Advanced School on Data Assimilation

CMCC, Bologna

7-11 June 2008

Adjoint Methods

Lecturer: Srdjan Dobricic

CMCC, Bologna

Model operator

The model state **x** is propagated in time using the non-linear model operator *N*():

$$\frac{\partial \mathbf{x}}{\partial t} = N(\mathbf{x})$$

By applying small perturbations around the state **x** we may approximate the non-linear dynamic operator by the linear operator **N**:

$$\frac{\partial(\delta\mathbf{x})}{\partial t} = \left[\frac{\partial N(\mathbf{x})}{\partial \mathbf{x}}\right]_{\mathbf{x}=\mathbf{x}_0} \delta\mathbf{x} = \mathbf{N}\delta\mathbf{x}$$

Adjoint operator

We can define the adjoint model operator with respect to the inner product of the linearized dynamic equation with an arbitrary vector $\delta\mathbf{x}^*$

In the discrete system we are dealing with vectors and matrices. We define the inner product by

$$\delta\mathbf{x}^{*T}\mathbf{G}\delta\mathbf{x}$$

G is symmetric positive definite. It defines the norm of interest. For example, in the Euclidian space **G**=**I**.

The definition of the adjoint model operator follows from the bilinear identity:

$$\delta\mathbf{x}^{*T}\mathbf{G}\mathbf{N}\delta\mathbf{x} = \delta\mathbf{x}^{T}\mathbf{N}^{T}\mathbf{G}\delta\mathbf{x}^{*} = \delta\mathbf{x}^{T}\mathbf{G}\mathbf{N}^{*}\delta\mathbf{x}^{*}$$

$$\mathbf{N}^{*} = \mathbf{G}^{-1}\mathbf{N}^{T}\mathbf{G}$$

Adjoint operator

Multiply the linearized dynamic equation by $\delta\mathbf{x}^{*}\mathbf{G}$

$$\delta\mathbf{x}^{*T}\mathbf{G}\frac{\partial(\delta\mathbf{x})}{\partial t}=\delta\mathbf{x}^{*T}\mathbf{GN}\delta\mathbf{x}$$

$$\frac{\partial}{\partial t}\left(\delta\mathbf{x}^{*T}\mathbf{G}\delta\mathbf{x}\right)-\delta\mathbf{x}^{T}\mathbf{G}\frac{\partial(\delta\mathbf{x}^{*})}{\partial t}=\delta\mathbf{x}^{T}\mathbf{GN}^{*}\delta\mathbf{x}^{*}$$

$$\frac{\partial}{\partial t}\left(\delta\mathbf{x}^{*T}\mathbf{G}\delta\mathbf{x}\right)=0$$

$$\Rightarrow-\frac{\partial(\delta\mathbf{x}^{*})}{\partial t}=\mathbf{N}^{*}\delta\mathbf{x}^{*}$$

The adjoint model operator integrates backwards in time!

Adjoint solution of the cost function gradient

$$\mathbf{J}' = \mathbf{B}_0^{-1}\delta\mathbf{x}_0 - \sum_{k=1}^{n}\mathbf{M}_{1,k}^{\mathrm{T}}\mathbf{H}_k^{\mathrm{T}}\mathbf{R}_k^{-1}\left(\mathbf{d}_k - \mathbf{H}_k\mathbf{M}_{0,k}\delta\mathbf{x}_0\right)$$

Initially we may set: $\delta\mathbf{x}_0 = \mathbf{0}$

$$\mathbf{J}' = -\sum_{k=1}^{n}\mathbf{M}_{1,k}^{\mathrm{T}}\mathbf{H}_k^{\mathrm{T}}\mathbf{R}_k^{-1}\mathbf{d}_k$$

Adjoint equations are defined by:

$$\begin{aligned}
\delta\mathbf{x}_{n+1}^{*} &= 0 \\
\delta\mathbf{x}_{n}^{*} &= \mathbf{H}_n^{*}\mathbf{R}_n^{-1}\mathbf{d}_n \\
\delta\mathbf{x}_{n-1}^{*} &= \mathbf{M}_n^{*}\delta\mathbf{x}_n^{*} + \mathbf{H}_{n-1}^{*}\mathbf{R}_{n-1}^{-1}\mathbf{d}_{n-1} \\
&\vdots \\
\delta\mathbf{x}_{k}^{*} &= \mathbf{M}_{k+1}^{*}\delta\mathbf{x}_{k+1}^{*} + \mathbf{H}_k^{*}\mathbf{R}_k^{-1}\mathbf{d}_k \\
&\vdots \\
\delta\mathbf{x}_{0}^{*} &= \mathbf{M}_1^{*}\delta\mathbf{x}_1^{*}
\end{aligned}$$

