4D-Var Data Assimilation (§5.6)

In OI and 3D-Var, the background error covariance matrix is estimated once and for all, as if the forecast errors were statistically stationary.

4D-Var Data Assimilation (§5.6)

In OI and 3D-Var, the background error covariance matrix is estimated once and for all, as if the forecast errors were statistically stationary.

The errors are estimated from the difference between the forecast and the analysis . . .

. . . that is, from the analysis increments.

4D-Var Data Assimilation (§5.6)

In OI and 3D-Var, the background error covariance matrix is estimated once and for all, as if the forecast errors were statistically stationary.

The errors are estimated from the difference between the forecast and the analysis . . .

. . . that is, from the analysis increments.

We can evaluate if this is indeed a good approximation.

4D-Var Data Assimilation (§5.6)

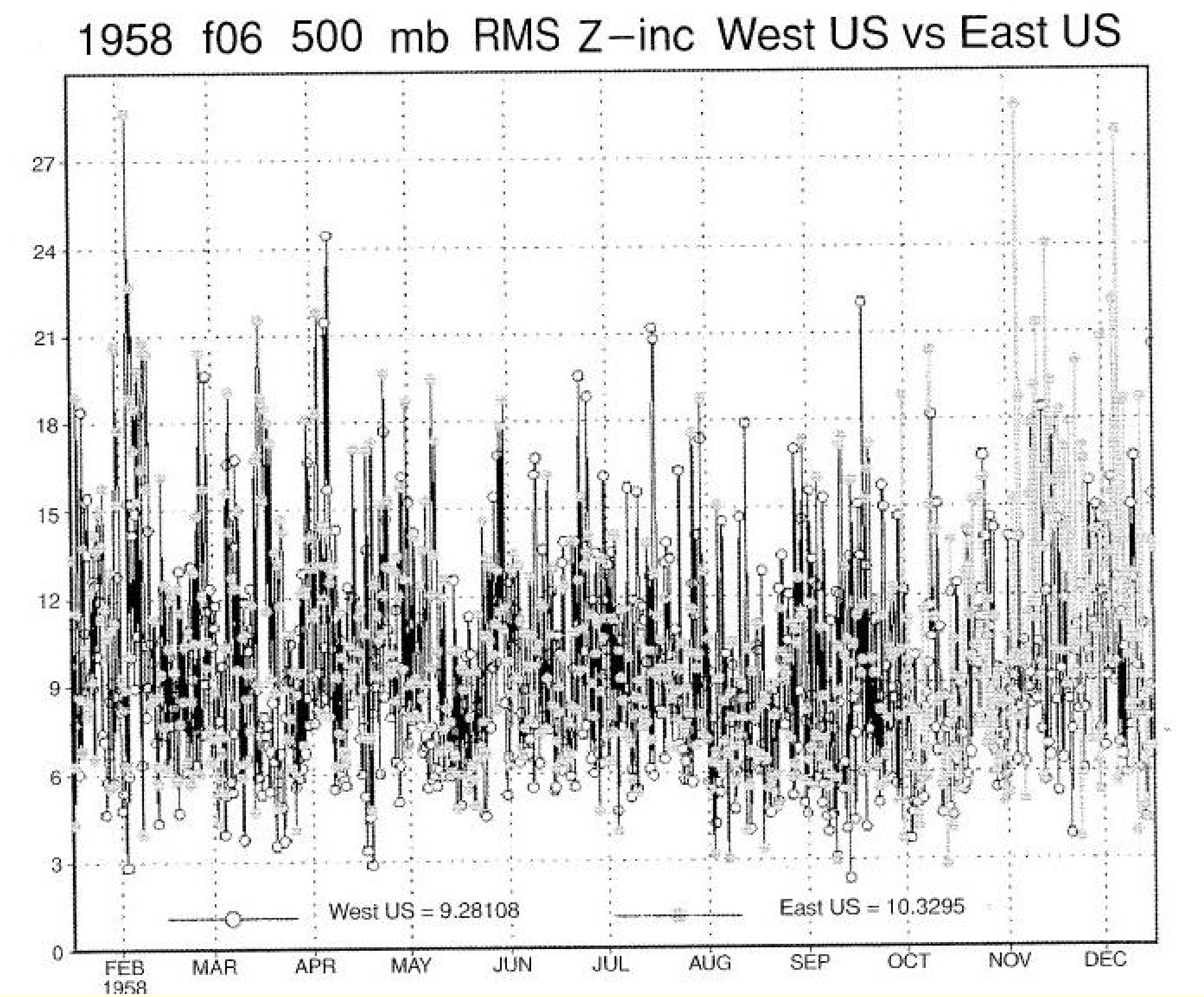
In OI and 3D-Var, the background error covariance matrix is estimated once and for all, as if the forecast errors were statistically stationary.

The errors are estimated from the difference between the forecast and the analysis . . .

. . . that is, from the analysis increments.

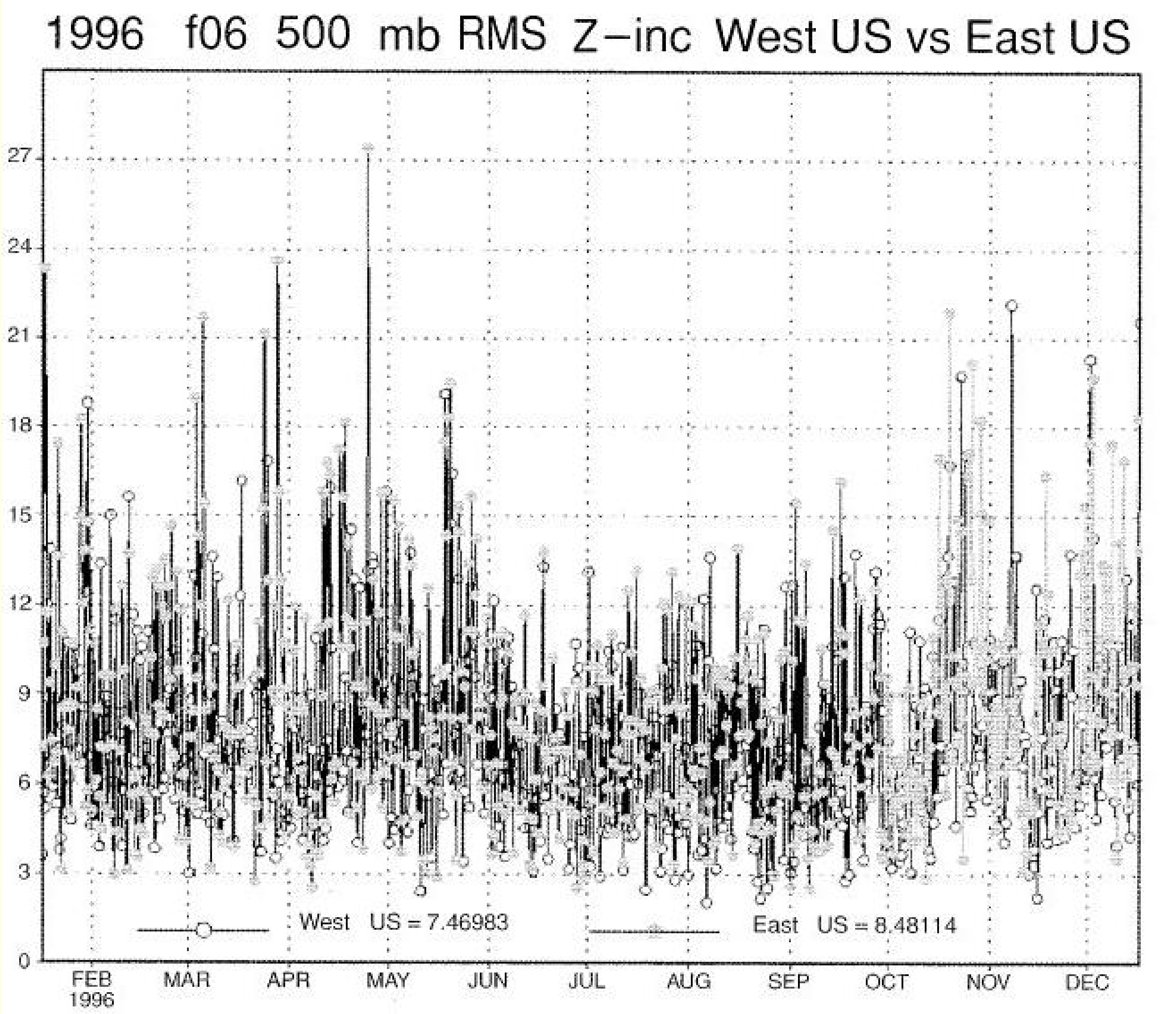
We can evaluate if this is indeed a good approximation.

The following figure shows the 6-h forecast errors over the USA from the NCEP/NCAR reanalysis.



Daily variation of the rms increment between the 6-h forecast and the analysis (NCEP-NCAR reanalysis).

1958: $\sigma \approx 10$ m



Daily variation of the rms increment between the 6-h forecast and the analysis (NCEP-NCAR reanalysis).

1996: $\sigma \approx 8$ m

The NCEP/NCAR reanalysis used a 3D-Var data assimilation system which did not change during the period.

The NCEP/NCAR reanalysis used a 3D-Var data assimilation system which did not change during the period.

Thus, the difference between the figures is due only to the changes in the observing system.

The NCEP/NCAR reanalysis used a 3D-Var data assimilation system which did not change during the period.

Thus, the difference between the figures is due only to the changes in the observing system.

Over these four decades the improvements in the observing system in the Northern Hemisphere show a positive impact.

The 6-h forecast errors decrease by about 20%, with the average analysis increment reduced from about 10 m to 8 m.

The most striking result apparent in the error statistics is that the day-to-day variability in the forecast error is about as large as the average error.

The most striking result apparent in the error statistics is that the day-to-day variability in the forecast error is about as large as the average error.

The figures emphasize the importance of the errors of the day which are dominated by baroclinic instabilities of synoptic time scales

The most striking result apparent in the error statistics is that the day-to-day variability in the forecast error is about as large as the average error.

The figures emphasize the importance of the errors of the day which are dominated by baroclinic instabilities of synoptic time scales

These errors are ignored when the forecast error covariance is assumed to be constant.

The most striking result apparent in the error statistics is that the day-to-day variability in the forecast error is about as large as the average error.

The figures emphasize the importance of the errors of the day which are dominated by baroclinic instabilities of synoptic time scales

These errors are ignored when the forecast error covariance is assumed to be constant.

The Kalman Filter technique predicts *both the model state and its error covariance.*

However, it is computationally very demanding, and is not practical for use in its complete form.

The most striking result apparent in the error statistics is that the day-to-day variability in the forecast error is about as large as the average error.

