Adjoint The adjoint is the transpose of a matrix or vector (if complex numbers are present, this should be accompanied by a complex conjugate of the elements). The adjoint turns rows into columns and vice-versa. Some operators are effectively matrices (linear operators), but cannot, due to storage limitations, be stored explicitly. The adjoint operator must then be formulated as a separate operator either manually or by automatic adjoint software. In 4d-Var, adjoint operators (matrices) help integrate the adjoint variables (cost function gradient vectors) backwards in time.

Analysis The initial conditions of a forecast model that 'best-fits' (via a sequence of time evolution and observation operators) a set of observations and a background state (also known as optimum analysis). Usually the analysis is initialized before the forecast is run (see the initialization entry).

Assimilation period The window of time in which observations are taken for the optimization (assimilation) process. The forecast period follows.

Background A model state which is the *a-priori* guess at the true state of the system. It is a forecast which has been started from the analysis at the previous assimilation cycle and is used, in conjunction with a set of observations, to help find the new analysis state. The background state appears in the cost function (Eq. (1)).

Control variables The variables that are actually adjusted during the descent algorithm. These are often not the meteorological variables in model space, but are some transformation involving a new set of parameters and in a new vector space. The gradient of the cost function with respect to each new control variable is required for the descent algorithm. The transformation to control variables is part of the preconditioning process.

Cost function A scalar quantity measuring the misfit between a guess of the model state and the background, and the guess (postprocessed to predict the observations) and the observations themselves. The cost function is minimised iteratively in variational data assimilation. Also known as the penalty function.

Covariance How one quantity varies with another. A covariance error matrix is a systematic means of storing the statistical information of how errors in each quantity represented depend on the errors in others. The covariance between x and y is found from: covariance $= \langle (x - \langle x \rangle)(y - \langle y \rangle) \rangle$ where $\langle\rangle$ = mean.

Descent algorithm An algorithm that will adjust the state vector (in control variable space) to yield a new value of the cost function which is lower. The aim of the algorithm is to minimize the cost function. The algorithm requires as its input the gradient of the cost function with respect to each control variable.

Errors Uncertainties of information pertaining to the system. All observations have errors as do all model states. Errors can be systematic (e.g. biases) or random. It is usual to correct for biases (if known) before the assimilation procedure starts and to assume that random errors are Gaussian in nature (this leads to the quadratic form of the cost function in 3d- and 4d-Var). Errors are related to variances.

Forecast period The period of time following the assimilation period where a model is run freely, and where no observations are available to correct it.

Forward model In 4d-Var there are two types of forward model. One kind predicts observations given a model state (observation operators). The other finds future model states from earlier ones (time evolution operator). In 3d-Var only the observation operators are used. 'Forward modelling' is regarded as the 'usual' means of modelling (as opposed to inverse modelling which tries to do

	the reverse).
Hessian	The matrix containing all of the second partial derivatives of the cost function, J, with respect to the control variables. Element i,j of the Hessian is $\partial^2 J / \partial x^i \partial x^j$ (see Eq. (8)).
Initialization	The processing of a set of uninitialized initial conditions for a model. The processing could, e.g., be a filter to remove undesirable gravity waves which could otherwise amplify and degrade the forecast.
Innovation vector	The difference between the observations and the model's version of the observations, found from the background state. The innovation vector is the residual (Eq. (24)) in the special case that the model integration is started from the background state.
Jacobian	A transformation matrix detailing how components of one set of variables varies with another set.
Multivariate	A fully consistent system where account is taken of the correlations, not just between values of a quantity at different points in space, but between different quantities.
NWP	Numerical Weather Prediction (the application of computer models to weather forecasting).
Observation operator	An operator which predicts observations consistent with a model state.
Optimization	The process of finding the initial conditions of a forecast model to best-fit a set of observations and a background state. The methods of 3d- or 4d-var are optimization techniques.
Penalty function	See cost function.
Positive (semi) definite	A variable J is positive (semi) definite if it is bounded by $J > (\geqslant) 0$.
Post processing operator	See observation operator.
Preconditioning	The process of choosing new control variables which are exactly, or approximately uncorrelated, and form a unit Hessian matrix. In a perfectly preconditioned system of equations, contours of constant cost function are circles in the new control variable space and the preconditioning number is unity.
Preconditioning number	The ratio of the highest to the smallest eigenvalues of the Hessian. In a perfectly preconditioned system of equations, the preconditioning number is unity.
Residual	The difference between an observation and the model's version of it. The residual vector is a structure containing many such differences, each an element of the vector in Eq. (24).
Strong constraint	A formulation of a 4d-Var data assimilation system that assumes a perfect model, or where no account is taken of model error (as assumed here).
Time evolution operator	The set of rules which advances the model state forward in time (see Eq. (3)).
Univariate	This is the situation where one type of quantity is, or assumed to be, uncorrelated with another. Meteorological variables are rarely uncorrelated with each other (the multivariate case), but sometimes new variables are chosen that are approximately uncorrelated. This is part of the preconditioning procedure.

Variance The variance of a variable is related to the error by: error = $\sqrt{\text{variance}}$ (for a single measurement). The variance of the quantity x is found from: variance = $\langle (x - \langle x \rangle)^2 \rangle$ where $\langle\rangle$ is the mean. A variance is the special case of a covariance where both quantities (x and y in the glossary entry for covariance) are the same. Variances occupy the diagonal elements of a covariance matrix.

Weak constraint A formulation of a 4d-Var data assimilation system that takes into account model error.

5. REFERENCES

[1] A.C. Lorenc, Analysis methods for numerical weather prediction, Q.J.R. Meteorol. Soc 112, pp. 1177-1194 (1986).

[2] D. Zupanski, A general weak constraint applicable to operational 4d-var data assimilation systems, Mon. Wea. Rev. 125, pp. 2774-2292 (1997).

[3] A.P. McNally, J.C. Derber, W. Wu, B.B. Katz, The use of TOVS level-1b radiances in the NCEP SSI analysis system, Q.J.R. Meteorol. Soc. 126, pp. 689-724 (2000).

J.R. Eyre, G.A. Kelley, A.P. McNally, E. Anderson, A. Persson, Assimilation of TOVS radiance information through one-dimensional variational analysis, Q.J.R. Meteorol. Soc 119, pp. 1427-1463 (1993).

R. Saunders, E. Anderson, G.A. Kelley, R. Munro, B. Harris, Recent developments at ECMWF in the assimilation of TOVS radiances, ITSC-X proceedings, Boulder Co., USA, pp. 463-474 (Jan-Feb 1999).

R. Munro, G.A. Kelley, M. Rohn, R. Saunders, Assimilation of geostationary water vapour radiance data at ECMWF, ITSC-X proceedings, Boulder Co., USA, pp. 408-419 (Jan-Feb 1999).

C. Chouinary, J. Halle, The impact of TOVS radiances in the CMC 3D variational analysis system, ITSC-X proceedings, Boulder Co., USA, pp. 92-98 (Jan-Feb 1999).

[4] R. Daley, Atmospheric data analysis (section 13.1), Cambridge University Press (1996).

[5] R.N. Bannister, Applications of the chain rule, http://www.met.rdg.ac.uk/~ross/Documents/Chain.html (2001).

[6] C.D. Rodgers, Inverse methods for atmospheric sounding: theory and practice (section 8.2.3), Series on atmospheric, oceanic and planetary physics, Vol. 2, World Scientific (2000).

[7] R. Giering, T. Kaminski, Recipes for adjoint code construction, ACM Transactions on Mathematical Software 24, pp. 437-474 (1998).