Variational Data Assimilation
Current Status

Elías Valur Hólm
with contributions from Mike Fisher and Yannick Trémolet

ECMWF

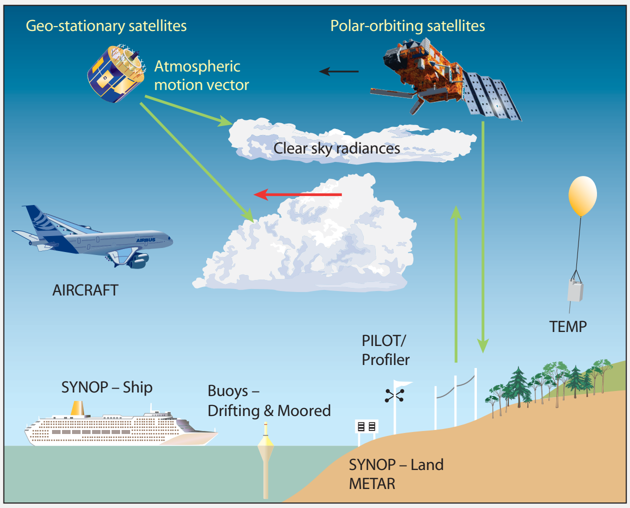
NORDITA Solar and stellar dynamos and cycles
October 2009

Outline

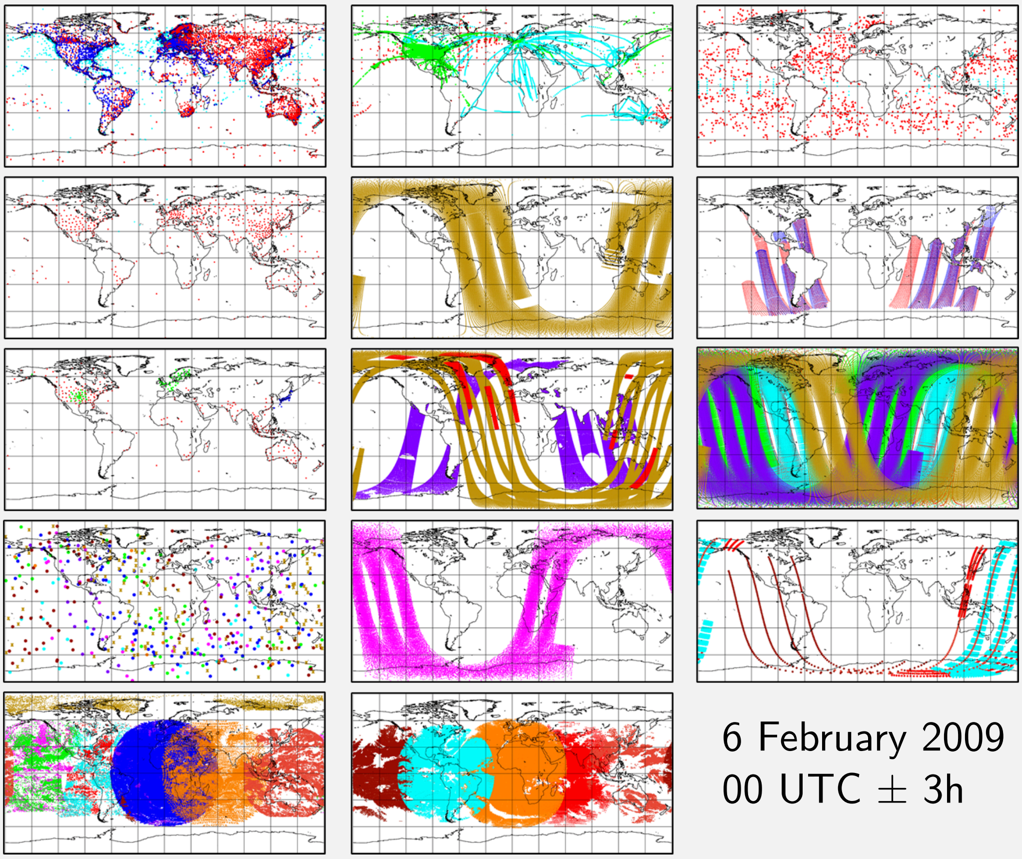
Outline

Models and observations

- Coupled atmosphere/ocean modelling combines sub-models:
  - Ocean
  - Ocean waves
  - Land surface
  - Atmosphere
  - Atmospheric chemistry/aerosols
- The observations of the system are a heterogeneos collection of direct/indirect measurements.
- Need a flexible and efficient approach to combining observations and models: (variational) data assimilation.