The figures emphasize the importance of the errors of the day which are dominated by baroclinic instabilities of synoptic time scales

These errors are ignored when the forecast error covariance is assumed to be constant.

The Kalman Filter technique predicts *both the model state and its error covariance.*

However, it is computationally very demanding, and is not practical for use in its complete form.

We will now consider four-dimensional variational assimilation (4D-Var), which has some of the advantages of Kalman Filtering.

The most striking result apparent in the error statistics is that the day-to-day variability in the forecast error is about as large as the average error.

The figures emphasize the importance of the errors of the day which are dominated by baroclinic instabilities of synoptic time scales

These errors are ignored when the forecast error covariance is assumed to be constant.

The Kalman Filter technique predicts *both the model state and its error covariance.*

However, it is computationally very demanding, and is not practical for use in its complete form.

We will now consider four-dimensional variational assimilation (4D-Var), which has some of the advantages of Kalman Filtering.

It includes, at least implicitly, the evolution of the forecast error covariance.

Model Error Covariance (Skip)

Let us represent the (nonlinear) model forecast that advances from time t_{i-1} to time t_i by

$$\mathbf{x}^f(t_i) = M_{i-1}\left[\mathbf{x}^a(t_{i-1})\right]$$

Since the model is imperfect, we write

$$\begin{aligned} \mathbf{x}^f(t_i) &= M_{i-1}[\mathbf{x}^t(t_{i-1})] \\ \mathbf{x}^t(t_i) &= M_{i-1}[\mathbf{x}^t(t_{i-1})] - \eta(t_{i-1}) \\ \mathbf{x}^f(t_i) &= \mathbf{x}^t(t_i) + \eta(t_{i-1}) \end{aligned}$$

The model error η is assumed to have zero mean, and covariance matrix $\mathbf{Q}_i = E(\eta_i \eta_i^T)$.

In other words, starting from perfect initial conditions, the forecast error is given by η_i.

(In reality model errors have significant biases, which must be taken into account.)

Note: **I am covering the following material in §5.6 of Eugenia Kalnay's book:**

- **Introductory paragraphs (pp. 175–177)**
- **§5.6.1, to the bottom of page 178**
- **§5.6.3, on 4D-Var**

I am *not* discussing Kalman Filtering in this course.

As this a topic of growing importance, you should read the remaining part of §5.6.1 (pages 179–180) and §5.6.2.

Tangent Linear Model

Consider the solution on the time interval t_i to t_{i+1}.

Tangent Linear Model

Consider the solution on the time interval t_i to t_{i+1}.

If we introduce a perturbation in the initial conditions, the final perturbation is given by

$$\begin{aligned} \mathbf{x}(t_{i+1}) + \delta\mathbf{x}(t_{i+1}) &= M_i\left[\mathbf{x}(t_i) + \delta\mathbf{x}(t_i)\right] \\ &= M_i\left[\mathbf{x}(t_i)\right] + \mathbf{L}_i\delta\mathbf{x}(t_i) + O(|\delta\mathbf{x}|^2) \end{aligned}$$

Tangent Linear Model

Consider the solution on the time interval t_i to t_{i+1}.

If we introduce a perturbation in the initial conditions, the final perturbation is given by

$$\begin{aligned}\mathbf{x}(t_{i+1}) + \delta\mathbf{x}(t_{i+1}) &= M_i\left[\mathbf{x}(t_i) + \delta\mathbf{x}(t_i)\right] \\ &= M_i\left[\mathbf{x}(t_i)\right] + \mathbf{L}_i\delta\mathbf{x}(t_i) + O(|\delta\mathbf{x}|^2)\end{aligned}$$

The matrix $\mathbf{L}_i$ is the linear tangent model operator

$$[\mathbf{L}_i]_{j,k} = \frac{\partial[M(\mathbf{x}(t_i)]_j}{\partial x_k(t_i)}$$

That is, it is the Jacobian of $M(\mathbf{x})$ with respect to x.

Tangent Linear Model

Consider the solution on the time interval t_i to t_{i+1}.

If we introduce a perturbation in the initial conditions, the final perturbation is given by

$$\begin{aligned}\mathbf{x}(t_{i+1}) + \delta\mathbf{x}(t_{i+1}) &= M_i\left[\mathbf{x}(t_i) + \delta\mathbf{x}(t_i)\right] \\ &= M_i\left[\mathbf{x}(t_i)\right] + \mathbf{L}_i\delta\mathbf{x}(t_i) + O(|\delta\mathbf{x}|^2)\end{aligned}$$

The matrix $\mathbf{L}_i$ is the linear tangent model operator

$$[\mathbf{L}_i]_{j,k} = \frac{\partial[M(\mathbf{x}(t_i)]_j}{\partial x_k(t_i)}$$

That is, it is the Jacobian of $M(\mathbf{x})$ with respect to x.

We have

$$\delta\mathbf{x}(t_{i+1}) = \mathbf{L}_i\delta\mathbf{x}(t_i) + \mathrm{H.O.T.}$$

The Adjoint Model

The linear tangent model $\mathbf{L}_i$ is a matrix that transforms an initial perturbation at time t_i to the final perturbation at time t_{i+1}.

$$\delta\mathbf{x}(t_{i+1}) = \mathbf{L}_i \delta\mathbf{x}(t_i) + \mathrm{H.O.T.}$$

The Adjoint Model

The linear tangent model $\mathbf{L}_i$ is a matrix that transforms an initial perturbation at time t_i to the final perturbation at time t_{i+1}.

$$\delta\mathbf{x}(t_{i+1}) = \mathbf{L}_i \delta\mathbf{x}(t_i) + \mathrm{H.O.T.}$$

The transpose of the linear tangent model is called the adjoint model.

$\star \qquad \star \qquad \star$

The Adjoint Model

The linear tangent model $\mathbf{L}_i$ is a matrix that transforms an initial perturbation at time t_i to the final perturbation at time t_{i+1}.

$$\delta\mathbf{x}(t_{i+1}) = \mathbf{L}_i \delta\mathbf{x}(t_i) + \text{H.O.T.}$$

The transpose of the linear tangent model is called the adjoint model.

$\star \qquad \star \qquad \star$

The linear tangent model $\mathbf{L}_i$ and the adjoint model $\mathbf{L}_i^T$ can be constructed by a systematic procedure.

The Adjoint Model

The linear tangent model $\mathbf{L}_i$ is a matrix that transforms an initial perturbation at time t_i to the final perturbation at time t_{i+1}.

$$\delta\mathbf{x}(t_{i+1}) = \mathbf{L}_i\delta\mathbf{x}(t_i) + \text{H.O.T.}$$

The transpose of the linear tangent model is called the adjoint model.

$\star \quad \star \quad \star$

The linear tangent model $\mathbf{L}_i$ and the adjoint model $\mathbf{L}_i^T$ can be constructed by a systematic procedure.

For a description of how to develop the computer codes, read Appendix B of Eugenia Kalnay's book.

If there are several steps in a time interval $t_0 - t_i$, the linear tangent model that advances a perturbation from t_0 to t_i is given by the product of the linear tangent model matrices.

If there are several steps in a time interval $t_0 - t_i$, the linear tangent model that advances a perturbation from t_0 to t_i is given by the product of the linear tangent model matrices.

Each one advance the solution over a single step.

$$\mathbf{L}(t_0, t_i) = \prod_{j=i-1}^{0} \mathbf{L}(t_j, t_{j+1}) = \prod_{j=i-1}^{0} \mathbf{L}_j = \mathbf{L}_{i-1}\mathbf{L}_{i-2} \cdots \mathbf{L}_1\mathbf{L}_0$$

(note the order of application, from right to left).

If there are several steps in a time interval $t_0 - t_i$, the linear tangent model that advances a perturbation from t_0 to t_i is given by the product of the linear tangent model matrices.

Each one advance the solution over a single step.

$$\mathbf{L}(t_0, t_i) = \prod_{j=i-1}^{0} \mathbf{L}(t_j, t_{j+1}) = \prod_{j=i-1}^{0} \mathbf{L}_j = \mathbf{L}_{i-1}\mathbf{L}_{i-2}\cdots\mathbf{L}_1\mathbf{L}_0$$

(note the order of application, from right to left).

Therefore, the adjoint model is given by

$$\mathbf{L}(t_i, t_0)^T = \prod_{j=0}^{i-1} \mathbf{L}(t_{j+1}, t_j)^T = \prod_{j=0}^{i-1} \mathbf{L}_j^T = \mathbf{L}_0^T\mathbf{L}_1^T\cdots\mathbf{L}_{i-2}^T\mathbf{L}_{i-1}^T$$

Note that the order of the terms is reversed.

If there are several steps in a time interval $t_0 - t_i$, the linear tangent model that advances a perturbation from t_0 to t_i is given by the product of the linear tangent model matrices.

Each one advance the solution over a single step.

$$\mathbf{L}(t_0, t_i) = \prod_{j=i-1}^{0} \mathbf{L}(t_j, t_{j+1}) = \prod_{j=i-1}^{0} \mathbf{L}_j = \mathbf{L}_{i-1}\mathbf{L}_{i-2} \cdots \mathbf{L}_1\mathbf{L}_0$$

(note the order of application, from right to left).

Therefore, the adjoint model is given by

$$\mathbf{L}(t_i, t_0)^T = \prod_{j=0}^{i-1} \mathbf{L}(t_{j+1}, t_j)^T = \prod_{j=0}^{i-1} \mathbf{L}_j^T = \mathbf{L}_0^T\mathbf{L}_1^T \cdots \mathbf{L}_{i-2}^T\mathbf{L}_{i-1}^T$$

Note that the order of the terms is reversed.

The adjoint model advances a perturbation backwards in time, from the final to the initial time.

Simple Case:

$$\mathbf{x}_2 = M_1(\mathbf{x}_1) = M_1(M_0(\mathbf{x}_0))$$

Simple Case:

$$\mathbf{x}_2 = M_1(\mathbf{x}_1) = M_1(M_0(\mathbf{x}_0))$$

Suppose $\mathbf{x}_0 \longrightarrow \mathbf{x}_0 + \delta\mathbf{x}_0$.

Simple Case:

$$\mathbf{x}_2 = M_1(\mathbf{x}_1) = M_1(M_0(\mathbf{x}_0))$$

Suppose $\mathbf{x}_0 \longrightarrow \mathbf{x}_0 + \delta\mathbf{x}_0$.

Then $\mathbf{x}_1 \longrightarrow \mathbf{x}_1 + \delta\mathbf{x}_1$ with

$$\mathbf{x}_1 + \delta\mathbf{x}_1 = M_0(\mathbf{x}_0 + \delta\mathbf{x}_0) = M_0(\mathbf{x}_0) + \mathbf{L}_0\delta\mathbf{x}_0$$

Simple Case:

$$\mathbf{x}_2 = M_1(\mathbf{x}_1) = M_1(M_0(\mathbf{x}_0))$$

Suppose $\mathbf{x}_0 \longrightarrow \mathbf{x}_0 + \delta\mathbf{x}_0$.

