Inverse methods and data assimilation

1

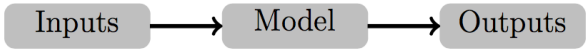
Introduction, and simple but essential exemple

Chapter outline

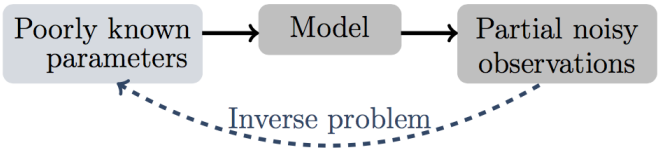
1.1 What is an inverse problem

1.1.1 Direct or inverse

Direct problem is close to "modelling". In that case, we have (or we are developping) a (mathematcial, numerical) model of our system. This model has input parameters, which we know fairly well. We are interested in the outputs produced by the model. The following diagram describes a direct problem :



Inverse problem do the reverse! In that case we also have a model (which should be fairly accurate), but now the input parameters are not well known (because of measuring errors, or because we cannot measure them directly, e.g., if they are di cult to access). However, some outputs corresponding to the unknown inputs can be measured. Inverse problem aims to combine the model equations and the outputs to infer the poorly known inputs, according to the diagram :



Famous examples are :

	Problem	Partial observation	Input parameters to be infered
	Sherlock Holmes	Observable consequences and clues	What *really* happened !
	Medical imaging	External measurements on the body surface	Tissue properties and diagnosis (bone, liquid, tumor, ...)
	Numerical Weather Prediction	Physical observations of the atmosphere and ocean	Initial condition to produce a weather forecast

1.1.2 Inverse methods and data assimilation

Inverse methods allow to combine optimally all sources of information available about a given (physical, biological, chemical, ...) system:

- mathematical equations (physical laws or the biological processes, ...);
- observations (measures of real experiments);
- error statistics (observation errors, model errors, ...).

These sources of information are usually heterogeneous: different nature, varying quality and quantity.

In geosciences, inverse methods are often called *data assimilation*. Historically, the idea was to estimate the initial state of the atmosphere, in order to produce weather forecasts. Today, it has many applications, not only initial state estimation (parameter estimation, physical law parameterisation, numerical parameter estimation, unknown forcing sources estimation...). It is also used in many application domains, not only weather forecasting (oceanography, oil drill, seismology, energy, medicine, biology, glaciology, agronomy, construction industry, ...).

These problems are not straightforward. The most commonly encountered issues are:

- complex systems (coupled systems, non linear laws, poorly known phenomenons, ...);

- indirect/sparse (in space/time)/noisy observations;
- ill-posed problem (underdetermined - not enough observations, overdetermined - redundant and contradictory observations).

1.2 A simple example

This example will introduce the BLUE (Best Linear Unbiased Estimation), which is a statistical estimation method, to solve a very simple least square problem.
The example writes this way: suppose that we have two observations $y_1 = 1$ and $y_2 = 2$ of un unknown quantity x. We aim to estimate x as a function of the observations.

1.2.1 Naive method

We formulate the problem as a simple least square problem:

$$\text{Find } x \text{ as the minimizer of } (x-1)^2 + (x-2)^2$$

The solution is easilly found and gives the estimator $\hat{x} = 3/2$.

This solution presents several shortcomings:

1. The result is sensitive to unit changes. Imagine that we now have $y_1 = 1$ a measure of x and $y_2 = 4$ a measure of $2x$ (in another unit), then we should minimize $(x-1)^2 + (2x-4)^2$, and the estimator now is $\hat{x} = 9/5$. $\rightarrow$ Some kind of normalisation is required.
2. The result is insensitive to the accuracy of the measurement instrument, the estimator is the same even if y_1 is more accurate than y_2.