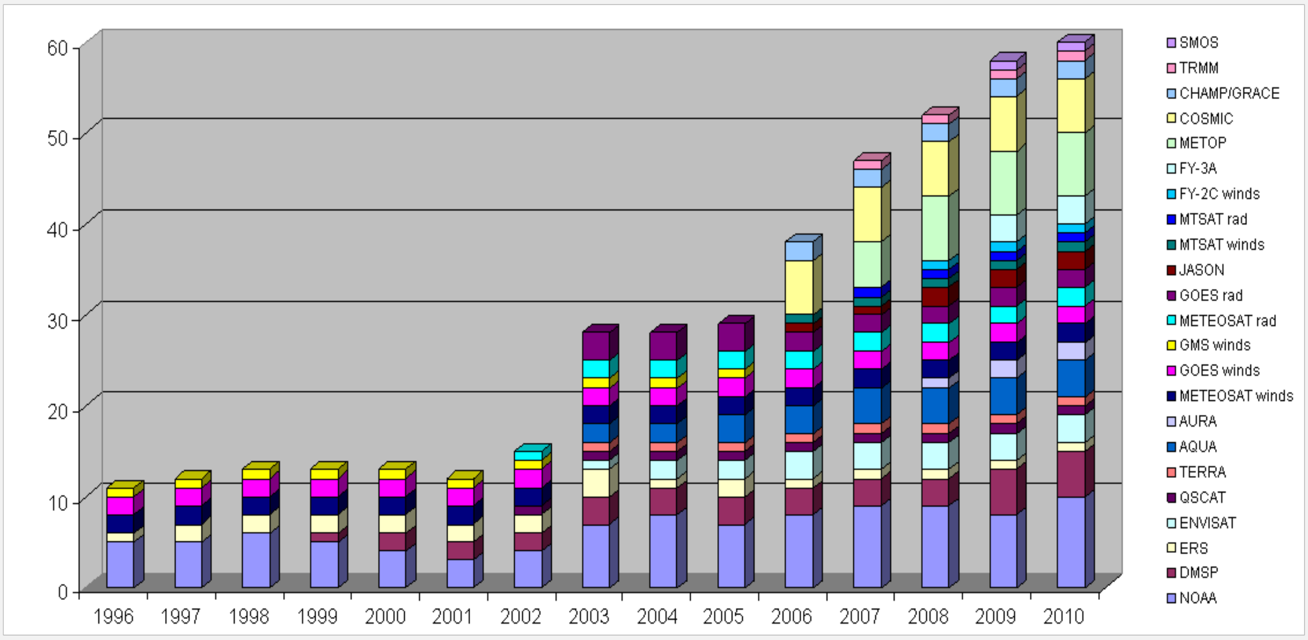


Observation Coverage



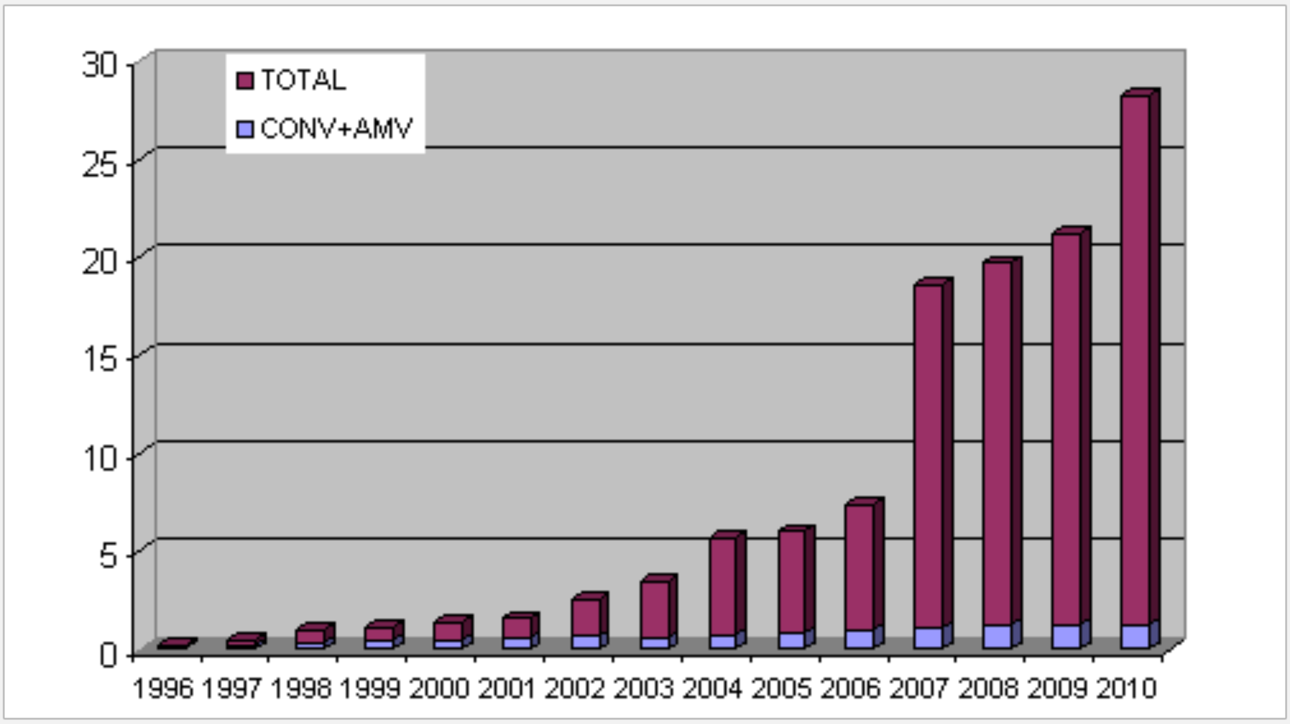
6 February 2009
00 UTC ± 3h

Observation Sources



Assimilating new data types requires a lot of resources (developments and computer time).

Observation Numbers



Observation numbers have increased regularly and will increase even faster in the future. Shown are used data (millions per 24 hours) which are ca 5% of all available data in 2008 (much redundant satellite information).

Combining models and observations

- Given are a model $\mathcal{M}(\mathbf{x})$, a set of observations $\mathbf{y}$, and a first guess model state, the background $\mathbf{x}_b$
- Each of those has associated errors $(\varepsilon_m, \varepsilon_o, \varepsilon_b)$.
- Each observation type has a model equivalent which maps observations to models: the observation operator $\mathcal{H}(\mathbf{y})$ (e. g. interpolation or ratiative transfer model).
- Find a best estimate $\mathbf{x}_a$ (the analysis) of the state vector $\mathbf{x}$ given $\mathbf{x}_b$ at $t = 0$ and observations y_n in the time interval $[0, \tau]$

$$\mathcal{M}(\mathbf{x}) = \varepsilon_m$$

$$\mathbf{x}(0) = \mathbf{x}_b + \varepsilon_b$$

$$y_n = \mathcal{H}_n(\mathbf{x}) + \varepsilon_{o,n} \quad n = 1, \ldots, N$$

Outline

Maximum Likelihood

- We define the analysis $\mathbf{x}_a$ as the most probable state of the system given a background state $\mathbf{x}_b$ and observations $\mathbf{y}$:

$$\mathbf{x}_a = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y} \text{ and } \mathbf{x}_b)$$

- It will be convenient to define a cost function:

$$J(\mathbf{x}) = -\log p(\mathbf{x}|\mathbf{y} \text{ and } \mathbf{x}_b) + K$$

where K is a constant.

- Since log is a monotonic function, $\mathbf{x}_a$ is also:

$$\mathbf{x}_a = \arg\min_{\mathbf{x}} J(\mathbf{x})$$

- Variational data assimilation comprises minimizing the cost function $J(\mathbf{x})$.

Maximum Likelihood and Bayes' Theorem

- Applying Bayes' theorem gives:

$$p(\mathbf{x}|\mathbf{y} \text{ and } \mathbf{x}_b) = \frac{p(\mathbf{y} \text{ and } \mathbf{x}_b|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y} \text{ and } \mathbf{x}_b)}$$

- $p(\mathbf{y} \text{ and } \mathbf{x}_b)$ is independent of $\mathbf{x}$ and *a priori* we know nothing about $\mathbf{x}$ (all values of $\mathbf{x}$ are equally likely) thus $p(\mathbf{x})$ is also independent of $\mathbf{x}$.
- Hence:

$$p(\mathbf{x}|\mathbf{y} \text{ and } \mathbf{x}_b) \propto p(\mathbf{y} \text{ and } \mathbf{x}_b|\mathbf{x})$$

- Finally, if observation errors and backgound errors are uncorrelated:

$$p(\mathbf{y} \text{ and } \mathbf{x}_b|\mathbf{x}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x}_b|\mathbf{x})$$

$$\Rightarrow J(\mathbf{x}) = -\log p(\mathbf{y}|\mathbf{x}) - \log p(\mathbf{x}_b|\mathbf{x}) + K$$