Then $\mathbf{x}_1 \longrightarrow \mathbf{x}_1 + \delta\mathbf{x}_1$ with

$$\mathbf{x}_1 + \delta\mathbf{x}_1 = M_0(\mathbf{x}_0 + \delta\mathbf{x}_0) = M_0(\mathbf{x}_0) + \mathbf{L}_0\delta\mathbf{x}_0$$

Now $\mathbf{x}_2 \longrightarrow \mathbf{x}_2 + \delta\mathbf{x}_2$ with

$$\begin{aligned}
\mathbf{x}_2 + \delta\mathbf{x}_2 &= M_1(\mathbf{x}_1 + \delta\mathbf{x}_1) \\
&= M_1(\mathbf{x}_1) + \mathbf{L}_1\delta\mathbf{x}_1 \\
&= M_1(M_0(\mathbf{x}_0)) + \mathbf{L}_1\mathbf{L}_0\delta\mathbf{x}_0 \\
&= \mathbf{x}_2 + \mathbf{L}_1\mathbf{L}_0\delta\mathbf{x}_0
\end{aligned}$$

Simple Case:

$$\mathbf{x}_2 = M_1(\mathbf{x}_1) = M_1(M_0(\mathbf{x}_0))$$

Suppose $\mathbf{x}_0 \longrightarrow \mathbf{x}_0 + \delta\mathbf{x}_0$.

Then $\mathbf{x}_1 \longrightarrow \mathbf{x}_1 + \delta\mathbf{x}_1$ with

$$\mathbf{x}_1 + \delta\mathbf{x}_1 = M_0(\mathbf{x}_0 + \delta\mathbf{x}_0) = M_0(\mathbf{x}_0) + \mathbf{L}_0\delta\mathbf{x}_0$$

Now $\mathbf{x}_2 \longrightarrow \mathbf{x}_2 + \delta\mathbf{x}_2$ with

$$\begin{aligned}
\mathbf{x}_2 + \delta\mathbf{x}_2 &= M_1(\mathbf{x}_1 + \delta\mathbf{x}_1) \\
&= M_1(\mathbf{x}_1) + \mathbf{L}_1\delta\mathbf{x}_1 \\
&= M_1(M_0(\mathbf{x}_0)) + \mathbf{L}_1\mathbf{L}_0\delta\mathbf{x}_0 \\
&= \mathbf{x}_2 + \mathbf{L}_1\mathbf{L}_0\delta\mathbf{x}_0
\end{aligned}$$

Therefore,

$$\delta\mathbf{x}_2 = \mathbf{L}_1\mathbf{L}_0\delta\mathbf{x}_0$$

Simple Case:

$$\mathbf{x}_2 = M_1(\mathbf{x}_1) = M_1(M_0(\mathbf{x}_0))$$

Suppose $\mathbf{x}_0 \longrightarrow \mathbf{x}_0 + \delta\mathbf{x}_0$.

Then $\mathbf{x}_1 \longrightarrow \mathbf{x}_1 + \delta\mathbf{x}_1$ with

$$\mathbf{x}_1 + \delta\mathbf{x}_1 = M_0(\mathbf{x}_0 + \delta\mathbf{x}_0) = M_0(\mathbf{x}_0) + \mathbf{L}_0\delta\mathbf{x}_0$$

Now $\mathbf{x}_2 \longrightarrow \mathbf{x}_2 + \delta\mathbf{x}_2$ with

$$\begin{aligned}
\mathbf{x}_2 + \delta\mathbf{x}_2 &= M_1(\mathbf{x}_1 + \delta\mathbf{x}_1) \\
&= M_1(\mathbf{x}_1) + \mathbf{L}_1\delta\mathbf{x}_1 \\
&= M_1(M_0(\mathbf{x}_0)) + \mathbf{L}_1\mathbf{L}_0\delta\mathbf{x}_0 \\
&= \mathbf{x}_2 + \mathbf{L}_1\mathbf{L}_0\delta\mathbf{x}_0
\end{aligned}$$

Therefore,

$$\delta\mathbf{x}_2 = \mathbf{L}_1\mathbf{L}_0\delta\mathbf{x}_0$$

The adjoint of $\mathbf{L}_1\mathbf{L}_0$ is $\mathbf{L}_0^T\mathbf{L}_1^T$

The reversal of the order of the terms corresponds to a reversal of time: the operations are preformed backwards.

4D-Var (Kalnay, §5.6.3)

Four-dimensional variational assimilation (4D-Var) is an extension of 3D-Var to allow for observations distributed within a time interval (t_0, t_n).

4D-Var (Kalnay, §5.6.3)

Four-dimensional variational assimilation (4D-Var) is an extension of 3D-Var to allow for observations distributed within a time interval (t_0, t_n).

The cost function includes a term measuring the distance to the background at the beginning of the interval.

4D-Var (Kalnay, §5.6.3)

Four-dimensional variational assimilation (4D-Var) is an extension of 3D-Var to allow for observations distributed within a time interval (t_0, t_n).

The cost function includes a term measuring the distance to the background at the beginning of the interval.

It also includes a summation over time of the cost function for each observational increment computed with respect to the model integrated to the time of the observation.

4D-Var (Kalnay, §5.6.3)

Four-dimensional variational assimilation (4D-Var) is an extension of 3D-Var to allow for observations distributed within a time interval (t_0, t_n).

The cost function includes a term measuring the distance to the background at the beginning of the interval.

It also includes a summation over time of the cost function for each observational increment computed with respect to the model integrated to the time of the observation.

$$
\begin{aligned}
J[\mathbf{x}(t_0)] = {} & \frac{1}{2}[\mathbf{x}(\mathrm{t}_0) - \mathbf{x}^b(t_0)]^T \mathbf{B}_0^{-1}[\mathbf{x}(\mathrm{t}_0) - \mathbf{x}^b(t_0)] \\
& + \frac{1}{2}\sum_{i=0}^{N}\left[H(\mathbf{x}_i) - \mathbf{y}_i^o\right]^T \mathbf{R}_i^{-1}\left[H(\mathbf{x}_i) - \mathbf{y}_i^o\right]
\end{aligned}
$$

4D-Var (Kalnay, §5.6.3)

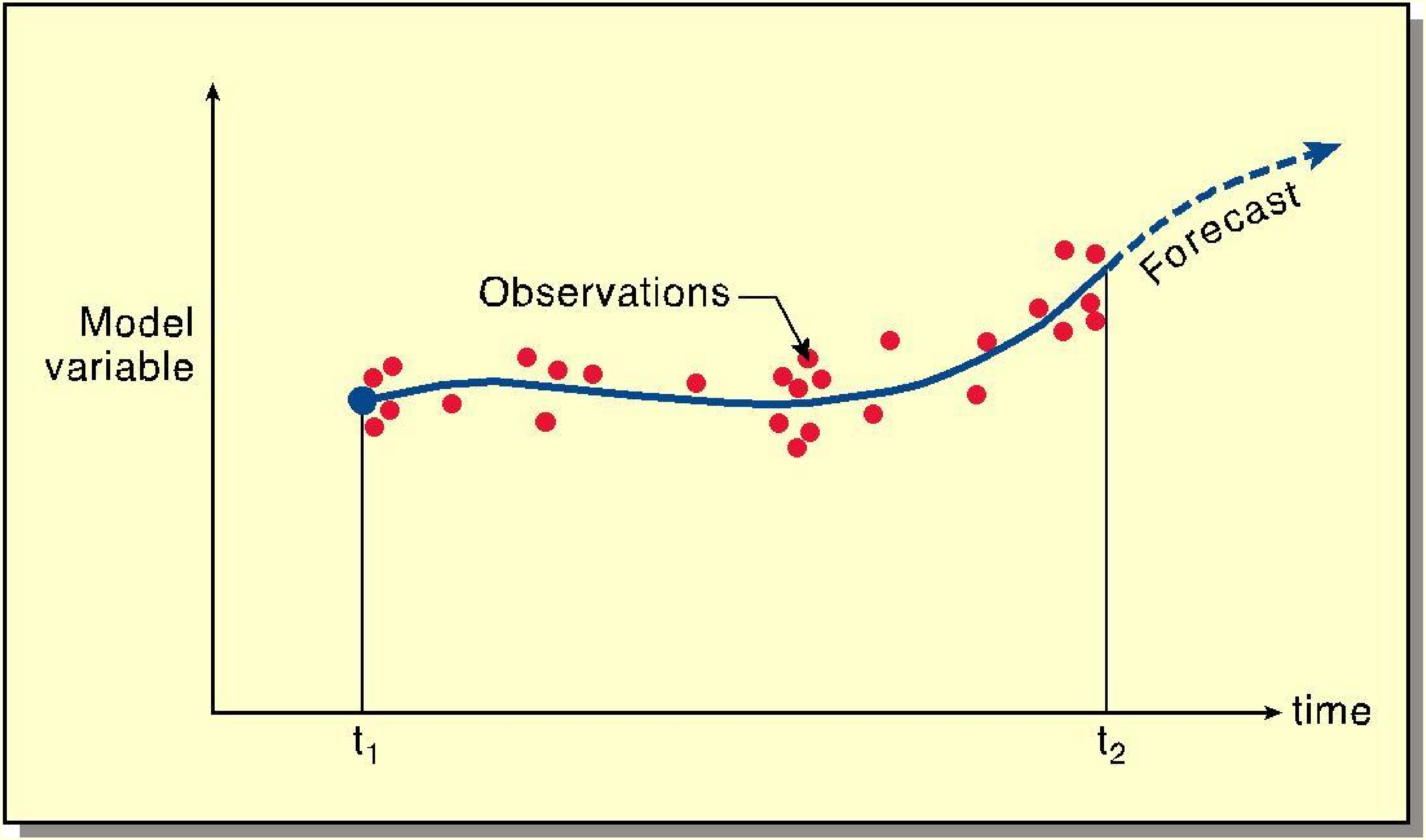
Four-dimensional variational assimilation (4D-Var) is an extension of 3D-Var to allow for observations distributed within a time interval (t_0, t_n).

The cost function includes a term measuring the distance to the background at the beginning of the interval.

It also includes a summation over time of the cost function for each observational increment computed with respect to the model integrated to the time of the observation.

$$
\begin{aligned}
J[\mathbf{x}(t_0)] &= \frac{1}{2}[\mathbf{x}(\mathrm{t}_0) - \mathbf{x}^b(t_0)]^T \mathbf{B}_0^{-1} [\mathbf{x}(\mathrm{t}_0) - \mathbf{x}^b(t_0)] \\
&\quad + \frac{1}{2}\sum_{i=0}^{N} \left[H(\mathbf{x}_i) - \mathbf{y}_i^o\right]^T \mathbf{R}_i^{-1} \left[H(\mathbf{x}_i) - \mathbf{y}_i^o\right]
\end{aligned}
$$

The control variable is the *initial state* of the model $\mathbf{x}\,(t_0)$.



