Hands-on derivation of tangent linear and adjoint codes

Marcin Chrust and Sébastien Massart

ECMWF

11 May 2021

ECMWF

Road map

1. Before starting
   - Commands to type in a terminal:
     git checkout ssh://git@git.ecmwf.int/tcd/dacourse.git
     cd dacourse
     module load python3
   - You should have a firefox window with the documentation of the two codes we will be using in this practical session

2. Simple exercise of adjoint derivation and test of adjoint

$$x = Ay + Bz^2$$

3. Manual derivation of tangent linear for a 1D advection code

$$\frac{\partial \xi}{\partial t} = u\frac{\partial \xi}{\partial x}$$

Simple exercise

1. Find the adjoint of this nonlinear statement using the matrix method:

$$x = Ay + Bz^2$$

Note: A and B are constants, x, y, z are variables

Simple exercise

1. Find the adjoint of this nonlinear statement using the matrix method:

$$x = A\,y + B\,z^2$$

   Note: A and B are constants, x, y, z are variables

2. Numerical example
   - Open the file *simple_example.py*
   - Naming conventions
     - suffix d for all tangent linear variables
     - suffix b for all adjoint variables
     - suffix 0 for the trajectory
   - Look at the structure of the source code
     Why do we check $\langle xd\,,\,xd\rangle \stackrel{?}{=} \langle yd\,,\,yb\rangle + \langle zd\,,\,zb\rangle$
   - Run the executable
     - *cd SIMPLE_EXAMPLE*
     - *python3 simple_example.py*
   - Repeat the test by changing the value of the adjoint initialisation
     - *uncomment line 167*
     - Try to fix the problem

Simple exercise (solution 1/2)

- Find the adjoint of this nonlinear statement using the matrix method:

$$x = Ay + Bz^2$$

- Nonlinear statement

$$\begin{aligned} z &= z \\ y &= y \\ x &= Ay + Bz^2 \end{aligned}$$

- Tangent linear statement

$$\begin{aligned} z_d &= z_d \\ y_d &= y_d \\ x_d &= Ay_d + 2Bz\,z_d \end{aligned}$$

- Tangent linear statement in matrix form

$$\begin{bmatrix} z_d \\ y_d \\ x_d \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2Bz & A & 0 \end{bmatrix} \begin{bmatrix} z_d \\ y_d \\ x_d \end{bmatrix} = \mathbf{F} \begin{bmatrix} z_d \\ y_d \\ x_d \end{bmatrix}$$

- Adjoint statement in matrix form

$$\begin{bmatrix} z_b \\ y_b \\ x_b \end{bmatrix} = \mathbf{F}^T \begin{bmatrix} z_b \\ y_b \\ x_b \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2Bz \\ 0 & 1 & A \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} z_b \\ y_b \\ x_b \end{bmatrix}$$

- Adjoint statement

$$\begin{aligned} z_b &= z_b + 2Bz\,x_b \\ y_b &= y_b + Ax_b \\ x_b &= 0 \end{aligned}$$

Simple exercise (solution 2/2)

- Find the adjoint of this nonlinear statement using the matrix method:

$$\begin{aligned} x_d &= A\,y_d + B\,z_d^2 \\ &= F \begin{pmatrix} y_d \\ z_d \end{pmatrix} \end{aligned}$$

- Adjoint test

$$\left\langle x\,,\, F\begin{pmatrix} y_d \\ z_d \end{pmatrix} \right\rangle \stackrel{?}{=} \left\langle F^T x\,,\, \begin{pmatrix} y_d \\ z_d \end{pmatrix} \right\rangle \quad \text{for any } x$$

$$\langle x\,,\, x_d \rangle \stackrel{?}{=} \left\langle \begin{pmatrix} y_b \\ z_b \end{pmatrix},\, \begin{pmatrix} y_d \\ z_d \end{pmatrix} \right\rangle \quad \text{with } \begin{pmatrix} y_b \\ z_b \end{pmatrix} = F^T x$$

$$\langle x\,,\, x_d \rangle \stackrel{?}{=} \langle y_b\,,\, y_d \rangle + \langle z_b\,,\, z_d \rangle \quad \text{with } \begin{pmatrix} y_b \\ z_b \end{pmatrix} = F^T x$$

$$\langle {\color{red}x_d}\,,\, x_d \rangle \stackrel{?}{=} \langle y_b\,,\, y_d \rangle + \langle z_b\,,\, z_d \rangle \quad \text{with } \begin{pmatrix} y_b \\ z_b \end{pmatrix} = F^T {\color{red}x_d}$$