Maximum Likelihood and Cost Function

- The maximum likelihood approach is applicable to any probability density functions $p(\mathbf{y}|\mathbf{x})$ and $p(\mathbf{x}_b|\mathbf{x})$.
- Consider the special case of Gaussian p.d.f's:

$$\begin{aligned} p(\mathbf{x}_b|\mathbf{x}) &= \frac{1}{(2\pi)^{N/2}|\mathbf{B}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^T\mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b)\right] \\ p(\mathbf{y}|\mathbf{x}) &= \frac{1}{(2\pi)^{M/2}|\mathbf{R}|^{1/2}} \exp\left[-\frac{1}{2}[\mathcal{H}(\mathbf{x})-\mathbf{y}]^T\mathbf{R}^{-1}[\mathcal{H}(\mathbf{x})-\mathbf{y}]\right] \end{aligned}$$

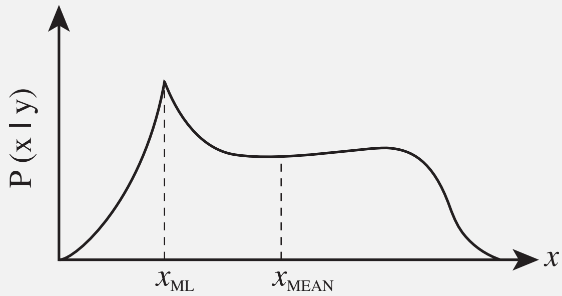
  where $\mathbf{B}$ and $\mathbf{R}$ are the background and observation error covariance matrices and $\mathcal{H}$ is the observation operator.
- With an appropriate choice of the constant:

$$J(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^T\mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}[\mathcal{H}(\mathbf{x})-\mathbf{y}]^T\mathbf{R}^{-1}[\mathcal{H}(\mathbf{x})-\mathbf{y}]$$

- This is the variational data assimilation cost function.

Maximum Likelihood: Remarks

- The maximum-likelihood approach is general: as long as we know the p.d.f's, we can define the cost function.
  - Finding the global minimum may not be easy for non-Gaussian p.d.f's.
- In practice, background errors are usually assumed to be Gaussian (or a nonlinear transformation is applied to *make* them Gaussian).
- Non-Gaussian observation errors are taken into account.
  - Directionally-ambiguous wind observations from scatterometers,
  - Observations contaminated by occasional gross errors, which make outliers much more likely than implied by a Gaussian model.
- For Gaussian errors and linear observation operators, the maximum likelihood analysis coincides with the minimum variance solution. This is not the case in general:



Outline

3D-Var and 4D-Var

$$J(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}_b)^T\mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \frac{1}{2}[\mathcal{H}(\mathbf{x})-\mathbf{y}]^T\mathbf{R}^{-1}[\mathcal{H}(\mathbf{x})-\mathbf{y}]$$

- We have not precisely defined the space over which the state variable $\mathbf{x}$ is defined or the observation operator $\mathcal{H}$.
- Depending on the choice of $\mathbf{x}$ and $\mathcal{H}$, the general approach described earlier will lead to different variational data assimilation methods.
- The simplest approach is to consider $\mathbf{x}$ as the state over the 3D spatial domain at analysis time, while $\mathcal{H}$ spatially interpolates this state and converts model variables to observed quantities: this is 3D-Var.
- Another more common approach is to consider $\mathbf{x}$ as the state over the 3D spatial domain and over the period for which observations are available, while $\mathcal{H}$ spatially and temporally interpolates this state and converts model variables to observed quantities: this is 4D-Var.

4D-Var

- We now discretize the assimilation window in time and define $\mathbf{x} = \{\mathbf{x}_i\}_{i=0,n}$ and $\mathbf{y} = \{\mathbf{y}_i\}_{i=0,n}$ where $\mathbf{x}_i$ and $\mathbf{y}_i$ are the state and observations at time t_i for $i = 0, \dots, n$.
- Assuming that observation errors are uncorrelated in time, $\mathbf{R}$ is block diagonal, with blocks $\mathbf{R}_i$ corresponding to the observations at time t_i.
- The 4D-Var cost function is:

$$J(\mathbf{x}) = \frac{1}{2}(\mathbf{x}_0-\mathbf{x}_b)^T\mathbf{B}^{-1}(\mathbf{x}_0-\mathbf{x}_b)+\frac{1}{2}\sum_{i=0}^{n}[\mathcal{H}_i(\mathbf{x}_i)-\mathbf{y}_i]^T\mathbf{R}_i^{-1}[\mathcal{H}_i(\mathbf{x}_i)-\mathbf{y}_i]$$

- $\mathcal{H}_i$ represents a spatial interpolation and transformation from model variables to observed variables (i.e. a 3D-Var-style observation operator).