Schematic diagram of four dimensional variational assimilation.

The analysis at the end of the interval is given by the model integration from the solution

$$\mathbf{x}(t_n) = M_{0-n}\left[\mathbf{x}(t_0)\right] = M_{n-1}\left[M_{n-2}\cdots\left[M_1\left[M_0\left[\mathbf{x}(t_0)\right]\right]\cdots\right]\right]$$

The analysis at the end of the interval is given by the model integration from the solution

$$\mathbf{x}(t_n) = M_{0-n}\left[\mathbf{x}(t_0)\right] = M_{n-1}\left[M_{n-2}\cdots\left[M_1\left[M_0\left[\mathbf{x}(t_0)\right]\right]\cdots\right]\right]$$

Thus, the model is used as a strong constraint. That is, the analysis solution has to satisfy the model equations.

The analysis at the end of the interval is given by the model integration from the solution

$$\mathbf{x}(t_n) = M_{0-n}\left[\mathbf{x}(t_0)\right] = M_{n-1}\left[M_{n-2}\cdots\left[M_1\left[M_0\left[\mathbf{x}(t_0)\right]\right]\cdots\right]\right]$$

Thus, the model is used as a strong constraint. That is, the analysis solution has to satisfy the model equations.

4D-Var thus seeks an initial condition such that the forecast best fits the observations within the assimilation interval.

$\star \qquad \star \qquad \star$

The analysis at the end of the interval is given by the model integration from the solution

$$\mathbf{x}(t_n) = M_{0-n}\left[\mathbf{x}(t_0)\right] = M_{n-1}\left[M_{n-2}\cdots\left[M_1\left[M_0\left[\mathbf{x}(t_0)\right]\right]\cdots\right]\right]$$

Thus, the model is used as a strong constraint. That is, the analysis solution has to satisfy the model equations.

4D-Var thus seeks an initial condition such that the forecast best fits the observations within the assimilation interval.

$\star \quad \star \quad \star$

The fact that the 4D-Var method assumes a perfect model is a disadvantage.

For example, it will give the same weight to older observations as to newer observations.

The analysis at the end of the interval is given by the model integration from the solution

$$\mathbf{x}(t_n) = M_{0-n}\left[\mathbf{x}(t_0)\right] = M_{n-1}\left[M_{n-2}\cdots\left[M_1\left[M_0\left[\mathbf{x}(t_0)\right]\right]\cdots\right]\right]$$

Thus, the model is used as a strong constraint. That is, the analysis solution has to satisfy the model equations.

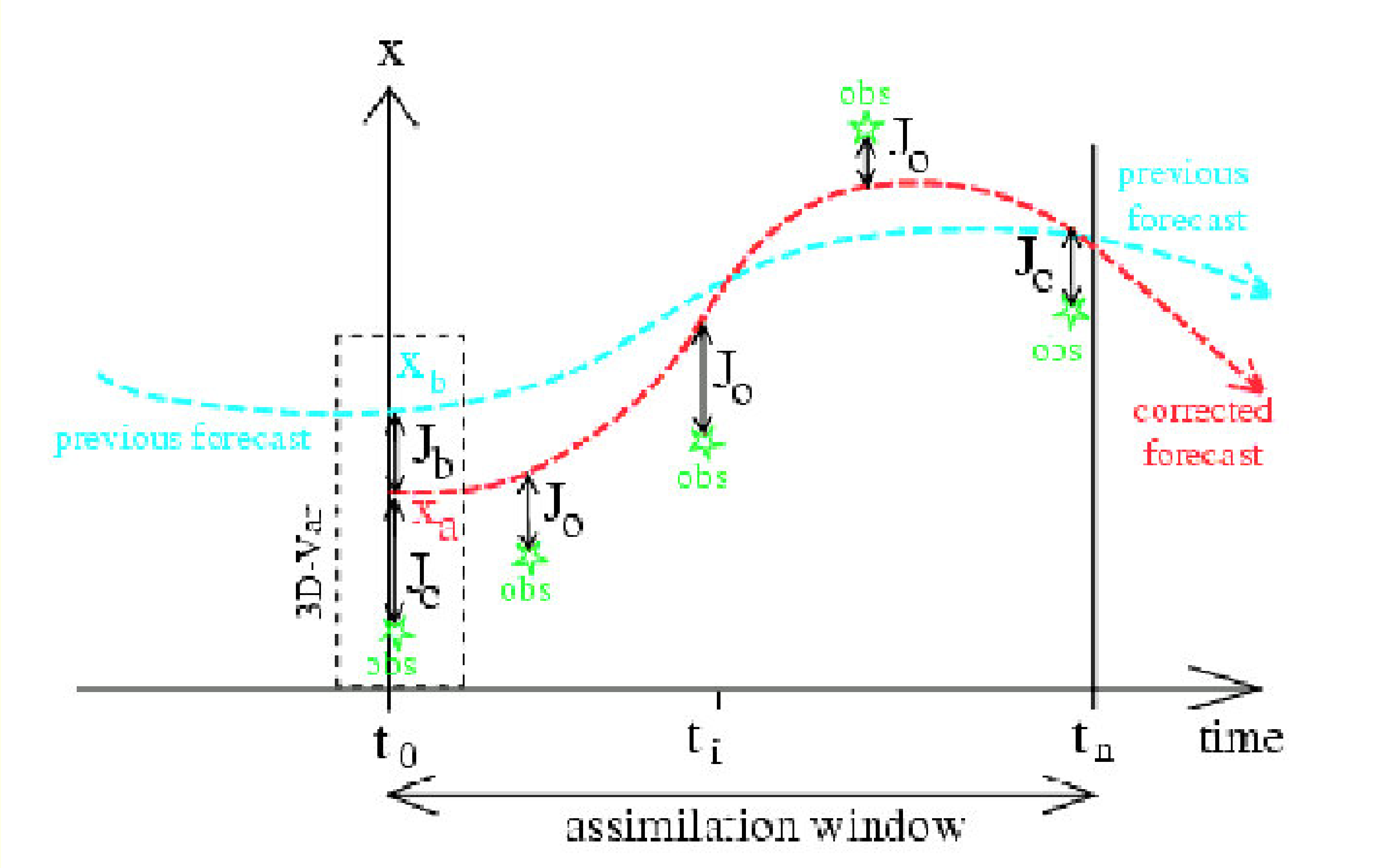
4D-Var thus seeks an initial condition such that the forecast best fits the observations within the assimilation interval.

$\star \qquad \star \qquad \star$

The fact that the 4D-Var method assumes a perfect model is a disadvantage.

For example, it will give the same weight to older observations as to newer observations.

Methods of correcting for a constant model error have been proposed (see references in Kalnay).



Schematic diagram of four dimensional variational assimilation

Let us consider the variation in the cost function when the control variable $\mathbf{x}(t_0)$ is changed by a small amount $\delta\mathbf{x}(t_0)$.

Let us consider the variation in the cost function when the control variable $\mathbf{x}(t_0)$ is changed by a small amount $\delta\mathbf{x}(t_0)$.

It is given by

$$\delta J = J[\mathbf{x}(t_0) + \delta\mathbf{x}(t_0)] - J[\mathbf{x}(t_0)] \approx \left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]^T \cdot \delta\mathbf{x}(t_0)$$

Let us consider the variation in the cost function when the control variable $\mathbf{x}(t_0)$ is changed by a small amount $\delta\mathbf{x}(t_0)$.

It is given by

$$\delta J = J[\mathbf{x}(t_0) + \delta\mathbf{x}(t_0)] - J[\mathbf{x}(t_0)] \approx \left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]^T \cdot \delta\mathbf{x}(t_0)$$

Here the gradient of the cost function

$$\nabla_{\mathrm{x}(t_0)} J = \left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]$$

is a column vector (of course, δJ is a scalar).

Let us consider the variation in the cost function when the control variable $\mathbf{x}(t_0)$ is changed by a small amount $\delta\mathbf{x}(t_0)$.

It is given by

$$\delta J = J[\mathbf{x}(t_0) + \delta\mathbf{x}(t_0)] - J[\mathbf{x}(t_0)] \approx \left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]^T \cdot \delta\mathbf{x}(t_0)$$

Here the gradient of the cost function

$$\nabla_{\mathrm{x}(t_0)} J = \left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]$$

is a column vector (of course, δJ is a scalar).

Its j-th component is

$$\left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]_j = \frac{\partial J}{\partial x_j(t_0)}$$

Let us consider the variation in the cost function when the control variable $\mathbf{x}(t_0)$ is changed by a small amount $\delta\mathbf{x}(t_0)$.

It is given by

$$\delta J = J[\mathbf{x}(t_0) + \delta\mathbf{x}(t_0)] - J[\mathbf{x}(t_0)] \approx \left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]^T \cdot \delta\mathbf{x}(t_0)$$

Here the gradient of the cost function

$$\nabla_{\mathrm{x}(t_0)} J = \left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]$$

is a column vector (of course, δJ is a scalar).

Its j-th component is

$$\left[\frac{\partial J}{\partial \mathbf{x}(t_0)}\right]_j = \frac{\partial J}{\partial x_j(t_0)}$$

We need this because iterative minimization schemes require the estimation of the gradient of the cost function.

In the simplest scheme, the steepest descent method, the change in the control variable after each iteration is chosen to be opposite to the gradient

$$\delta \mathbf{x}(t_0) = -a \nabla_{\mathrm{x}(t_0)} J = -a \, \partial J / \partial \mathbf{x}(t_0) \, .$$

where a is chosen empirically.

In the simplest scheme, the steepest descent method, the change in the control variable after each iteration is chosen to be opposite to the gradient

$$\delta\mathbf{x}(t_0) = -a\nabla_{\mathrm{x}(t_0)} J = -a\,\partial J/\partial\mathbf{x}(t_0)\,.$$

where a is chosen empirically.

More efficient methods, such as the conjugate gradient or quasi-Newton method, also require the use of the gradient.

In the simplest scheme, the steepest descent method, the change in the control variable after each iteration is chosen to be opposite to the gradient

$$\delta\mathbf{x}(t_0) = -a\nabla_{\mathrm{x}(t_0)}J = -a\,\partial J/\partial\mathbf{x}(t_0)\,.$$

where a is chosen empirically.

More efficient methods, such as the conjugate gradient or quasi-Newton method, also require the use of the gradient.

Thus, in order to solve this minimization problem efficiently, we need to be able to compute the gradient of J with respect to the elements of the control variable.