Advection code: discretisation

- Equation

$$\frac{\partial \xi}{\partial t} = u \frac{\partial \xi}{\partial x}$$

- Discretisation: ξ_i^t for $\xi(x,t)$ at timestep t and grid point i
- Time discretisation

$$\frac{\partial \xi_i}{\partial t} = \frac{\xi_i^{t+1} - \xi_i^t}{\Delta t}$$

- Space discretisation (for a constant u)
  - Finite volume (for $\alpha \in [0\,,\,1]$)

$$u\,\Delta t\,\frac{\partial \xi_i^t}{\partial x} = c\,\left[\alpha\,\xi_{i+1}^t + (1-2\alpha)\,\xi_i^t + (\alpha-1)\,\xi_{i-1}^t\right] \quad \text{with} \quad c = \frac{u\,\Delta t}{\Delta x}$$

  - Lax Wendroff

$$u\,\Delta t\,\frac{\partial \xi_i^t}{\partial x} = \left[(c_1+c_2)\,\xi_{i+1}^t - 2c_2\,\xi_i^t + (c_2-c_1)\,\xi_{i-1}^t\right]$$

$$\text{with} \quad c_1 = \frac{1}{2}\frac{u\,\Delta t}{\Delta x} \quad \text{and} \quad c_2 = \frac{1}{2}\left(\frac{u\,\Delta t}{\Delta x}\right)^2$$

Advection code: schemes

- Equation

$$\frac{\partial \xi}{\partial t} = u \frac{\partial \xi}{\partial x}$$

- Finite volume explicit (for $\alpha \in [0\,,\,1]$)

$$\xi_i^{t+1} = c\,\alpha\,\xi_{i+1}^t + [1 + c\,(1 - 2\alpha)]\,\xi_i^t + c(\alpha - 1)\,\xi_{i-1}^t$$

- Finite volume implicit (for $\alpha \in [0\,,\,1]$)

$$-c\,\alpha\,\xi_{i+1}^{t+1} + [1 - c\,(1 - 2\alpha)]\,\xi_i^{t+1} - c(\alpha - 1)\,\xi_{i-1}^{t+1} = \xi_i^t$$

- Lax Wendroff

$$\xi_i^{t+1} = (c_1 + c_2)\,\xi_{i+1}^t + (1 - 2c_2)\,\xi_i^t + (c_2 - c_1)\,\xi_{i-1}^t$$

Advection code: adjoint

Adjoint

- Equation

$$\frac{\partial \xi}{\partial t} = u \frac{\partial \xi}{\partial x}$$

- Adjoint equation

$$\frac{\partial \xi}{\partial t} = -u \frac{\partial \xi}{\partial x}$$

Integration back in time

- Discretisation of the adjoint equation
  - Can we use the discrete model of the linear equation but with a velocity $-u$?

Advection code: adjoint

Adjoint

- Equation

$$\frac{\partial \xi}{\partial t} = u \frac{\partial \xi}{\partial x}$$

- Adjoint equation

$$\frac{\partial \xi}{\partial t} = -u \frac{\partial \xi}{\partial x}$$

Integration back in time

- Discretisation of the adjoint equation
  - Can we use the discrete model of the linear equation but with a velocity $-u$?

