Practical Session: Data Assimilation Experiments using Simple 3D-Var and 4D-Var Systems

Mike Fisher

ECMWF

March 11, 2014

Outline

The Assimilation System

- The assimilation system is the Object-Oriented Prediction System (OOPS).
- OOPS is still under development.
- The aim is to develop a forecast and assimilation system that incorporates a range of simple and complex models.
- Currently, we have the Lorenz95 model and a Quasi-geostrophic channel model.
- OOPS is written in a mixture of C++ and Fortran.
- A key aim of OOPS is to separate the assimilation algorithm from the choice of model.
- This will allow algorithms to be developed with a simple model to be translated easily to more complex models.
- Eventually, we expect OOPS to evolve to become the operational forecast and assimilation system at ECMWF.

Task 1: Make the executables, etc.

- Run the magic script!
  - `/scratch/rd/dai/OOPS/DA_training_course_2014/makeoops`
  - `cd ~/DA_TC_2014`
- Set your PATH
  - `export PATH=$PATH:/tmp/DA_TC_2014/Build/bin`
- View a copy of these notes commands
  - `acroread TC_oops_2014.pdf&`

NB: The file `copypaste` contains a list of all the commands you will be asked to run. You can save yourself a lot of typing by opening `copypaste` in an editor, and copy-pasting to the command line.

The Lorenz-95 Model

- The Lorenz 1995 model is a widely-used low-dimensional dynamical system for data assimilation studies.
- The system is defined by a set of coupled ordinary differential equations:

$$\frac{dx_i}{dt} = -x_{i-2}x_{i-1} + x_{i-1}x + i + 1 + F \quad \text{for } i = 1, 2 \ldots N.$$

- The boundary conditions are cyclic: $x_0 = x_N$, $x_{-1} = x_{N-1}$, etc.
- For a range of values of $F \approx 8$, the system is chaotic, and has similar error growth characteristics to an operational NWP system if we equate one time unit of the Lorenz 1995 system to 5 days of an NWP system.
- For $N = 40$, the system has 13 positive Lyapunov exponents.
- The equations are solved using a fourth-order Runge-Kutta scheme, using $\Delta t = 0.025$

Task 2: Run a cycle of 3D-Var with many observations

- Generate the "truth":
  - `l95_forecast.x l95_truth.xml`
- Generate observations from the truth
  - `l95_makeobs.x l95_makeobs3d.xml`
- Generate the background Forecast:
  - `l95_forecast.x l95_forecast.xml`
- Run a 3D-Var analysis
  - `l95_4dvar.x l95_3dvar.xml`

Task 8: Run a cycle of 3D-Var with many observations

- Plot truth
  - `./l95_plotFields.py --expver=truth --type=fc --step=P1D &`
- Plot background
  - `./l95_plotFields.py --expver=test --type=fc --step=P1D &`
- Plot analysis
  - `./l95_plotFields.py --expver=3dvar --type=an &`

Task 3: Run a cycle of 3D-Var with a single observation

- Generate a single observation from the truth
  - `l95_makeobs.x l95_makeobs3d_single.xml`
- Run a 3D-Var analysis with a single observation
  - `l95_4dvar.x l95_3dvar_single.xml`

Task 3: Run a cycle of 3D-Var with a single observation

- Plot analysis
  - `./l95_plotFields.py --expver=3dvar_single --type=an &`
- Plot analysis increment (2=analysis, 1=background)
  - `./l95_plotDiffs.py --expver2=3dvar_single --type2=an --expver1=test --type1=fc --step1=P1D &`

Task 4: Try changing the background covariance matric

- Edit the file `l95_3dvar_single.xml` and change the parameters of the covariance matrix:
  - `standard_deviation`
  - `length_scale`

Note: you can restore the original xml file by re-running the magic script:
`/scratch/rd/dai/OOPS/DA_training_course_2014/makeoops`

Single Observation Experiments

- Remember the Linear Analysis Equation:

$$x_a = x_b + K(y - Hx_b)$$
$$\text{where } K = BH^T \left(HBH^T + R\right)^{-1}$$

- For a single observation, located at a gridpoint:
  $H = (0, 0, \ldots, 0, 1, 0, \ldots, 0, 0)$
- Hence

$$\begin{aligned} x_a - x_b &= K(y - Hx_b) \\ &= B(0, 0, \ldots, 0, z, 0, \ldots, 0, 0) \end{aligned}$$
$$\text{where } z = \left(HBH^T + R\right)^{-1} (y - Hx_b)$$