$\star \quad \star \quad \star$

Lemma I:

Given a symmetric matrix A and a functional $J = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x}$**, the gradient is given by**

$$\frac{\partial J}{\partial \mathbf{x}} = \mathbf{A}\mathbf{x}\,.$$

(we proved this already).

Lemma I:

Given a symmetric matrix A and a functional $J = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x}$, the gradient is given by

$$\frac{\partial J}{\partial \mathbf{x}} = \mathbf{A}\mathbf{x}\,.$$

(we proved this already).

Lemma II:

If $J = \mathbf{y}^T\mathbf{A}\mathbf{y}$, and $\mathbf{y} = \mathbf{y}(\mathbf{x})$, then

$$\frac{\partial J}{\partial \mathbf{x}} = \left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]^T \frac{\partial J}{\partial \mathbf{y}} = \left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]^T \mathbf{A}\mathbf{y}$$

where $[\partial\mathbf{y}/\partial\mathbf{x}]_{k,l} = \partial y_k/\partial x_l$ is a matrix.

Proof of Lemma II:

Consider $J = J(y_1, \ldots, y_n)$ **where** $y_i = y_i(x_1, \ldots, x_n)$**.**

Proof of Lemma II:

Consider $J = J(y_1, \ldots, y_n)$ **where** $y_i = y_i(x_1, \ldots, x_n)$**.**

Then

$$\frac{\partial J}{\partial x_k} = \sum_j \frac{\partial y_j}{\partial x_k}\frac{\partial J}{\partial y_j}$$

Proof of Lemma II:

Consider $J = J(y_1, \ldots, y_n)$ **where** $y_i = y_i(x_1, \ldots, x_n)$**.**

Then

$$\frac{\partial J}{\partial x_k} = \sum_j \frac{\partial y_j}{\partial x_k} \frac{\partial J}{\partial y_j}$$

But we have

$$\left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]_{j,k} = \frac{\partial y_j}{\partial x_k} \qquad \textbf{Thus} \qquad \left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]^T_{k,j} = \frac{\partial y_j}{\partial x_k}$$

Proof of Lemma II:

Consider $J = J(y_1, \ldots, y_n)$ **where** $y_i = y_i(x_1, \ldots, x_n)$**.**

Then

$$\frac{\partial J}{\partial x_k} = \sum_j \frac{\partial y_j}{\partial x_k} \frac{\partial J}{\partial y_j}$$

But we have

$$\left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]_{j,k} = \frac{\partial y_j}{\partial x_k} \qquad \textbf{Thus} \qquad \left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]^T_{k,j} = \frac{\partial y_j}{\partial x_k}$$

Thus, in vector form, the result is

$$\frac{\partial J}{\partial \mathbf{x}} = \left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]^T \frac{\partial J}{\partial \mathbf{y}}$$

Proof of Lemma II:

Consider $J = J(y_1, \ldots, y_n)$ **where** $y_i = y_i(x_1, \ldots, x_n)$**.**

Then

$$\frac{\partial J}{\partial x_k} = \sum_j \frac{\partial y_j}{\partial x_k}\frac{\partial J}{\partial y_j}$$

But we have

$$\left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]_{j,k} = \frac{\partial y_j}{\partial x_k} \qquad \textbf{Thus} \qquad \left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]^T_{k,j} = \frac{\partial y_j}{\partial x_k}$$

Thus, in vector form, the result is

$$\frac{\partial J}{\partial \mathbf{x}} = \left[\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right]^T \frac{\partial J}{\partial \mathbf{y}}$$

Q.E.D.

Conclusion of the foregoing

$J = J_b + J_o$

We can write the cost function J as a sum of the background error term and the observation error term

$$J = J_b + J_o\,.$$

$J = J_b + J_o$

We can write the cost function J as a sum of the background error term and the observation error term

$$J = J_b + J_o\,.$$

First, we require the gradient, with respect to $\mathbf{x}(t_0)$, of the background component of the cost function

$$J_b = \frac{1}{2}[\mathbf{x}(t_0) - \mathbf{x}^b(t_0)]^T \mathbf{B}_0^{-1}[\mathbf{x}(t_0) - \mathbf{x}^b(t_0)]$$

$J = J_b + J_o$

We can write the cost function J as a sum of the background error term and the observation error term

$$J = J_b + J_o\,.$$

First, we require the gradient, with respect to $\mathbf{x}\,(t_0)$, of the background component of the cost function

$$J_b = \frac{1}{2}[\mathbf{x}(t_0) - \mathbf{x}^b(t_0)]^T \mathbf{B}_0^{-1}[\mathbf{x}(t_0) - \mathbf{x}^b(t_0)]$$

This is given by

$$\frac{\partial J_b}{\partial \mathbf{x}(t_0)} = \mathbf{B}_0^{-1}[\mathbf{x}(t_0) - \mathbf{x}^b(t_0)]$$

$J = J_b + J_o$

We can write the cost function J as a sum of the background error term and the observation error term

$$J = J_b + J_o \, .$$

First, we require the gradient, with respect to $\mathbf{x}(t_0)$, of the background component of the cost function

$$J_b = \frac{1}{2}[\mathbf{x}(t_0) - \mathbf{x}^b(t_0)]^T \mathbf{B}_0^{-1}[\mathbf{x}(t_0) - \mathbf{x}^b(t_0)]$$

This is given by

$$\frac{\partial J_b}{\partial \mathbf{x}(t_0)} = \mathbf{B}_0^{-1}[\mathbf{x}(t_0) - \mathbf{x}^b(t_0)]$$

We are half-way there (but it is the easy half).

The gradient of the term J_o is more complicated.

The gradient of the second term,

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

is more complicated because $\mathbf{x}_i = M_{0-i}[\mathbf{x}(t_0)]$ **depends on** $\mathbf{x}(t_0)$ **through the model.**

The gradient of the second term,

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

is more complicated because $\mathbf{x}_i = M_{0-i}[\mathbf{x}(t_0)]$ depends on $\mathbf{x}(t_0)$ through the model.

If we perturb the initial state, then $\delta\mathbf{x}_i = \mathbf{L}(t_0, t_i)\delta\mathbf{x}_0$.

The gradient of the second term,

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

is more complicated because $\mathbf{x}_i = M_{0-i}[\mathbf{x}(t_0)]$ depends on $\mathbf{x}(t_0)$ through the model.

If we perturb the initial state, then $\delta\mathbf{x}_i = \mathbf{L}(t_0, t_i)\delta\mathbf{x}_0$.

Therefore,

$$\frac{\partial\big(H(\mathbf{x}_i) - \mathbf{y}_i^o\big)}{\partial \mathbf{x}_0} = \frac{\partial H}{\partial \mathbf{x}_i}\frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_o} = \mathbf{H}_i \mathbf{L}(t_0, t_i)\,.$$

The gradient of the second term,

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

is more complicated because $\mathbf{x}_i = M_{0-i}[\mathbf{x}(t_0)]$ depends on $\mathbf{x}(t_0)$ through the model.

If we perturb the initial state, then $\delta\mathbf{x}_i = \mathbf{L}(t_0, t_i)\delta\mathbf{x}_0$.

Therefore,

$$\frac{\partial\big(H(\mathbf{x}_i) - \mathbf{y}_i^o\big)}{\partial\mathbf{x}_0} = \frac{\partial H}{\partial\mathbf{x}_i}\frac{\partial\mathbf{x}_i}{\partial\mathbf{x}_o} = \mathbf{H}_i\mathbf{L}(t_0, t_i)\,.$$

The matrices $\mathbf{H}_i$ and $\mathbf{L}(t_0, t_i)$ are the linearized Jacobians:

$$\mathbf{H}_i = \frac{\partial H}{\partial\mathbf{x}_i} \qquad \textbf{and} \qquad \mathbf{L}(t_0, t_i) = \frac{\partial M}{\partial\mathbf{x}_o}$$

The gradient of the second term,

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T\mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

is more complicated because $\mathbf{x}_i = M_{0-i}[\mathbf{x}(t_0)]$ depends on $\mathbf{x}(t_0)$ through the model.

If we perturb the initial state, then $\delta\mathbf{x}_i = \mathbf{L}(t_0, t_i)\delta\mathbf{x}_0$.

Therefore,

$$\frac{\partial\big(H(\mathbf{x}_i) - \mathbf{y}_i^o\big)}{\partial\mathbf{x}_0} = \frac{\partial H}{\partial\mathbf{x}_i}\frac{\partial\mathbf{x}_i}{\partial\mathbf{x}_o} = \mathbf{H}_i\mathbf{L}(t_0, t_i)\,.$$

The matrices $\mathbf{H}_i$ and $\mathbf{L}(t_0, t_i)$ are the linearized Jacobians:

$$\mathbf{H}_i = \frac{\partial H}{\partial\mathbf{x}_i} \qquad \textbf{and} \qquad \mathbf{L}(t_0, t_i) = \frac{\partial M}{\partial\mathbf{x}_o}$$

Expanding the linear tangent model operator step by step,

$$\mathbf{H}_i\mathbf{L}(t_0, t_i) = \mathbf{H}_i\prod_{j=i-1}^{0}\mathbf{L}(t_j, t_{j+1}) = \mathbf{H}_i\left[\mathbf{L}_{i-1}\mathbf{L}_{i-2}\cdots\mathbf{L}_1\mathbf{L}_0\right]\,.$$

Recall that

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

and its gradient w.r.t. $\mathbf{x}_0$ is

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = \left[\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0}\right]^T \frac{\partial J_o}{\partial H(\mathbf{x}_i)}$$

Recall that

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

and its gradient w.r.t. $\mathbf{x}_0$ is

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = \left[\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0}\right]^T \frac{\partial J_o}{\partial H(\mathbf{x}_i)}$$

But we have shown that

$$\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0} = \mathbf{H}_i \mathbf{L}(t_0, t_i) \qquad \textbf{so that} \qquad \left[\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0}\right]^T = \mathbf{L}^T(t_0, t_i)\mathbf{H}_i^T$$

Recall that

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

and its gradient w.r.t. $\mathbf{x}_0$ is

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = \left[\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0}\right]^T \frac{\partial J_o}{\partial H(\mathbf{x}_i)}$$

But we have shown that

$$\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0} = \mathbf{H}_i \mathbf{L}(t_0, t_i) \qquad \textbf{so that} \qquad \left[\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0}\right]^T = \mathbf{L}^T(t_0, t_i)\mathbf{H}_i^T$$

Therefore, the gradient of the observation cost function is

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = \sum_{i=0}^{N} \mathbf{L}(t_i, t_0)^T \mathbf{H}_i^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

Recall that

$$J_o = \frac{1}{2}\sum_{i=0}^{N}[H(\mathbf{x}_i) - \mathbf{y}_i^o]^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

and its gradient w.r.t. $\mathbf{x}_0$ is

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = \left[\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0}\right]^T \frac{\partial J_o}{\partial H(\mathbf{x}_i)}$$

