Computer Training Sessions

January, 11, 2018

1 Burgers' equation

Let consider the inviscid Burgers' equation:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\frac{u^2}{2} = 0 \qquad x \in]0, L[, \ t > 0$$

with periodic limit conditions: $u(0,t) = u(L,t)$
and the initial condition is a sinusoid $u(x,0) = \sin 2\pi x$.

We discretize this equation using a finite difference method on a regular grid with steps of Δx in space and Δt in time. Let note u_j^n the approximation of $u(j\Delta x, n\Delta t)$, and let's make use of the Lax-Friedrich scheme:

$$\frac{u_j^{n+1} - \frac{1}{2}(u_{j-1}^n + u_{j+1}^n)}{\Delta t} + \frac{\frac{1}{2}(u_{j+1}^n)^2 - \frac{1}{2}(u_{j-1}^n)^2}{2\Delta x} = 0$$

The explicit form of the equation is then:

$$u_j^{n+1} = \frac{1}{2}(u_{j-1}^n + u_{j+1}^n) + \frac{\Delta t}{4\Delta x}((u_{j-1}^n)^2 - (u_{j+1}^n)^2)$$

2 Presentation of the `python` nootebooks

This practical training is done with python, this programming language has the advantage to propose a simple syntax for matrix handling. The main scripts you will have to play with are available as jupyter notebooks:

- `run_free_model.ipynb` : performs a free run (without assimilation):
- `run_analyse.ipynb`: handles a BLUE analysis experiment;
- `run_EKF.ipynb` : handles the assimilation with an Extended Kalman Filter.
- `test_var.ipynb` : tests the cost function gradient.

- `run_var.ipynb`: handles the variational assimilation itself;

They make use of a set pre-programmed python objects and methods gathered in several files:

- `burgers.py` : three functions: `burgers`, which performs one time step of the Burgers' model; `burger_adj`, which performs one adjoint step; `burger_adj_cont`, which performs one adjoint step using the *continuous* adjoint method for deriving the adjoint.
- `gausscov.py`: creates a gaussian error covariance matrix as well as its inverse and its decomposition (LU or SVD)
- `obsopt.py`: manage the observation opertator
- `analyseKF.py`: performs the BLUE analysis;
- `simvar.py` : computes the cost function and its gradient

In the notebooks, you will be asked to tweak several parameters:

- `sigmao`: specified standard deviation for the observation error;
- `sigmab`: specified standard deviation for the background error;
- `Lb`: Correlation scales for the background error;
- `lprecond` (for variational method only): flag for the use or not of the preconditionning by $\sqrt{B}$ for the minimization;
- `iobstsub`: subsampling in time of the observations;
- `iobsxsub`: subsampling in space of the observations;

3 Kalman filtering

3.1 BLUE analysis

We work with script `run_analyse.py` : we choose a true state, from which we extract a background state and a set of observations. We then perform a BLUE analysis. Initial values are sigmao=0.1 ; sigmab=0.5 ; Lb=0.025 ; iobsxsub=8. The script creates 3 graphic windows: (0) the true state, the background state and the analyzed state. The stars indicate the observed points; (1) the correction brought by the analysis. The circles at $y = -0.5$ indicate the observed points ; (2) the background error covariance matrix.

1. What is the observation operator H ? After the script has been run, check that it is the same as the H operator already coded.

2. Visualize P^f and look at the effect of the error covariances in the analysis.

3. Run the system with the following sets of parameter values, and interpret the results:

   - sigmao=0.1 ; sigmab=0.5 ; Lb=0.000025 ; iobsxsub=8 ;
   - sigmao=0.1 ; sigmab=0.005 ; Lb=0.025 ; iobsxsub=8 ;
   - sigmao=0.001 ; sigmab=0.5 ; Lb=0.025 ; iobsxsub=8 ;
   - sigmao=0.001 ; sigmab=0.5 ; Lb=0.000025 ; iobsxsub=1 ;
   - sigmao=0.1 ; sigmab=0.5 ; Lb=0.000025 ; iobsxsub=1 .

3.2 Kalman filtering

We use now the script `run_EKF.py` : we choose a true state, from which we extract a background state, a set of observations, and the corresponding error covariances. Initial values are sigmao=0.1 ; sigmab=0.5 ; Lb=0.025 ; iobsxsub=4. The script creates 8 graphic windows.

1. The forecast, the analysis and the true state at final time are plotted in window 1. Explain why the analysis and the forecast are similar, and why they are not exactly equal to the true state.

2. Improve this by choosing sigmao=0.01. What is the limit beyond which data assimilation is no more really useful ?

3. A limit in this system may be that the model error is not taken into account. Should it be ?

4. Code a model error. The easiest way consists in applying some inflation to the covariances (one line of code). One can also build a Q matrix, to add it to P, and then to decompose again P.

5. Does the model error help solving the preceding problems ?

4 Variational Data Assimilation

In this case, the data assimilation is done through the minimization of the cost function:

$$J(u^0) = J_b(u^0)+J_o(u^0) = \frac{1}{2}\,(u^0-u^b)^T B^{-1}(u^0-u^b)+\frac{1}{2}\sum_{k=0}^{N}(H_k u^k-u^k_{\text{obs}})^T R^{-1}(H_k u^k-u^k_{\text{obs}})$$

with

$$B(x_i, x_j) = \sigma_B^2 \exp\left(-\frac{(x_i - x_j)^2}{2L_B^2}\right)$$

or

$$B_{i,j} = \sigma_B^2 \exp\left(-\frac{((i-j)\Delta x)^2}{2L_B^2}\right)$$

and

$$R = \sigma_R^2 Id$$

1. Check that the computation of the gradient thanks to the adjoint model is correct (gradient test). Redo the same using `burger_adj_cont` in `simvar`.

2. Perform one data assimilation experiment (using `run_var`) with the continuous adjoint discretized and then the same one using the adjoint of the discrete model.

3. Use or not the $\sqrt{B}$ preconditionning and look at the influence of this choice on the decreasing rate of the cost function using `run_var`.

4. Visualize the structure of the background error covariance matrix by doing an experiment with a single observation (`iobsxsub` $\geq 1+$`nx`$/2$). at the beginning of the time window (`nt`$=0$, `iobstsub`$=1$). Check that the influence of the observation becomes more local when one reduces the correlation length scale L_B. Try with one observation at the end of the time window (`iobstsub`$=$`nt`).

   Then go back to the original settings.

5. Check that the analysis becomes closer to the observation when σ_B becomes larger and that the analysis becomes closer to the background when σ_R becomes larger. Is it always true? Why?

6. Play with the sampling of the observations in order to determine the minimal number of observations required to correct efficiently the background, and the sampling from which it is not necessary to add observation anymore.

   Does it depend on B settings ?

7. From a case that is working reasonably well, try to increase the length of the assimilation window.

8. What would be the possible solution(s) to cope with longer assimilation window? If time permits, try implementing one