Exercice

- Numerical example
  - Domain: $[0, 1]$ divided in 101 points
  - Time: $[0, 1]$ divided in 1000 steps
  - Velocity: $u = 1$
  - $\alpha = 0.75$ for the finite volume methods
- Let's play
  1. Open the file *../ADVECTION/advection.py*
  2. Look the structure of the source code
  3. Run the executable: *python3 advection.py -h*
  4. For #step in 1, 400, 800 run
     python3 advection.py -d -s #step
     python3 advection.py -a -s #step
     python3 advection.py -t -s #step
  5. Try to fix the adjoint code of the advection schemes

Advection code: adjoint (solution)

- Finite volume explicit (linear)

$$\xi_i^{t+1} = c\alpha\xi_{i+1}^t + [1 + c\,(1 - 2\alpha)]\,\xi_i^t + c(\alpha - 1)\,\xi_{i-1}^t$$

- Finite volume explicit (adjoint)

$$\begin{cases} \xi_{i+1}^{*,t} &= \xi_{i+1}^{*,t} + c\alpha\xi_i^{*,t+1} \\ \xi_i^{*,t} &= \xi_i^{*,t} + [1 + c\,(1 - 2\alpha)]\,\xi_i^{*,t+1} \\ \xi_{i-1}^{*,t} &= \xi_{i-1}^{*,t} + c(\alpha - 1)\,\xi_i^{*,t+1} \end{cases}$$

$$\text{or} \quad \begin{cases} \xi_i^{*,t} &= \xi_i^{*,t} + c\alpha\xi_{i-1}^{*,t+1} \\ \xi_i^{*,t} &= \xi_i^{*,t} + [1 + c\,(1 - 2\alpha)]\,\xi_i^{*,t+1} \\ \xi_i^{*,t} &= \xi_i^{*,t} + c(\alpha - 1)\,\xi_{i+1}^{*,t+1} \end{cases}$$

$$\text{or} \quad \xi_i^{*,t} = \xi_i^{*,t} + c(\alpha - 1)\,\xi_{i+1}^{*,t+1} + [1 + c\,(1 - 2\alpha)]\,\xi_i^{*,t+1} + c\alpha\xi_{i-1}^{*,t+1}$$

- Discrete adjoint equation different than discrete linear equation with the velocity $-u$ (Note: $\alpha = 0.5$ is a particular case)

Advection code: adjoint (solution)

- ✗ Finite volume implicit (linear)

$$\mathbf{A}\xi^{t+1} = \xi^{t} \quad \text{with } \xi^{t} \text{ vector of } \xi_i^{t}$$

- ✗ Finite volume explicit (adjoint)

$$\xi^{*,t} = \mathbf{A}^{-1^{T}}\xi^{*,t+1} \quad \text{with } \xi^{*,t} \text{ vector of } \xi_i^{*,t}$$

Advection code: adjoint (solution)

- Lax Wendroff (linear)

$$\xi_i^{t+1} = (c_1 + c_2)\,\xi_{i+1}^t + (1 - 2c_2)\,\xi_i^t + (c_2 - c_1)\,\xi_{i-1}^t$$

- Lax Wendroff (adjoint)

$$\begin{cases} \xi_{i+1}^{*,t} &= \xi_{i+1}^{*,t} + (c_1 + c_2)\,\xi_i^{*,t+1} \\ \xi_i^{*,t} &= \xi_i^{*,t} + (1 - 2c_2)\,\xi_i^{*,t+1} \\ \xi_{i-1}^{*,t} &= \xi_{i-1}^{*,t} + (c_2 - c_1)\,\xi_i^{*,t+1} \end{cases}$$

$$\text{or} \quad \begin{cases} \xi_i^{*,t} &= \xi_i^{*,t} + (c_1 + c_2)\,\xi_{i-1}^{*,t+1} \\ \xi_i^{*,t} &= \xi_i^{*,t} + (1 - 2c_2)\,\xi_i^{*,t+1} \\ \xi_i^{*,t} &= \xi_i^{*,t} + (c_2 - c_1)\,\xi_{i+1}^{*,t+1} \end{cases}$$

$$\text{or} \quad \xi_i^{*,t} = \xi_i^{*,t} + (c_2 - c_1)\,\xi_{i+1}^{*,t+1} + (1 - 2c_2)\,\xi_i^{*,t+1} + (c_1 + c_2)\,\xi_{i-1}^{*,t+1}$$