But we have shown that

$$\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0} = \mathbf{H}_i \mathbf{L}(t_0, t_i) \qquad \textbf{so that} \qquad \left[\frac{\partial H(\mathbf{x}_i)}{\partial \mathbf{x}_0}\right]^T = \mathbf{L}^T(t_0, t_i)\mathbf{H}_i^T$$

Therefore, the gradient of the observation cost function is

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = \sum_{i=0}^{N} \mathbf{L}(t_i, t_0)^T \mathbf{H}_i^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o]$$

Defining the innovation $\mathbf{d}_i = [\mathbf{y}_i^o - H(\mathbf{x}_i)]$, this is

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots \ \mathbf{L}_{i-1}^T\right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

Again,

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots \ \mathbf{L}_{i-1}^T\right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

Again,

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots \ \mathbf{L}_{i-1}^T \right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

Every iteration of the 4D-Var minimization requires the computation of the gradient. It involves

Again,

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots\ \mathbf{L}_{i-1}^T\right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

Every iteration of the 4D-Var minimization requires the computation of the gradient. It involves

- **Computing the increments** $\mathbf{d}_i = -[H(\mathbf{x}_i) - \mathbf{y}_i^o]$ **at the observation times** t_i **during a forward integration**

Again,

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots \ \mathbf{L}_{i-1}^T \right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

Every iteration of the 4D-Var minimization requires the computation of the gradient. It involves

- **Computing the increments** $\mathbf{d}_i = -[H(\mathbf{x}_i) - \mathbf{y}_i^o]$ **at the observation times** t_i **during a forward integration**
- **Multiplying them by** $\mathbf{H}_i^T \mathbf{R}_i^{-1}$

Again,

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots \ \mathbf{L}_{i-1}^T\right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

Every iteration of the 4D-Var minimization requires the computation of the gradient. It involves

- **Computing the increments $\mathbf{d}_i = -[H(\mathbf{x}_i) - \mathbf{y}_i^o]$ at the observation times t_i during a forward integration**
- **Multiplying them by $\mathbf{H}_i^T \mathbf{R}_i^{-1}$**
- **Integrating these weighted increments backward to the initial time using the adjoint model.**

Since parts of the backward adjoint integration are common to several time intervals, the summation

$$\sum_{i=0}^{N} \mathbf{L}(t_i, t_0)^T \mathbf{H}_i^T \mathbf{R}_i^{-1} \left[H(\mathbf{x}_i) - \mathbf{y}_i^o \right]$$

for $\partial J_o / \partial \mathbf{x}_0$ can be arranged more conveniently.

Since parts of the backward adjoint integration are common to several time intervals, the summation

$$\sum_{i=0}^{N} \mathbf{L}(t_i, t_0)^T \mathbf{H}_i^T \mathbf{R}_i^{-1} \left[H(\mathbf{x}_i) - \mathbf{y}_i^o \right]$$

for $\partial J_o / \partial \mathbf{x}_0$ can be arranged more conveniently.

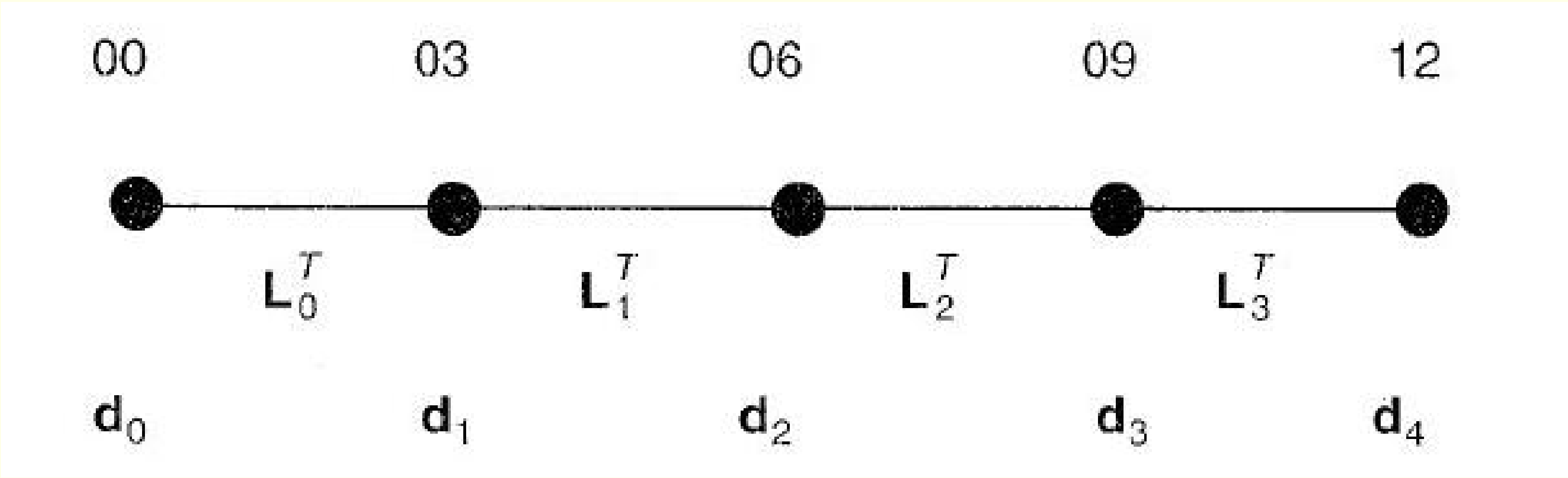
For example, suppose the interval of assimilation is from 00 Z to 12 Z, with observations every 3 hours.

Since parts of the backward adjoint integration are common to several time intervals, the summation

$$\sum_{i=0}^{N} \mathbf{L}(t_i, t_0)^T \mathbf{H}_i^T \mathbf{R}_i^{-1} \left[H(\mathbf{x}_i) - \mathbf{y}_i^o \right]$$

for $\partial J_o / \partial \mathrm{x}_0$ can be arranged more conveniently.

For example, suppose the interval of assimilation is from 00 Z to 12 Z, with observations every 3 hours.



Schematic of the computation of the gradient of the observational cost function for a period of 12 h, with observations every 3 hours.

We compute, during the forward integration, the weighted negative observation increments

$$\overline{\mathbf{d}}_i = \mathbf{H}_i^T \mathbf{R}_i^{-1} [H(\mathbf{x}_i) - \mathbf{y}_i^o] = -\mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i \, .$$

We compute, during the forward integration, the weighted negative observation increments

$$\overline{\mathbf{d}}_i = \mathbf{H}_i^T \mathbf{R}_i^{-1} [H(\mathbf{x}_i) - \mathbf{y}_i^o] = -\mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i \,.$$

The adjoint model $\mathbf{L}^T(t_i, t_{i-1}) = \mathbf{L}_{i-1}^T$ applied to a vector $\overline{\mathbf{d}}_\mathbf{i}$ "converts" it from time t_i to time t_{i-1}.

We compute, during the forward integration, the weighted negative observation increments

$$\overline{\mathbf{d}}_i = \mathbf{H}_i^T \mathbf{R}_i^{-1} [H(\mathbf{x}_i) - \mathbf{y}_i^o] = -\mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i \,.$$

The adjoint model $\mathbf{L}^T(t_i, t_{i-1}) = \mathbf{L}_{i-1}^T$ applied to a vector $\overline{\mathbf{d}}_\mathbf{i}$ "converts" it from time t_i to time t_{i-1}.

Recall the equation

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots \; \mathbf{L}_{i-1}^T \right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

We compute, during the forward integration, the weighted negative observation increments

$$\overline{\mathbf{d}}_i = \mathbf{H}_i^T \mathbf{R}_i^{-1} [H(\mathbf{x}_i) - \mathbf{y}_i^o] = -\mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i \,.$$

The adjoint model $\mathbf{L}^T(t_i, t_{i-1}) = \mathbf{L}_{i-1}^T$ applied to a vector $\overline{\mathbf{d}}_\mathbf{i}$ "converts" it from time t_i to time t_{i-1}.

Recall the equation

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots \ \mathbf{L}_{i-1}^T \right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

This can be written

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = \Big[\overline{\mathbf{d}}_0 + \mathbf{L}_0^T \overline{\mathbf{d}}_1 + \mathbf{L}_0^T \mathbf{L}_1^T \overline{\mathbf{d}}_2 + \\ \mathbf{L}_0 \mathbf{L}_1^T \mathbf{L}_2^T \overline{\mathbf{d}}_3 + \mathbf{L}_0 \mathbf{L}_1 \mathbf{L}_2^T \mathbf{L}_3^T \overline{\mathbf{d}}_4 \Big]$$

We compute, during the forward integration, the weighted negative observation increments

$$\overline{\mathbf{d}}_i = \mathbf{H}_i^T \mathbf{R}_i^{-1}[H(\mathbf{x}_i) - \mathbf{y}_i^o] = -\mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i \, .$$

The adjoint model $\mathbf{L}^T(t_i, t_{i-1}) = \mathbf{L}_{i-1}^T$ applied to a vector $\overline{\mathbf{d}}_\mathbf{i}$ "converts" it from time t_i to time t_{i-1}.

Recall the equation

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = -\sum_{i=0}^{N} \left[\mathbf{L}_0^T \mathbf{L}_1^T \cdots \; \mathbf{L}_{i-1}^T\right] \mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$$

This can be written

$$\frac{\partial J_o}{\partial \mathbf{x}_0} = \Big[\overline{\mathbf{d}}_0 + \mathbf{L}_0^T \overline{\mathbf{d}}_1 + \mathbf{L}_0^T \mathbf{L}_1^T \overline{\mathbf{d}}_2 + \\ \mathbf{L}_0 \mathbf{L}_1^T \mathbf{L}_2^T \overline{\mathbf{d}}_3 + \mathbf{L}_0 \mathbf{L}_1 \mathbf{L}_2^T \mathbf{L}_3^T \overline{\mathbf{d}}_4\Big]$$

Thus, we can write the gradient of J as

$$\frac{\partial J_o}{\partial \mathbf{x}_o} = \overline{\mathbf{d}}_o + \mathbf{L}_0^T \left\{\overline{\mathbf{d}}_1 + \mathbf{L}_1^T \left[\overline{\mathbf{d}}_2 + \mathbf{L}_2^T \left(\overline{\mathbf{d}}_3 + \mathbf{L}_3^T \overline{\mathbf{d}}_4\right)\right]\right\}$$

The minimization algorithm is now applied, modifying the control variable $\mathbf{x}(t_0)$ at each stage.

The minimization algorithm is now applied, modifying the control variable $\mathbf{x}(t_0)$ at each stage.

