BE: Data Assimilation

Selime Gürol
CERFACS
selime.gurol@cerfacs.fr

Emilien Flayac
ISAE
emilien.flayac@isae-supaero.fr

For the following practical we encourage the students to work in pairs. A report is to submit on the LMS platform. **Deadline : 2 March 2026**.

1 Statistical Analysis

The aim of this hands-on is to understand the analysis sensitivity with respect to different settings of the statistical parameters in a simple 1D data assimilation system. Assume that we have noisy observations :

$$\mathbf{y} = \mathbf{H}\mathbf{x}_{true} + \epsilon_o$$

with $\mathbf{H} \in \mathbb{R}^{m \times n}$ being an observation operator and ϵ_o follows a Gaussian distribution with zero mean and observation error covariance matrix ($\mathbf{R}$), i.e. $\epsilon_o \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$. Assume also that we have noisy a priori (background) knowledge :

$$\mathbf{x}^b = \mathbf{x}_{true} + \epsilon_b$$

with $\epsilon_b \sim \mathcal{N}(\mathbf{0}, \mathbf{B})$, with $\mathbf{B}$ being the background error covariance matrix. We assume that the observation and background errors are unbiased. We also assume that the background and observation errors are uncorrelated with each other. Then, the Best Linear Unbiased Estimater (BLUE) is given by

$$\mathbf{x}^a = \mathbf{x}^b + \mathbf{K}(\mathbf{y} - \mathbf{H}\mathbf{x}^b) \tag{1}$$

where

$$\mathbf{K} = \mathbf{B}\mathbf{H}^T(\mathbf{H}\mathbf{B}\mathbf{H}^T + \mathbf{R})^{-1}$$

1.1 Implementation

The python code `statistical_analysis.py` allows solving a simple statistical analysis problem to understand fundamental concepts. In this code, the observation error covariance matrix is defined as a diagonal matrix :

$$\mathbf{R} = \sigma_o^2 \mathbf{I}$$

where σ_o is the observation error standard deviation. The background error covariance matrix can be chosen either as a diagonal matrix, i.e. $\mathbf{B} = \sigma_b^2 \mathbf{I}$ or as a dense matrix where the correlations between the grid points are considered. In the latter case, the elements of the background error covariance matrix $\mathbf{B}$ are given by

$$\mathbf{B}_{ij} = \sigma_b^2 \, \rho_{ij}, \tag{2}$$

where σ_b is the background error standard deviation. The ρ_{ij} are given by a second-order autoregressive (SOAR) correlation function of the distance r_{ij} between the grid-points i and j

$$\rho_{ij} = \left(1 + \frac{r_{ij}}{L}\right) \exp\left(-\frac{r_{ij}}{L}\right), \tag{3}$$

where L is a correlation length scale parameter. The observation operator $\mathbf{H}(\cdot)$ is chosen as a selection operator.

Question 1 *Complete the **statistical_analysis.py** code. Explore the sensitivity of the estimate (analysis) with respect to given parameters :*
(Q1.1) *What is the influence of the observations on the analysis by changing σ_o ;*
(Q1.2) *Choose B as a diagonal matrix. Explain the influence of the B matrix on the analysis*
(i) at the grid points where there is an observation
(ii) at the grid points without observation
(Q1.3) *Choose B as a soar function. Explain the influence of the B matrix on the analysis*
(i) at the grid points where there is an observation
(ii) at the grid points without observation
(Q1.4) *Choose B as a soar function. What is the effect of the*
(i) the correlation length scale parameter on the analysis
(ii) σ_b on the analysis

2 Ensemble Kalman Filter

[1]In order to analyse the different data assimilation algorithms, simulated data ("identical twin") assimilation experiments are conducted with *Python*. In the identical twin experiments, the Lorenz-95 model is used to generate a true model state, $\mathbf{x}_t$, at time t. The background (*a priori*) state and observations are then generated by adding Gaussian noise to fields drawn from the true model state :

$$\begin{aligned} \mathbf{x}_b &= \mathbf{x}_t + \epsilon_b, \quad \epsilon_b \sim \mathcal{N}(\mathbf{0}, \mathbf{B}) \\ \mathbf{y} &= \mathcal{H}(\mathbf{x}_t) + \epsilon_o, \quad \epsilon_o \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \end{aligned}$$

Since the true model state is known, we can compare solutions obtained from different data assimilation algorithm with the truth.