Strong Constraint 4D-Var

- The states at various times are not independent: they are related through the forecast model:

$$\mathbf{x}_i = \mathcal{M}_i(\mathbf{x}_{i-1})$$

where $\mathcal{M}_i$ is the forecast model integrated from time t_{i-1} to time t_i.

- By introducing the vectors $\mathbf{x}_i$, the unconstrained minimization problem:

$$\mathbf{x}_a = \arg\min_{\mathbf{x}} J(\mathbf{x})$$

became a strong constraints minimization problem:

$$\begin{aligned} \mathbf{x}_a &= \arg\min_{\mathbf{x}_0} J(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_k) \\ \text{subject to } \mathbf{x}_i &= \mathcal{M}_i(\mathbf{x}_{i-1}) \text{ for } i = 1, \ldots, n \end{aligned}$$

- This form of 4D-Var is called strong constraint 4D-Var.

Strong Constraint 4D-Var

- The 4D-Var cost function is:

$$\begin{aligned} J(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_k) &= \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}_b)^T \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_b) \\ &+ \frac{1}{2}\sum_{i=0}^{n}[\mathcal{H}_i(\mathbf{x}_i) - \mathbf{y}_i]^T \mathbf{R}_i^{-1}[\mathcal{H}_i(\mathbf{x}_i) - \mathbf{y}_i] \end{aligned}$$

- 4D-Var determines the analysis state at every gridpoint and at every time within the analysis window i.e. a four-dimensional analysis of the available data.
- In deriving strong constraint 4D-Var, we have assumed that the observation operators and the model are perfect.
- As a consequence of the perfect model assumption, the analysis corresponds to a trajectory (i.e. an integration) of the forecast model.
- The 4D-Var framework can also includes model errors – weak constraint 4D-Var.

Outline

Minimizing the cost function

- We want to minimize the cost function:

$$J(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}_b)^T \mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}_b) + \frac{1}{2}[\mathcal{H}(\mathbf{x}) - \mathbf{y}]^T \mathbf{R}^{-1}[\mathcal{H}(\mathbf{x}) - \mathbf{y}]$$

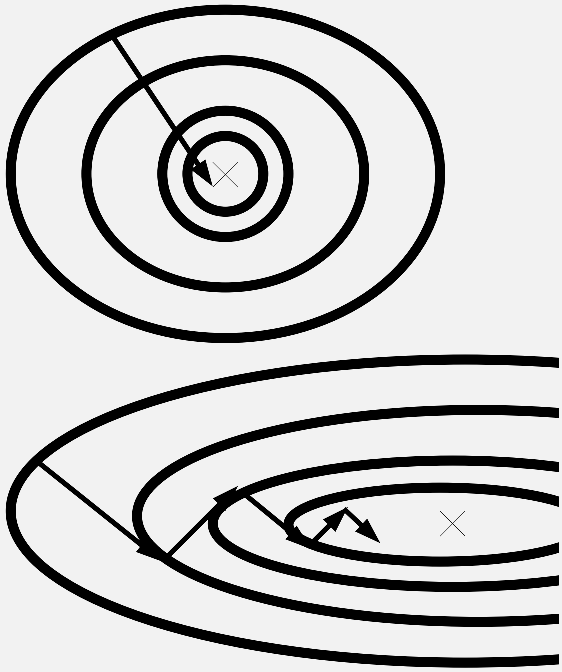
- This is a very large-scale minimization problem ($\dim(\mathbf{x}) \approx 300 \times 10^6$ for the operational system at ECMWF.)
- Derivative-free algorithms are too slow (because each function evaluation gives very limited information about the shape of the cost function and in which direction the minimum might be).
- Practical algorithms for minimizing the cost function require its gradient.
- For poorly conditioned problems, where the iso-surfaces of the costfunction are far from spherical, a preconditioner is needed for efficient minimization.