After this change, a new forward integration and new observational increments are computed and the process is repeated until convergence is satisfactory.

The minimization algorithm is now applied, modifying the control variable $\mathbf{x}(t_0)$ at each stage.

After this change, a new forward integration and new observational increments are computed and the process is repeated until convergence is satisfactory.

- **Integrate the full model forward, computing and storing the increments $\mathbf{d}_i$ at the observation times t_i.**

The minimization algorithm is now applied, modifying the control variable $\mathbf{x}(t_0)$ at each stage.

After this change, a new forward integration and new observational increments are computed and the process is repeated until convergence is satisfactory.

- Integrate the full model forward, computing and storing the increments $\mathbf{d}_i$ at the observation times t_i.
- Integrate the adjoint model backwards, accumulating the terms $\overline{\mathbf{d}}_i = -\mathbf{H}_i^T\mathbf{R}_i^{-1}\mathbf{d}_i$, using the adjoint model.

The minimization algorithm is now applied, modifying the control variable $\mathbf{x}(t_0)$ at each stage.

After this change, a new forward integration and new observational increments are computed and the process is repeated until convergence is satisfactory.

- **Integrate the full model forward, computing and storing the increments $\mathbf{d}_i$ at the observation times t_i.**
- **Integrate the adjoint model backwards, accumulating the terms $\overline{\mathbf{d}}_i = -\mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{d}_i$, using the adjoint model.**
- **Iterate these forward-backward cycles until convergence.**

Reduced Inner Loops

4D-Var can also be written in an incremental form.

Reduced Inner Loops

4D-Var can also be written in an incremental form.

We define the cost function as

$$
\begin{aligned}
J(\delta\mathbf{x}_0) &= \frac{1}{2}(\delta\mathbf{x}_0)^T\mathbf{B}_0^{-1}(\delta\mathbf{x}_0) \\
&+ \frac{1}{2}\sum_{i=0}^{N}\left[\mathbf{H}_i\mathbf{L}(t_0,t_i)\delta\mathbf{x}_0 - \mathbf{d}_i^o\right]^T\mathbf{R}_i^{-1}\left[\mathbf{H}_i\mathbf{L}(t_0,t_i)\delta\mathbf{x}_0 - \mathbf{d}_i^o\right].
\end{aligned}
$$

Reduced Inner Loops

4D-Var can also be written in an incremental form.

We define the cost function as

$$
\begin{aligned}
J(\delta\mathbf{x}_0) &= \frac{1}{2}(\delta\mathbf{x}_0)^T\mathbf{B}_0^{-1}(\delta\mathbf{x}_0) \\
&\quad + \frac{1}{2}\sum_{i=0}^{N}\left[\mathbf{H}_i\mathbf{L}(t_0,t_i)\delta\mathbf{x}_0 - \mathbf{d}_i^o\right]^T\mathbf{R}_i^{-1}\left[\mathbf{H}_i\mathbf{L}(t_0,t_i)\delta\mathbf{x}_0 - \mathbf{d}_i^o\right].
\end{aligned}
$$

With the incremental formulation, we introduce a *"simplification operator"* S.

This converts the variables to a lower dimensional space than that of the original model variables x:

$$\delta\mathbf{w} = \mathbf{S}\delta\mathbf{x}$$

Reduced Inner Loops

4D-Var can also be written in an incremental form.

We define the cost function as

$$
\begin{aligned}
J(\delta\mathbf{x}_0) = {} & \frac{1}{2}(\delta\mathbf{x}_0)^T\mathbf{B}_0^{-1}(\delta\mathbf{x}_0) \\
& + \frac{1}{2}\sum_{i=0}^{N}\left[\mathbf{H}_i\mathbf{L}(t_0,t_i)\delta\mathbf{x}_0 - \mathbf{d}_i^o\right]^T\mathbf{R}_i^{-1}\left[\mathbf{H}_i\mathbf{L}(t_0,t_i)\delta\mathbf{x}_0 - \mathbf{d}_i^o\right].
\end{aligned}
$$

With the incremental formulation, we introduce a *"simplification operator"* S.

This converts the variables to a lower dimensional space than that of the original model variables x:

$$\delta\mathbf{w} = \mathbf{S}\delta\mathbf{x}$$

Typically, S is a projection to a lower dimensional subspace of the total model space.

A number of iterations are now executed in the reduced space. These are called the *"inner loops"*.

A number of iterations are now executed in the reduced space. These are called the *"inner loops"*.

Normally, the inverse of S doesn't exist: If we project to a lower-dimensional space, we cannot transform back unambiguously; information is lost.

A number of iterations are now executed in the reduced space. These are called the *"inner loops"*.

Normally, the inverse of S doesn't exist: If we project to a lower-dimensional space, we cannot transform back unambiguously; information is lost.

To return to the full space, we have to use a generalized inverse $\mathbf{S}^{-I} = [\mathbf{S}\mathbf{S}^T]^{-1}\mathbf{S}^T$.

A number of iterations are now executed in the reduced space. These are called the *"inner loops"*.

Normally, the inverse of S doesn't exist: If we project to a lower-dimensional space, we cannot transform back unambiguously; information is lost.

To return to the full space, we have to use a generalized inverse $\mathbf{S}^{-I} = [\mathbf{S}\mathbf{S}^T]^{-1}\mathbf{S}^T$.

We compute $\delta\mathbf{x} = \mathbf{S}^{-I}\delta\mathbf{w}$ and use this to modify x.

A number of iterations are now executed in the reduced space. These are called the *"inner loops"*.

Normally, the inverse of S doesn't exist: If we project to a lower-dimensional space, we cannot transform back unambiguously; information is lost.

To return to the full space, we have to use a generalized inverse $\mathbf{S}^{-I} = [\mathbf{S}\mathbf{S}^T]^{-1}\mathbf{S}^T$.

We compute $\delta\mathbf{x} = \mathbf{S}^{-I}\delta\mathbf{w}$ and use this to modify x.

At this stage, a new *"outer iteration"* at the full model resolution can be carried out.

$\star \quad \star \quad \star$

A number of iterations are now executed in the reduced space. These are called the *"inner loops"*.

Normally, the inverse of S doesn't exist: If we project to a lower-dimensional space, we cannot transform back unambiguously; information is lost.

To return to the full space, we have to use a generalized inverse $\mathbf{S}^{-I} = [\mathbf{S}\mathbf{S}^T]^{-1}\mathbf{S}^T$.

We compute $\delta\mathbf{x} = \mathbf{S}^{-I}\delta\mathbf{w}$ and use this to modify x.

At this stage, a new *"outer iteration"* at the full model resolution can be carried out.

$\star \quad \star \quad \star$

Note that the complete documentation of the ECMWF variational assimilation system is available at:

`http://www.ecmwf.int`

Pre-conditioning

The iteration process can be accelerated through the use of pre-conditioning.

Pre-conditioning

The iteration process can be accelerated through the use of pre-conditioning.

This involves a change of control variables that makes the cost function more spherical.

Pre-conditioning

The iteration process can be accelerated through the use of pre-conditioning.

This involves a change of control variables that makes the cost function more spherical.

An example of a change of variables might be to use the vorticity and divergence instead of the wind components.

Pre-conditioning

The iteration process can be accelerated through the use of pre-conditioning.

This involves a change of control variables that makes the cost function more spherical.

An example of a change of variables might be to use the vorticity and divergence instead of the wind components.

After pre-conditioning, each iteration gets closer to the minimum of the cost function, reducing computation time.

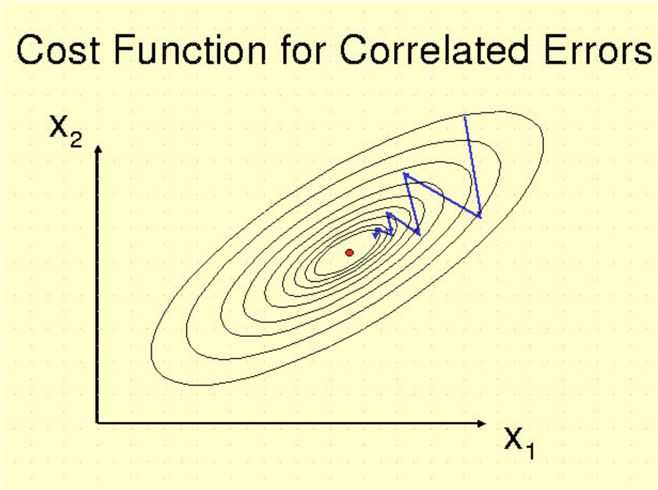
Pre-conditioning

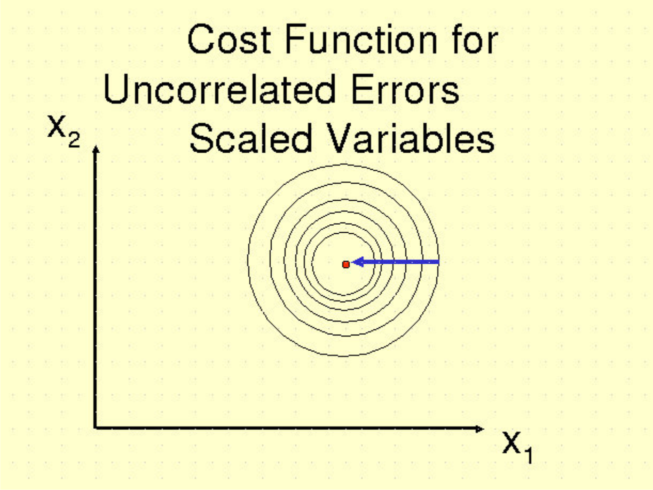
The iteration process can be accelerated through the use of pre-conditioning.

This involves a change of control variables that makes the cost function more spherical.

An example of a change of variables might be to use the vorticity and divergence instead of the wind components.

After pre-conditioning, each iteration gets closer to the minimum of the cost function, reducing computation time.





Advantages of 4D-Var

The most important advantage of 4D-Var is this:
We assume that:

- **(a) the model is perfect, and**
- **(b) the *a priori* error covariance B_0 at the initial time is known exactly.**

Advantages of 4D-Var

The most important advantage of 4D-Var is this:
We assume that:

- **(a) the model is perfect, and**
- **(b) the *a priori* error covariance B_0 at the initial time is known exactly.**

Then it can be shown that the 4D-Var analysis at the final time is identical to that of the extended Kalman filter.

Advantages of 4D-Var

The most important advantage of 4D-Var is this:
We assume that:

- **(a) the model is perfect, and**
- **(b) the** *a priori* **error covariance B_0 at the initial time is known exactly.**

Then it can be shown that the 4D-Var analysis at the final time is identical to that of the extended Kalman filter.