- That is, $x_a - x_b \propto$ a column of B

Task 5: Run a cycle of 4D-Var with many observations

- Generate the observations
  - `l95_makeobs.x l95_makeobs4d.xml`
- Run 4D-Var
  - `l95_4dvar.x l95_4dvar.xml`

Task 8: Run a cycle of 4D-Var with many observations

- Plot truth
  - `./l95_plotFields.py --expver=truth --type=fc --step=PT3H &`
- Plot background
  - `./l95_plotFields.py --expver=test --type=fc --step=PT3H &`
- Plot analysis
  - `./l95_plotFields.py --expver=4dvar --type=an &`

Task 6: Run a cycle of 4D-Var with a single observation at the start of the analysis window

- Generate the observations
  - `l95_makeobs.x l95_makeobs4d_single_start.xml`
- Run 4D-Var
  - `l95_4dvar.x l95_4dvar_single_start.xml`

Task 12: Run a cycle of 4D-Var with an observation at the start of the window

- Plot analysis
  - ./l95_plotFields.py --expver=4dvar_single_start --type=an &
- Plot analysis increment (2=analysis, 1=background)
  - ./l95_plotDiffs.py --expver2=4dvar_single_start --type2=an --expver1=test --type1=fc --step1=PT3H &

Task 7: Run a cycle of 4D-Var with a single observation 12h into the analysis window

- Generate the observations
  - `l95_makeobs.x l95_makeobs4d_single_12h.xml`
- Run 4D-Var
  - `l95_4dvar.x l95_4dvar_single_12h.xml`

Task 12: Run a cycle of 4D-Var with a single observation 12h into the analysis window

- Plot analysis
  - `./l95_plotFields.py --expver=4dvar_single_12h --type=an &`
- Plot analysis increment (2=analysis, 1=background)
  - `./l95_plotDiffs.py --expver2=4dvar_single_12h --type2=an --expver1=test --type1=fc --step1=PT3H &`

The Quasi-geostrophic Model

The model you will be using in this session is a simple, two-layer quasi-geostrophic model. The equations are from Fandry and Leslie (1984) (see also Pedlosky, 1979 pp386-393), and describe conservation of potential vorticity:

$$\begin{aligned}\frac{\mathrm{D}q_1}{\mathrm{D}t} &= 0\\ \frac{\mathrm{D}q_2}{\mathrm{D}t} &= 0\end{aligned}$$

where

$$\begin{aligned}q_1 &= \nabla^2\psi_1 - F_1(\psi_1 - \psi_2) + \beta y\\ q_2 &= \nabla^2\psi_2 - F_2(\psi_2 - \psi_1) + \beta y + R_s\end{aligned}$$

ECMWF

The Quasi-Geostrophic Model

- The model domain is a cyclic channel.
- The equations are solved on a 40×20 grid, representing a domain of 12000km$\times$6300km.
- The layer depths in the truth run are 6000m (top) and 4000m (bottom), and a 600s timestep is used.
- The assimilating model layer depths are 5500m and 4500m, and the timestep is 3600s.
- The solution method uses a simple semi-Lagrangian advection of potential vorticity. The advecting winds are determined by inverting the potential vorticity operator.

Task 8: Run a cycle of 3D-Var with many observations

- Generate the "truth":
  - `qg_forecast.x qg_truth.xml`
- Generate observations from the truth
  - `qg_makeobs.x qg_makeobs3d.xml`
- Generate the background Forecast:
  - `qg_forecast.x qg_forecast.xml`
- Run a 3D-Var analysis
  - `qg_4dvar.x qg_3dvar.xml`

Task 8: Run a cycle of 3D-Var with many observations

- Plot truth
  - `./qg_plotFields.py --expver=truth --type=fc --step=P17DT12H &`
- Plot background
  - `./qg_plotFields.py --expver=example --type=fc --step=P1DT12H &`
- Plot analysis
  - `./qg_plotFields.py --expver=3dvar --type=an &`

Task 8: Run a cycle of 3D-Var with many observations

Differences between fieilds can be plotted using qg_plitDiffs.py.
For example

- Plot background error (2=background, 1=truth)
  - ./qg_plotDiffs.py --expver2=example --type2=fc --step2=P1DT12H --expver1=truth --type1=fc --step1=P17DT12H &
- Plot analysis error (2=analysis, 1=truth)
  - ./qg_plotDiffs.py --expver2=3dvar --type2=an --expver1=truth --type1=fc --step1=P17DT12H &
- Plot analysis increment (2=analysis, 1=background)
  - ./qg_plotDiffs.py --expver2=3dvar --type2=an --expver1=example --type1=fc --step1=P1DT12H &