[1] This tutorial is based on CERFACS Data Assimilation tutorial prepared by Anthony Fillion and Mayeul Destouches (CERFACS/UK-MetOffice)

2.1 Dynamical Model : The Lorenz-95 system

The Lorenz-95 system contains K variables : $X_1, X_2, \ldots, X_K$, and is governed by the K equations :

$$\frac{dX_k}{dt} = \underbrace{-X_{k-2}X_{k-1} + X_{k-1}X_{k+1}}_{\text{advection}} - \underbrace{X_k}_{\text{internal dissipation}} + \underbrace{F}_{\text{forcing}} \; . \tag{4}$$

In Eq. (4) the quadratic terms represent the advection that conserves the total energy, the linear term represents the damping through which the energy decreases, and the constant term represents external forcing keeping the total energy away from zero. The K variables may be thought of as *values of some atmospheric quantity in K sectors of a latitude circle.* (Figure 1). For small values of F, all solutions decay to the steady solution $X_1 = .. = X_k = F$, while when F is somewhat larger, most solutions are periodic. For still larger values of $F \approx 8$, the system is chaotic. The system has similar error growth characteristics to an operational Numerical Weather Prediction (NWP) system if one time unit of the Lorenz-95 system is associated with 5 days of an NWP system.

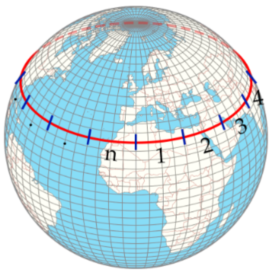

Figure 1 – Example of a latitude circle of the Earth, divided into K equal sized sectors.

For this exercise, $F = 8$, $K = 40$ and the boundary conditions are cyclic, i.e., $x_0 = x_{40}$, $x_{-1} = x_{39}$ and $x_{41} = x_1$. The equations are solved using a fourth-order Runge-Kutta scheme, using $\Delta t = 0.025$ (a 3 hour time step).

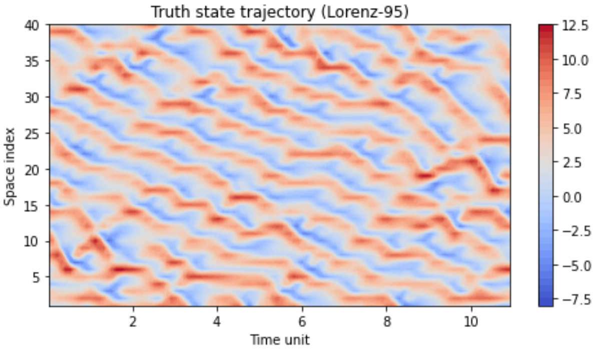

Figure 2 – Space-time diagram of a model run in the configuration of the exercise.

2.2 EnKF Algorithm

The Ensemble Kalman Filter (EnKF) is a variation of the Kalman filter in which the state variable is represented by a sample. It allows to approximate the error covariance matrix used for the Kalman gain computation by sample estimation.

As the Extended Kalman Filter (EKF), the EnKF is able to handle model non-linearity by evaluating the model for each ensemble member while requiring far less model integrations than EKF. Because of its computational performances, the EnKF is a suitable data assimilation method for very high dimensional problems such as geophysical models.

EnKF is easier to implement because it does not need adjoint or tangent linear operators. It also provides a probabilistic representation of the state estimate instead of a pointwise estimation.

```
Stochastic Ensemble Kalman Filter algorithm

1. Initialization
   — Initial system state estimation (background) $\mathbf{x}^b$
   — Initial background error covariance matrix $\mathbf{B}$
   — Sample the initial background ensemble $\mathbf{E}^b = (\mathbf{x}_1^b, ..., \mathbf{x}_i^b, ..., \mathbf{x}_{N_e}^b)$
     with $\mathbf{x}_i^b \sim \mathcal{N}(\mathbf{x}^b, \mathbf{B})$
2. Data assimilation cycles : for $t_k = 1, ..., N_c$,
   (a) Analysis
       — Compute the Kalman Gain $\mathbf{K}^* = \mathbf{B}\mathbf{H}^T(\mathbf{H}\mathbf{B}\mathbf{H}^T + \mathbf{R})^{-1}$
       — Draw an ensemble of perturbed observations $\mathbf{Y} = (\mathbf{y}_1^o, ..., \mathbf{y}_{N_e}^o)$
         with $\mathbf{y}_i^o \sim \mathcal{N}(\mathbf{y}^o, \mathbf{R})$
       — Compute the analysis update for each ensemble member
         $\mathbf{E}^a = \mathbf{E}^b + \mathbf{K}^*(\mathbf{Y} - \mathcal{H}(\mathbf{E}^b))$
   (b) Forecast
       — Compute the ensemble forecast for each member
         $\mathbf{E}^f = (\mathcal{M}(\mathbf{x}_1^a), ..., \mathcal{M}(\mathbf{x}_{N_e}^a)) = \mathcal{M}(\mathbf{E}^a)$
       — The ensemble forecast becomes the background ensemble of the next
         assimilation cycle $\mathbf{E}^b = \mathbf{E}^f$
```