This means that ***implicitly*****, 4D-Var is able to evolve the forecast error covariance from B_0** ***to the final time.***

Advantages of 4D-Var

The most important advantage of 4D-Var is this:
We assume that:

- **(a) the model is perfect, and**
- **(b) the *a priori* error covariance B_0 at the initial time is known exactly.**

Then it can be shown that the 4D-Var analysis at the final time is identical to that of the extended Kalman filter.

This means that *implicitly*, 4D-Var is able to evolve the forecast error covariance from B_0 *to the final time*.

Unfortunately, this implicit covariance is not available at the end of the cycle, and neither is the new analysis error covariance.

4D-Var is able to find the best linear unbiased estimation but not its error covariance.

4D-Var is able to find the best linear unbiased estimation but not its error covariance.

4D-Var has been successfully implemented at ECMWF, Météo France, the Met Office, JMA and CMC.

4D-Var is able to find the best linear unbiased estimation but not its error covariance.

4D-Var has been successfully implemented at ECMWF, Météo France, the Met Office, JMA and CMC.

(3D-Var is used now in most other centres).

4D-Var is able to find the best linear unbiased estimation but not its error covariance.

4D-Var has been successfully implemented at ECMWF, Météo France, the Met Office, JMA and CMC.

(3D-Var is used now in most other centres).

Intensive research is under way in the HIRLAM Project to develop a limited-area 4D-Var system.

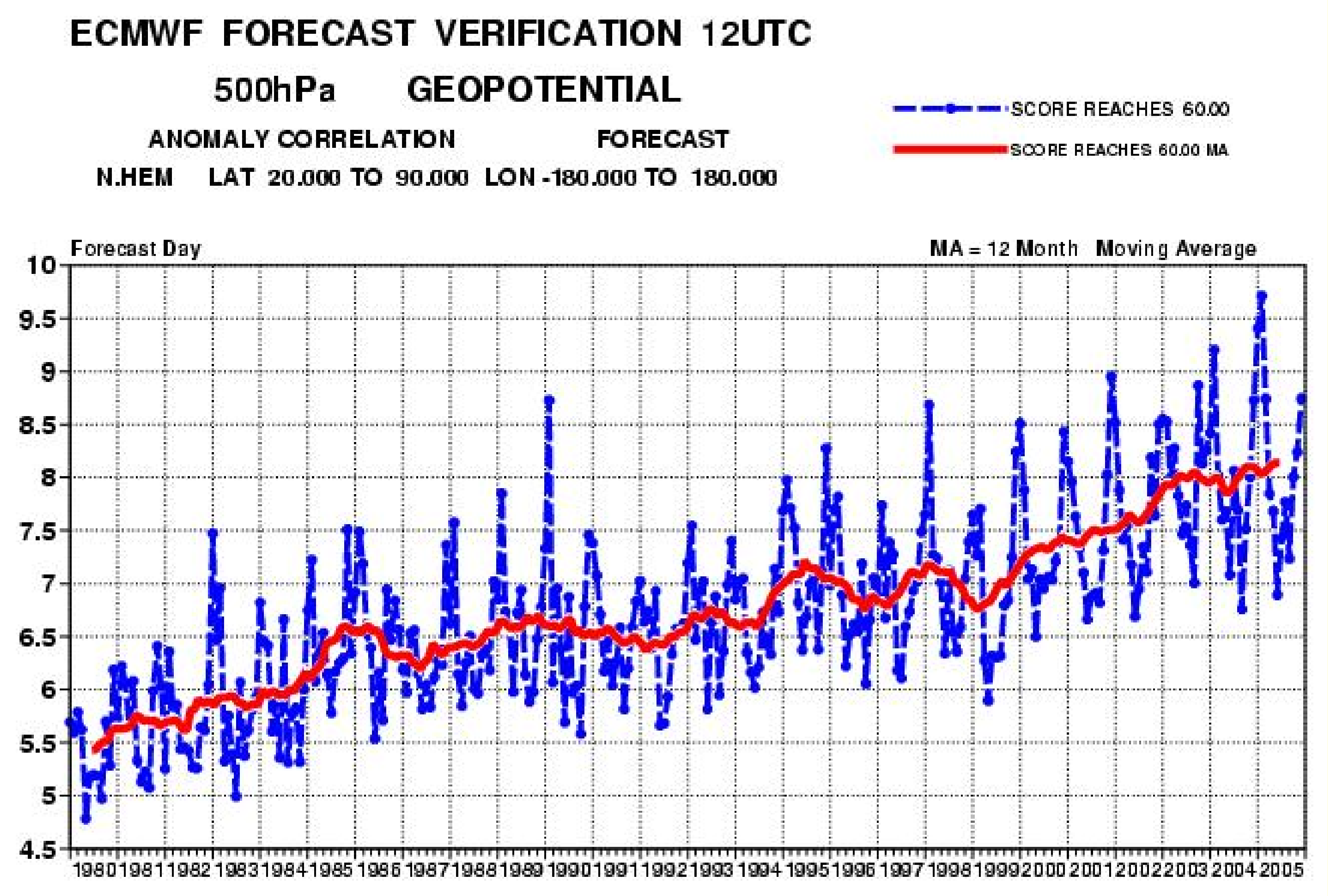
4D-Var is able to find the best linear unbiased estimation but not its error covariance.

4D-Var has been successfully implemented at ECMWF, Météo France, the Met Office, JMA and CMC.

(3D-Var is used now in most other centres).

Intensive research is under way in the HIRLAM Project to develop a limited-area 4D-Var system.

The following figure shows that implementation of 4D-Var has resulted in improved forecast scores.



ECMWF Forecast Verification

WGNE List of Operational Global Numerical Weather Prediction Systems (as of January 2006)

Forecast Centre (Country)	Computer (Peak in TFlop/s)	High resolution Model (FC Range in days)	Ensemble Model (FC Range in days)	Type of Data Assimilation
ECMWF (Europe)	IBM p690, 2x68 nodes (20)	T_L511 L60 (10)	T_L255 L40; M51 (10)	4D-VAR (T_L159)
Met Office (UK)	NEC SX6, 34 nodes NEC SX8 16 nodes (4)	~40km L50 (6)	~90km L38; M24 (3)	4D-Var (~120km)
Météo France (France)	Fujitsu VPP5000 (1.2)	T_L358 (C2.4) L41 (3)	T_L358(C2.4) L41; M11 (2.5)	4D-Var (T_L149)
DWD (Germany)	IBM p575; 2x52 nodes (2x3.1)	40 km L40 (7)	No EPS	3D-OI
HMC (Russia)	Itanium 4x4; Xeon 2x4 (0.10; 0.028)	T85 L31 (10); 0.72°x0.9° L28 (10)	No EPS	3D-OI
NCEP (USA)	IBM p655 (Cluster 1600) (7.8)	T382 L64 (7.5) T190 L64 (16)	T126 L28; M45 (16)	3D-Var (T382)
Navy/NRL (USA)	SGI O3000 (1024 proc) (1.125)	T239 L30 (6)	T119 L30; M10 (10)	3D-Var
CMC (Canada)	IBM p690, 108 nodes (4.3)	0.9°x0.9° L28 (10)	SEF (T_L149); GEM (1.2°); M16 (16)	Det: 4D-Var (1.5°, 0.9°) EPS: EnKF M96 (1.2°)
CPTEC/INPE (Brazil)	NEC SX6, 12 nodes (0.768)	T126L28, T213 L42 (15, 7)	T126 L28; M15 (15)	3D-Var
JMA (Japan)	Hitachi SR8000-E1, 80 nodes (0.768)	T_L319 L40 (9)	T106 L40; M25 (9)	4D-Var (T63)
CMA (China)	SW1; IBM P655/P690 (0.384; 7)	T213 L31 (10)	T106 L19; M33 (10)	3D-OI
KMA (Korea)	Cray X1E-8/1024-L (18.4)	T426 L40 (10)	T106 L30; M17 (8)	3D-Var
NCMRWF (India)	Cray SV1 24 processor (0.028)	T170 L28 (5)	No EPS	3D-VAR
BMRC (Australia)	NEC SX6, 28 nodes (1.792)	T_L239 L29 (10)	T_L119 L19; M33 (10)	3D-OI

Operational global NWP systems (January, 2006)

WGNE Overview of Plans at NWP Centres with Global Forecasting Systems

Part II: Global Modelling

c) Global Data Assimilation Scheme (Type, resolution, number of layers)

Forecast Centre (Country)	2006	2007	2008	2009	2010	2011
ECMWF (Europe)	4D-Var; T_L799 with T255 final inner loop; L91	4D-Var; T_L799 with T255 final inner loop; L91	4D-Var; T_L799 with T255 final inner loop; L91	4D-Var; T_L799 with T255 final inner loop; L91	?	?
Met Office (UK)	4D-Var; 120 km; L50	4D-Var; 120 km; L70	4D-Var; 120 km; L70	4D-Var; 75 km; L90	4D-Var; 75 km; L90	4D-Var; 75 km; L90
Météo France (France)	4D-Var; T159	4D-Var; T250	4D-Var; T250	4D-Var; T250	4D-Var; T350	4D-Var; T350
DWD (Germany)	OI; 40 km; L40	3D-Var; 40 km; L40	3D-Var; 40 km; L40	ETKF?	ETKF?	ETKF?
HMC (Russia)	OI; 0.9x0.72; L28	OI; 0.9x0.72; L28	3D-Var; 0.9x0.72; L28	?	?	?
NCEP (USA)	3D-Var; T382	Advanced-Var; T511	Advanced-Var; T511	Adv or 4D-Var; 20 km	Adv or 4D-Var; 20 km	4D-Var; 20 km
Navy/NRL (USA)	3D-Var; T239; L30	3D-Var; T239; L30	3D-Var; T319; L36	4D-Var	4D-Var	4D-Var
CMC (Canada)	4D-Var; 1.5°, 35 km; L58	4D-Var; 1.5°, 35 km; L58	4D-Var; 0.9°, 35 km; L80	4D-Var/EnKF?	4D-Var/EnKF?	4D-Var/EnKF?
CPTEC/INPE (Brazil)	3D-Var; 100 km	3D-Var; 60 km	LENKF; 40 km	LENKF; 40 km	LENKF; 40 km	LENKF; 20 km
JMA (Japan)	4D-Var; 120 km; L40	4D-Var; 80 km; L60	4D-Var; 60 km; L60	4D-Var; 60 km; L60	ETKF	ETKF
CMA (China)	NO RESPONSE					
KMA (Korea)	3D-Var; T426; L40	3D-Var; T426; L40	3D-Var; T426; L70	4D-Var? EnKF?	4D-Var? EnKF?	4D-Var? EnKF?
NCMRWF (India)						
BMRC (Australia)	3D-OI	Met Office 4D-VAR under ACCESS (?)	?	?	?	?

Planned future global data assimilation systems.

End of §5.6