Minimizing the cost function

- Minimization algorithms work best if the iso-surfaces of the cost function are nearly spherical, as measured by the eigenvalues of the Hessian.
- Each eigenvalue corresponds to the curvature in the direction of the corresponding eigenvector. The convergence rate will depend on the condition number:

$$\kappa = \lambda_{\max}/\lambda_{\min}$$

- The convergence can be accelerated by reducing the condition number



Preconditioning

- We can speed up the convergence of the minimization by a change of variables $\chi = \mathbf{L}^{-1}\delta\mathbf{x}$ (i.e. $\delta\mathbf{x} = \mathbf{L}\chi$), where $\mathbf{L}$ is chosen to make the cost function more spherical.
- A common choice is $\mathbf{L} = \mathbf{B}^{1/2}$.
- The 4D-Var cost function becomes:

$$J(\chi) = \frac{1}{2}\chi^T\chi + \frac{1}{2}(\mathbf{HL}\chi - \mathbf{d})^T\mathbf{R}^{-1}(\mathbf{HL}\chi - \mathbf{d}).$$

- With this change of variables, the Hessian becomes:

$$J''_\chi = \mathbf{I} + \mathbf{L}^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{HL}.$$

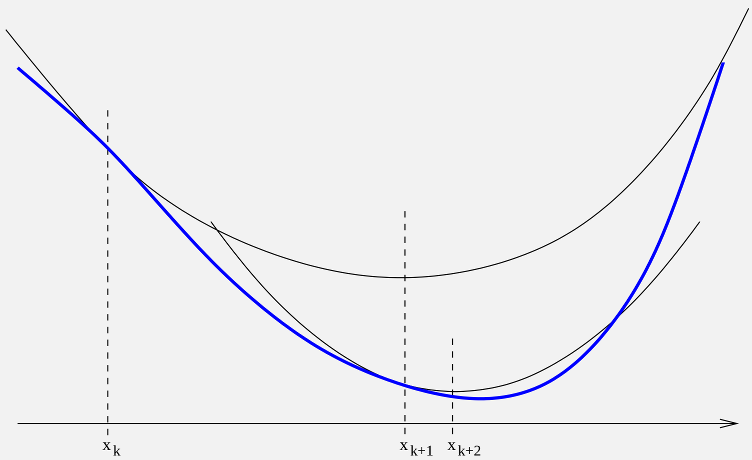
- The presence of the identity matrix in this expression guarantees that all eigenvalues are ≥ 1.
- There are no small eigenvalues to destroy the conditioning of the problem.

Minimizing the cost function

- For general nonlinear functions use quasi-Newton methods, e. g. a (limited memory) BFGS algorithm, available at http://www-rocq.inria.fr/ gilbert/modulopt/optimization-routines/m1qn3/m1qn3.html
- An important special case occurs if the observation operator $\mathcal{H}$ is linear. In this case, the cost function is strictly quadratic, and the gradient is linear.
- In this case, it makes sense to determine the analysis by solving the linear equation $\nabla J(\mathbf{x}) = 0$.
- Since the matrix $J'' = \mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}$ is symmetric and positive definite, the best algorithm to use is conjugate gradients.
- A good introduction to the method can be found online: *An Introduction to the Conjugate Gradient Method Without the Agonizing pain*, Shewchuk (1994).
- This will be useful in the incremental 4D-Var.

The Incremental Method

- A variant of the Newton method can be used: the nonlinear cost function is approximated by a quadratic cost function around the current guess. This quadratic cost function is minimized to provide an updated guess and the process is repeated.
- One complex problem is replaced by a series of (slightly) easier problems.



- The conjugate gradient algorithm can be used to solve efficiently the quadratic minimization problems.

The Incremental Method

- The cost function is written as a function of the correction to the first guess (the increment) $\delta\mathbf{x} = \mathbf{x} - \mathbf{x}_g$:

$$\begin{aligned} J(\mathbf{x}_g + \delta\mathbf{x}) &= \frac{1}{2}(\mathbf{x}_g + \delta\mathbf{x} - \mathbf{x}_b)^T \mathbf{B}^{-1}(\mathbf{x}_g + \delta\mathbf{x} - \mathbf{x}_b) \\ &+ \frac{1}{2}[\mathcal{H}(\mathbf{x}_g + \delta\mathbf{x}) - \mathbf{y}]^T \mathbf{R}^{-1}[\mathcal{H}(\mathbf{x}_g + \delta\mathbf{x}) - \mathbf{y}] \end{aligned}$$

- The quadratic approximation of the cost function is obtained by linearizing around the curent guess:

$$J(\delta\mathbf{x}) = \frac{1}{2}(\delta\mathbf{x} + \mathbf{b})^T \mathbf{B}^{-1}(\delta\mathbf{x} + \mathbf{b}) + \frac{1}{2}(\mathbf{H}\delta\mathbf{x} + \mathbf{d})^T \mathbf{R}^{-1}(\mathbf{H}\delta\mathbf{x} + \mathbf{d})$$

  where $\mathbf{b} = \mathbf{x}_g - \mathbf{x}_b$, $\mathbf{d} = \mathcal{H}(\mathbf{x}_g) - \mathbf{y}$ and $\mathbf{H}$ is the Jacobian of $\mathcal{H}$.