One of the main advantages of the EnKF is that it does not require the tangent linear or the adjoint of the observation operator. Indeed the matrices $\mathbf{B}\mathbf{H}^T$ and $\mathbf{H}\mathbf{B}\mathbf{H}^T$ can be estimated without performing matrices products, directly from the ensemble :

$$
\begin{aligned}
\mathbf{B}\mathbf{H}^T &= \frac{1}{N_e - 1} \sum_{i=1}^{N_e} (\mathbf{x}_i^b - \overline{\mathbf{x}}^b) \left[\mathbf{H}(\mathbf{x}_i^b - \overline{\mathbf{x}}^b)\right]^{\mathrm{T}}, \qquad (5) \\
&\approx \frac{1}{N_e - 1} \sum_{i=1}^{N_e} (\mathbf{x}_i^b - \overline{\mathbf{x}}^b)(\mathcal{H}(\mathbf{x}_i^b) - \overline{\mathcal{H}(\mathbf{x}^b)})^{\mathrm{T}},
\end{aligned}
$$

$$\mathbf{HBH}^T = \frac{1}{N_e - 1}\sum_{i=1}^{N_e}\Big[\mathbf{H}(\mathbf{x}_i^b - \mathbf{x}^b)\Big]\Big[\mathbf{H}(\mathbf{x}_i^b - \mathbf{x}^b)\Big]^{\mathrm{T}}, \tag{6}$$

$$\approx \frac{1}{N_e - 1}\sum_{i=1}^{N_e}(\mathcal{H}(\mathbf{x}_i^b) - \overline{\mathcal{H}(\mathbf{x}^b)})(\mathcal{H}(\mathbf{x}_i^b) - \overline{\mathcal{H}(\mathbf{x}^b)})^{\mathrm{T}},$$

with $\overline{\mathcal{H}(\mathbf{x}^b)} = \frac{1}{N_e}\sum_{i=1}^{N_e}\mathcal{H}(\mathbf{x}_i^b)$.

Note : these approximations can be condensed unto the following matrix notation :

$$\mathbf{BH}^T = \frac{1}{N_e - 1}\mathbf{A}_b\mathbf{A}_o^{\mathrm{T}}$$

$$\mathbf{HBH}^T = \frac{1}{N_e - 1}\mathbf{A}_o\mathbf{A}_o^{\mathrm{T}}$$

where $\mathbf{A}_b$ and $\mathbf{A}_o$ are the (non-normalized) anomaly matrices of the prior ensemble, repectively in the state space or in the observation space.

$$\mathbf{A}_b = \Big(\mathbf{x}_1^b - \overline{\mathbf{x}}^b, ..., \mathbf{x}_i^b - \overline{\mathbf{x}}^b, ..., \mathbf{x}_{N_e} - \overline{\mathbf{x}}^b\Big)$$

$$\mathbf{A}_o = \Big(\mathcal{H}(\mathbf{x}_1^b) - \overline{\mathcal{H}(\mathbf{x}^b)}, ..., \mathcal{H}(\mathbf{x}_i^b) - \overline{\mathcal{H}(\mathbf{x}^b)}, ..., \mathcal{H}(\mathbf{x}_{N_e}^b) - \overline{\mathcal{H}(\mathbf{x}^b)}\Big)$$