Task 9: Run a cycle of 3D-Var with a single observation

- Generate a single observation from the truth
  - qg_makeobs.x qg_makeobs3d_single.xml
- Run a 3D-Var analysis with a single observation
  - qg_4dvar.x qg_3dvar_single.xml

Task 9: Run a cycle of 3D-Var with a single observation

- Plot analysis
  - `./qg_plotFields.py --expver=3dvar_single --type=an &`
- Plot analysis increment (2=analysis, 1=background)
  - `./qg_plotDiffs.py --expver2=3dvar_single --type2=an --expver1=example --type1=fc --step1=P1DT12H &`

Task 10: Try changing the background covariance matrix

- Edit the file `qg_3dvar_single.xml` and change the parameters of the covariance matrix (e.g. by a factor of two):
  - `standard_deviation`
  - `vertical_correlation`
  - `horizontal_length_scale`
- Re-run 3dVar and plot the analysis increment.

Note: you can restore the original xml file by re-running the magic script:
`/scratch/rd/dai/OOPS/DA_training_course_2014/makeoops`

Task 11: Run a cycle of 4D-Var with many observations

- Generate the observations
  - `qg_makeobs.x qg_makeobs4d.xml`
- Run 4D-Var
  - `qg_4dvar.x qg_4dvar.xml`

Task 11: Run a cycle of 4D-Var with many observations

- Plot truth
  - `./qg_plotFields.py --expver=truth --type=fc --step=P17D &`
- Plot background
  - `./qg_plotFields.py --expver=example --type=fc --step=P1D &`
- Plot analysis
  - `./qg_plotFields.py --expver=4dvar --type=an &`

Task 11: Run a cycle of 4D-Var with many observations

- Plot background error (2=background, 1=truth)
  - `./qg_plotDiffs.py --expver2=example --type2=fc --step2=P1D --expver1=truth --type1=fc --step1=P17D &`
- Plot analysis error (2=analysis, 1=truth)
  - `./qg_plotDiffs.py --expver2=4dvar --type2=an --expver1=truth --type1=fc --step1=P17D &`
- Plot analysis increment (2=analysis, 1=background)
  - `./qg_plotDiffs.py --expver2=4dvar --type2=an --expver1=example --type1=fc --step1=P1D &`

Task 12: Run a cycle of 4D-Var with an observation at the start of the window

- Generate an observation of streamfunction at the start of the window
  - `qg_makeobs.x qg_makeobs4d_single_start.xml`
- Run 4D-Var
  - `qg_4dvar.x qg_4dvar_single_start.xml`

Task 12: Run a cycle of 4D-Var with an observation at the start of the window

- Plot analysis
  - `./qg_plotFields.py --expver=4dvar_single_start --type=an &`
- Plot analysis increment (2=analysis, 1=background)
  - `./qg_plotDiffs.py --expver2=4dvar_single_start --type2=an --expver1=example --type1=fc --step1=P1D &`

Task 13: Run a cycle of 4D-Var with an observation 12h into the window

- Generate an observation of streamfunction 12h into the window
  - `qg_makeobs.x qg_makeobs4d_single_12h.xml`
- Run 4D-Var
  - `qg_4dvar.x qg_4dvar_single_12h.xml`

Task 13: Run a cycle of 4D-Var with an observation 12h into the window

- Plot analysis
  - `./qg_plotFields.py --expver=4dvar_single_12h --type=an &`
- Plot analysis increment (2=analysis, 1=background)
  - `./qg_plotDiffs.py --expver2=4dvar_single_12h --type2=an --expver1=example --type1=fc --step1=P1D &`

Task 14: Try changing the time of the observation

- Edit the file `qg_makeobs4d_single_12h.xml` and change the time of the observation:

```
<Observations>
    <Generate>
        <begin>P12H</begin>
```

- change P12H to P*nn*H where *nn* is in the range 1...23.
- Re-make the observations, re-run 4dVar and plot the analysis increment.

Note: you can restore the original xml file by re-running the magic script:
`/scratch/rd/dai/OOPS/DA_training_course_2014/makeoops`

Have a look at the code...

- Build and view the documentation
  - `cd /tmp/DA_TC_2014/oops/Documents`
  - `make`
  - `firefox html/index.html`