- The gradient is:

$$\nabla J(\delta\mathbf{x}) = \mathbf{B}^{-1}(\delta\mathbf{x} + \mathbf{b}) + \mathbf{H}^T \mathbf{R}^{-1}(\mathbf{H}\delta\mathbf{x} + \mathbf{d})$$

Calculating the Gradient: Tangent Linear and Adjoint

- To minimize the cost function, we must be able to calculate its gradient:

$$\nabla J(\delta\mathbf{x}) = \mathbf{B}^{-1}(\delta\mathbf{x} + \mathbf{b}) + \mathbf{H}^T \mathbf{R}^{-1}(\mathbf{H}\delta\mathbf{x} + \mathbf{d})$$

- The Jacobians $\mathbf{H}$ and $\mathbf{H}^T$ are much too large to be represented explicitly: we can only represent these as operators (subroutines) that calculate matrix-vector products.
- These codes are called the tangent linear code for $\mathbf{H}$ and the adjoint code for $\mathbf{H}^T$.
- For a good introduction about writing adjoints, see:
  X. Y. Huang and X. Yang, 1996, Variational Data Assimilation with the Lorenz model, HIRLAM Technical Report 26.

Writing the Adjoint Code

- Each line of the subroutine that applies $\mathcal{H}$ (including the forecast model) can be considered as a function h_k, so that

$$\mathcal{H}(\mathbf{x}) \equiv h_K \circ h_{K-1} \circ \cdots \circ h_1(\mathbf{x}).$$

- Each of the functions h_k can be linearized, to give the corresponding linear function $\mathbf{h}_k$. Each of these is extremely simple, and can be represented by one or two lines of code.
- The resulting code is called the tangent linear of $\mathcal{H}$ and:

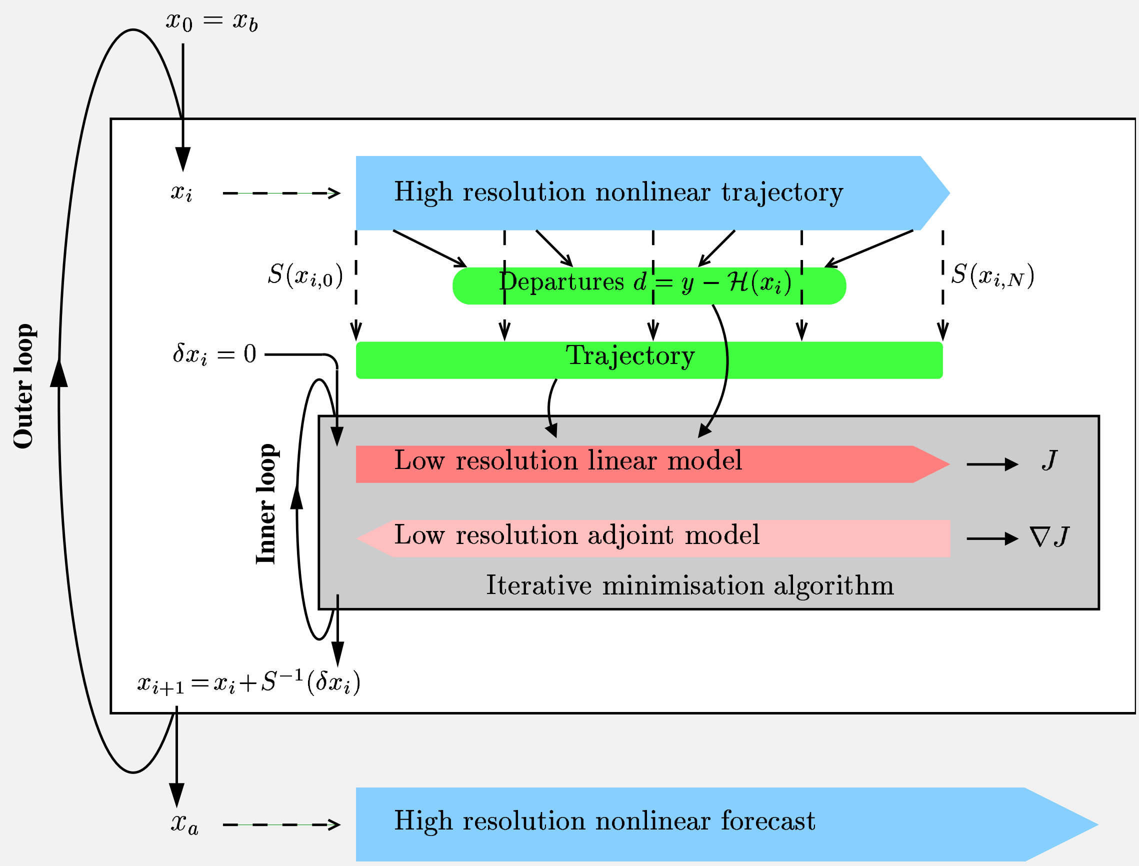
$$\mathbf{H}\delta\mathbf{x} \equiv \mathbf{h}_K \mathbf{h}_{K-1} \cdots \mathbf{h}_1 \delta\mathbf{x}$$