Unlike the standard Kalman Filter (or the Extended Kalman Filter), the EnKF does not require to compute expensive matrix products to estimate the analysis error covariance as in :

$$\mathbf{P^a} = (\mathbf{I} - \mathbf{K}^*\mathbf{H})\mathbf{B}(\mathbf{I} - \mathbf{K}^*\mathbf{H})^{\mathrm{T}} + \mathbf{K}^*\mathbf{R}\mathbf{K}^{*\mathrm{T}} = (\mathbf{I} - \mathbf{K}^*\mathbf{H})\mathbf{B},$$

indeed, the error covariance matrices can be simply and cost-effectively estimated by sample estimation. For instance given the analysis ensemble $\mathbf{E}^a = (\mathbf{x}_1^a, ..., \mathbf{x}_i^a, ..., \mathbf{x}_{N_e}^a)^{\mathrm{T}}$ the analysis error covariance matrix can be estimated by :

$$\mathbf{P^a} = \frac{1}{N_e - 1}\sum_{i=1}^{N_e}(\mathbf{x}_i^a - \overline{\mathbf{x}}^a)(\mathbf{x}_i^a - \overline{\mathbf{x}}^a)^{\mathrm{T}}. \tag{7}$$

3 Exercise : Code the stochastic EnKF

1. Complete the EnKF code in file `/Codes/EnsembleMethods/EnKF_to_complete.py`. Only the middle of the code is missing.

Question 2 *Check that the EnKF is working correctly : you should have an RMSE lower than* 1.0 *on average, and the analyses should have reduced errors compared to the backgrounds. Provide plots for one example.*

3.1 First Trials

Set the EnKF hyper-parameters to `Ne = 100`, `sigmaB = 2.0`, `sigmaR = 1.0`, `exp_length = 10` time units and `Ndt = 4` (corresponding to an assimilation cycle every 12 h). We first test the performance of the EnKF in a context without localization and without inflation, over a 50 day period (10 time units). Hereafter, do not hesitate to frequently change the seed of the random generator (initially 1234).

Note that the $\mathbf{B}$ matrix defined as $\sigma_B \mathbf{I}$ in the code is only used to initialize the ensemble for the first cycle. The following cycles will use ensemble covariances, which are more realistic.

Question 3 *Test sensitivity to the values of* $\mathbf{R}$. *What do you expect for small values of the observation error* σ_R *? What happens for large* σ_R *?*

Question 4 *Set the values of observation and background errors to their default values and try ensemble sizes smaller than 100. What is the minimum ensemble size* N_{min} *that avoids divergence of the filter (i.e. uncontrolled evolution of error) ?*

Question 5 *Is the EnKF is subject to ensemble collapse ? Look at the Figure "Evolution of the local ensemble spread" figure for different values of N. What happens when* $N = 10$ *and* 100 *?*

3.2 Inflation

To cope with ensemble collapse, a straightforward solution called inflation consists in multiplying the analysis error covariance matrix by a factor $\lambda^2 > 1$:

$$\mathbf{P^a} \leftarrow \lambda^2 \times \mathbf{P^a},$$

which can be simply implemented in the EnKF algorithm by rescaling each ensemble member right after the analysis :

$$\mathbf{x}_i^a \leftarrow \overline{\mathbf{x}^a} + \lambda(\mathbf{x}_i^a - \overline{\mathbf{x}^a}) \tag{8}$$

Question 6 *Choose 'apply inflation' parameter as 'True' in the code. Can inflation improve this minimum viable ensemble size on this case ?*

3.3 Bonus : Covariance localization

In practice, ensemble covariances in the computation of the Kalman gain should be localized :

$$\mathbf{K}^* = \left(\mathbf{L}_{\mathrm{xy}} \circ \left(\mathbf{B}\mathbf{H}^T\right)\right) \left(\mathbf{L}_{\mathrm{yy}} \circ \left(\mathbf{H}\mathbf{B}\mathbf{H}^T\right) + \mathbf{R}\right)^{-1} \tag{9}$$

This is done in the code if you set `apply_localization = True`. The localization length is uniform over the domain, and set to 10 grid points by default. Set `apply_inflation = False` for this section.

5. For an ensemble size of 40, try a few values of localization radius. What is the optimal one for this ensemble size (plus or minus 5 grid points) ? Note that the localization functions accept a boolean parameter plot that can help you visualize how localization is affecting the sample covariance matrices.

6. Can localization make EnKF works with less than N_{min} ensemble members ? Is it more efficient than inflation ?

7. How does the optimal localization radius vary as the ensemble size increases ? Decreases ? (Testing only 3 ensemble sizes should be enough)

3.4 Bonus : Both inflation and localization

8. Can inflation and localization together allow the EnKF to work with less than 10 members ? Is it robust to changes in the random seed ?