Adjoint solution of the cost function gradient

$$\delta\mathbf{x}_0^* = \mathbf{M}_1^*\delta\mathbf{x}_1^* = \mathbf{M}_1^*\left(\mathbf{M}_2^*\delta\mathbf{x}_2^* + \mathbf{H}_1^*\mathbf{R}_1^{-1}\mathbf{d}_1\right) =$$

$$= \mathbf{M}_1^*\mathbf{M}_2^*\delta\mathbf{x}_2^* + \mathbf{M}_1^*\mathbf{H}_1^*\mathbf{R}_1^{-1}\mathbf{d}_1$$

$$\vdots \qquad\qquad 0$$

$$\delta\mathbf{x}_0^* = \mathbf{M}_{1,\mathrm{n}}^*\delta\mathbf{x}_{\mathrm{n}+1}^* + \sum_{k=1}^{n}\mathbf{M}_{1,k}^*\mathbf{H}_k^*\mathbf{R}_k^{-1}\mathbf{d}_k$$

$$\mathbf{M}_{1,k}^* = \mathbf{M}_{1,k}^{\mathrm{T}}; \mathbf{H}_k^* = \mathbf{H}_k^{\mathrm{T}} \qquad \Rightarrow \delta\mathbf{x}_0^* = -\mathbf{J}'$$

Calculation of adjoint equations in practice

The first step is to linearize the model *(if it is non-linear)*.

- Potential problems are non-differentiable statements.
- “If loops” are solved by saving the trajectory which corresponds to the option which is executed in the non-linear model.

The adjoint executes tangent linear model operations in the reverse mode.

Calculation of adjoint equations in practice

Equation:

$$c = a + b$$

In fact this is a set of equations:

$$\begin{aligned} c &= a + b \\ a &= a \\ b &= b \end{aligned}$$

In the matrix form:

$$\begin{vmatrix} c \\ a \\ b \end{vmatrix} = \begin{vmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} c \\ a \\ b \end{vmatrix}$$

Calculation of adjoint equations in practice: Method of the transpose of the linear operator

Linear operator:

$$\mathbf{A} = \begin{vmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Transpose of the linear operator:

$$\mathbf{A}^T = \begin{vmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{vmatrix}$$

Calculation of adjoint equations in practice: Method of the transpose of the linear operator

Adjoint equations in the matrix form:

$$\begin{vmatrix} c^* \\ a^* \\ b^* \end{vmatrix} = \begin{vmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{vmatrix} \begin{vmatrix} c^* \\ a^* \\ b^* \end{vmatrix}$$

Adjoint equations:

$$b^* = b^* + c^*$$

$$a^* = a^* + c^*$$

$$c^* = 0$$

Calculation of adjoint equations in practice: Method of the partial derivative of the inner product

$$\frac{\partial}{\partial w}\left\langle \mathbf{w}^{*},\mathbf{A}\mathbf{w}\right\rangle = \frac{\partial}{\partial w}\left(\mathbf{w}^{*T}\mathbf{A}\mathbf{w}\right) = \frac{\partial}{\partial w}\left(\mathbf{w}^{T}\mathbf{A}^{T}\mathbf{w}^{*}\right) = \mathbf{A}^{T}\mathbf{w}^{*}$$

$$\left\langle \mathbf{w}^{*},\mathbf{A}\mathbf{w}\right\rangle \quad \rightarrow \quad \begin{vmatrix} c^{*} \\ a^{*} \\ b^{*} \end{vmatrix}^{T} \cdot \left\{ \begin{vmatrix} c \\ a \\ b \end{vmatrix} = \begin{vmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} c \\ a \\ b \end{vmatrix} \right\}$$

$$\left\langle \mathbf{w}^{*},\mathbf{A}\mathbf{w}\right\rangle \quad \rightarrow \quad \begin{vmatrix} c^{*} & a^{*} & b^{*} \end{vmatrix} \begin{vmatrix} c \\ a \\ b \end{vmatrix} = \begin{vmatrix} c^{*} & a^{*} & b^{*} \end{vmatrix} \begin{vmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} c \\ a \\ b \end{vmatrix}$$

Calculation of adjoint equations in practice: Method of the partial derivative of the inner product

$$\left\langle \mathbf{w}^*, \mathbf{A}\mathbf{w} \right\rangle \quad \rightarrow \quad cc^* + aa^* + bb^* = \begin{vmatrix} c^* & a^* & b^* \end{vmatrix} \begin{vmatrix} a+b \\ a \\ b \end{vmatrix}$$

$$\left\langle \mathbf{w}^*, \mathbf{A}\mathbf{w} \right\rangle \quad \rightarrow \quad cc^* + aa^* + bb^* = (a+b)c^* + aa^* + bb^*$$

$$\frac{\partial}{\partial b}\left\langle \mathbf{w}^*, \mathbf{A}\mathbf{w} \right\rangle \quad \rightarrow \quad b^* = b^* + c^*$$

$$\frac{\partial}{\partial a}\left\langle \mathbf{w}^*, \mathbf{A}\mathbf{w} \right\rangle \quad \rightarrow \quad a^* = a^* + c^*$$

$$\frac{\partial}{\partial c}\left\langle \mathbf{w}^*, \mathbf{A}\mathbf{w} \right\rangle \quad \rightarrow \quad c^* = 0$$

Adjoint equations

Calculation of adjoint equations in practice: Automatic compilers

- Both methods, transpose of the linear operator and the partial derivative of the inner product may be used to define a set of formulae to write the adjoint computer code from an existing computer code of a linear operator.

- Automatic adjoint compilers are TAMC, TAF and TEPENADE (TAMC and TEPENADE are free).

- The major difference is in the way they treat the non-linear trajectory (saving or recomputing).

- Potential problems with automatic compilers are that they may produce computationally inefficient codes.

- A good coding rule is to make both hand coded program and the automatically generated program, and to compare them.