$$\text{or} \quad \xi_i^{*,t} = \xi_i^{*,t} + (c_1^* + c_2)\,\xi_{i+1}^{*,t+1} + (1 - 2c_2)\,\xi_i^{*,t+1} + (c_2 - c_1^*)\,\xi_{i-1}^{*,t+1}$$

$$\text{whith} \quad c_1^* = -c_1 = \frac{1}{2}\,\frac{-u\,\Delta t}{\Delta x}$$

Adjoint advection equation

The adjoint can be derived using two different approaches

- ✘ Continuous approach: The adjoint equations can be derived from the continuous direct equations;
  - ➾ Derivation follows standard steps, the direct model code can be reused;
  - ➾ Boundary conditions are tricky, accuracy depends on discretization;
- ✘ Discrete approach: The adjoint is derived for the discretized direct equations;
  - ➾ Symmetry can be achieved to the machine precision;
  - ➾ Line by line adjoint coding is tedious, no code reuse;

Note: Discrete adjoint and continuous adjoint can be very different depending on the discretization.

Adjoint advection equation

Consider a linear advection equation for the evolution of $\xi(x,t)$:

$$\frac{\partial \xi}{\partial t} = u \frac{\partial \xi}{\partial x}$$

with initial and boundary conditions given by $\xi(x,0) = \xi_0$ and $\xi(0,t) = \xi(L,t) = 0$. Let's introduce a scalar cost function (for instance measuring the misfit between the model and observations at time T):

$$J = \int_0^L h(x)\xi(x,T)d\xi,$$

We would like to find the sensitivity of J with respect to the initial condition ξ_0, i.e.:

$$\delta J / \delta \xi_0$$

Let's define the scalar product:

$$\int_0^T \langle \xi(x,t), \psi(x,t) \rangle dt = \int_0^T \int_0^L \xi(x,t)\, \psi(x,t)\, dx\, dt$$

Adjoint advection equation

Let's integrate by parts in x and t:

$$\int_0^T \int_0^L \psi \left(\frac{\partial \xi}{\partial t} - u\frac{\partial \xi}{\partial x}\right) dx\, dt = \int_0^T \int_0^L \xi \left(-\frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x}\right) dx\, dt + \int_0^L [\xi(x,T)\psi(x,T) - \xi(x,0)\psi(x,0)]\, dx + \int_0^T [-u\xi(L,t)\psi(L,t) + u\xi(0,t)\psi(0,t)]\, dt$$

Note that the left hand side is zero by definition, hence the right hand side must be equal to 0. Note that the last term on the RHS vanishes. We have:

$$-\frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x} = 0$$

$$\int_0^L [\xi(x,T)\psi(x,T) - \xi(x,0)\psi(x,0)] dx = 0$$

Adjoint advection equation

Consider a special choice $\psi(x, T) = h(x)$:

$$\begin{aligned}
\int_0^L [\xi(x,T)\psi(x,T) - \xi(x,0)\psi(x,0)]dx &= 0 \\
\int_0^L [\xi(x,T)h(x) - \xi(x,0)\psi(x,0)]dx &= 0 \\
J - \int_0^L \xi(x,0)\psi(x,0)dx &= 0
\end{aligned}$$

and

$$J = \int_0^L h(x)\xi(x,T)dx = \int_0^L \psi(x,0)\xi(x,0)dx = \langle \frac{\partial J}{\partial \xi(x,0)}, \xi(x,0) \rangle$$

Adjoint advection equation

Recall the cost function:

$$J = \int_0^L h(x)\xi(x,T)d\xi$$

We found that the gradient of the cost function:

$$\frac{\partial J}{\partial \xi(x,0)} = \psi(x,0)$$

Which can be obtained by solving the adjoint equations:

$$-\frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x} \; = \; 0$$

with $\psi(x,T) = h(x)$ and $\psi(0,t) = \psi(L,t) = 0$. The adjoint equations are integrated backwards in time.