- The transpose, $\mathbf{H}^T\delta\mathbf{x} \equiv \mathbf{h}_1^T \mathbf{h}_2^T \cdots \mathbf{h}_K^T \delta\mathbf{x}$, is called the adjoint of $\mathcal{H}$.
- Again, each $\mathbf{h}_k^T$ is extremely simple – just a few lines of code.
- The difficulties in writing an adjoint can come from:
  - Non differentiable functions in the nonlinear cost function (physical processes, e. g. phase transitions),
  - The length of the code (automatic tools can help).

The Incremental Method

- The 4D-Var cost function and its gradient can be evaluated for the cost of:
  - one integration of the forecast model,
  - one integration of the adjoint model.
- This cost is still prohibitive:
  - A typical minimization requires between 10 and 100 iterations,
  - The cost of the adjoint is typically 3 times that of the forward model.
  - The cost of the analysis would be roughly equivalent to between 20 and 200 days of model integration (with a 12h window).
- The incremental algorithm reduces the cost of 4D-Var by reducing the resolution of the model and using simplified physics (or by using a perturbation forecast model).
- The analysis increments are calculated at reduced resolution and must be interpolated to the high-resolution model's grid.
- The departures **d** are always evaluated using the full-resolution versions of $\mathcal{H}$ (and $\mathcal{M}$) i.e. the observations are always compared with the *full resolution* state.

Incremental 4D-Var



Outline

Summary

- The Maximum Likelihood approach is general and can in principle be applied to non-Gaussian, nonlinear analysis.
- 3D-Var and 4D-Var derive from the maximum likelihood principle.
- 4D-Var is an extension of 3D-Var to the case where observations are distributed in time.
- The cost function is minimized using algorithms based on knowledge of its gradient.
- The incremental method with appropriate preconditioning allows the computational cost to be reduced to acceptable levels.
- In strong constraint 4D-Var the model is assumed to be perfect, so that the four-dimensional analysis state corresponds to an integration (trajectory) of the model. Model errors also fit into the 4D-Var framework (formulation straightforward, the difficulty is to usefully characterize these errors beyond the trivial random noise).

Summary

- Most (all?) of the main NWP centres run variational data assimilation schemes operationally.
  - ECMWF, United Kingdom, France, Germany, Canada, USA (NCEP, NRL, GMAO), Japan, Korea, Taiwan, China, Australia, HIRLAM countries, ALADIN countries...
- Forecast performance has improved over the years, in particular because of the ability of variational systems to adapt to and benefit from the varying components of the global observing system.
- Other aspects are important but were not covered in this talk:
  - Modelling of **B** and balance considerations,
  - Definition of the observation operators,
  - Observation variational bias correction.

Outline

Some online resources

- Excellent presentations on data assimilation including practical details at the JCSDA Summer Colloquium on Data Assimilation July 2009: *http : //www.jcsda.noaa.gov/meetings*$_2$009*SummerColloq.php*
- Classic practical guide to starting 4D-Var from the HIRLAM group Technical Report 26: *http : //www.hirlam.org/publications/TechReports/index.html*
- Downloadable ensemble Kalman filter data assimilation system from NCAR (Fortran 90): *http : //www.image.ucar.edu/DAReS/DART/*
- Resources for automatic tangent linear/adjoint code development available online. They are an assistance in development, but the resulting code always needs some reorganization for speed. Example is INRIA's Tapenade code, *http : //tapenade.inria.fr* : 8080/*tapenade/index.jsp.*
- General minimization routine (limited memory quasi-Newton) from INRIA at *http : //www − rocq.inria.fr/ gilbert/modulopt/optimization − routines/m1qn3/m1qn3.html*