1.2.2 Statistical formalisation

Let us denote $y_i = x + e_i$ for $i = 1, 2$. The observation errors e_i are assumed to be:

- unbiased (with 0 mean): $\mathrm{E}(e_i) = 0$ for $i = 1, 2$
- with known variances: $\mathrm{Var}(e_i) = \sigma_i^2$ for $i = 1, 2$
- uncorrelated: $\mathrm{Cov}(e_1, e_2) = 0$ (or $\mathrm{E}(e_1 e_2) = 0$)

We now look for a linear estimation, which is unbiased and with minimal variance. It is called the BLUE (Best Linear Unbiased Estimator):

$$\hat{x} = \alpha_1 y_1 + \alpha_2 y_2$$

The unbiased hypothesis is $\mathrm{E}(\hat{x} - x) = 0$, it follows easily that $\alpha_1 + \alpha_2 = 1$. We can then compute the variance of $\hat{x}$:

$$\mathrm{Var}(\hat{x}) = \alpha_1^2 \sigma_1^2 + (1-\alpha_1)^2 \sigma_2^2 \tag{1.1}$$

We can that compute the estimation $\hat{x}$, which minimises the variance, and then compute $\mathrm{Var}(\hat{x})$. We find:

$$\alpha_1 = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \qquad \hat{x} = \frac{\sigma_2^2 y_1 + \sigma_1^2 y_2}{\sigma_1^2 + \sigma_2^2} \qquad \mathrm{Var}(\hat{x}) = \left(\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}\right)^{-1} \tag{1.2}$$

1.2.3 Remarks

First, we can show that we obtain the same result if we look for the minimizer of the cost function

$$J(x) = \frac{1}{2}\left(\frac{(x-y_1)^2}{\sigma_1^2} + \frac{(x-y_2)^2}{\sigma_2^2}\right) \tag{1.3}$$

This is still a least square problem, but the BLUE enables us to rationalise the choice of a normalisation in the cost function J. This normalisation solves both issues of sensitivity to units and measure accuracy.
Moreover, the concavity of J gives a measure of the estimator accuracy, as we have:

$$J''(x) = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} = \frac{1}{\mathrm{Var}(\hat{x})} \tag{1.4}$$

If we consider that $y_1 = x^b$ is a first estimation of x (which we will call the *background*) and $y_2 = y$ is an independant observation, then we can rewrite $\hat{x}$ as

$$\hat{x} = x^b + \frac{{\sigma_b}^2}{\sigma^2 + {\sigma_b}^2}(y - x^b)$$

The value $y - x^b$ is called *innovation*, it contains the new information brought by y with respect to x^b. The equation above can be summed up as

$$\hat{x} = \text{ background} + \text{something} \times \text{ innovation}$$

.

1.2.4 Data assimilation methods

There exist two families of data assimilation (DA) methods: statistical methods (we compute the BLUE directly) and variational methods (minimisation of the cost function J). Of course, in 1D as in the example, these methods seem similar, but we will see that in multi dimensional cases (multi = 10^6) they are pretty different:

- (Bayesian) probabilistic methods: we treat x as a random variable, and try to compute the probability density of the state knowing the set of observations. It sometimes ends to estimate $\hat{x}$ as $\hat{x}$ = background + *gain matrix* $\times$ innovation, so we have to compute the gain matrix. It is a huge matrix involving covariances for background and observations (as everything is vectorial, covariances are -very large- matrices).
- variational methods: we look for $\hat{x}$ as $J(\hat{x}) = \min_x J(x)$. As x lives in a very high dimensional space, minimisation is not easy, and many methods exist (Newton, quasi Newton, other gradient methods, stochastic methods like simulated annealing, ...).

Common advantages and drawbacks of these methods:

- under some (restrictive) hypotheses they lead to the same result;
- under the same (restrictive) hypotheses they are optimal (ie. both lead to the optimal solution of the least squares problem);
- common issues:
  - in case of non-linearities the optimality and equivalence between the methods are lost;
  - curse of dimensionality (huge matrix sizes, impossible to compute or even store them in memory);
  - errors statistics are required but unknown.

1.3 Exercises

Complete the calculations of this chapter (§1.2):

1. Prove that $\alpha_1 + \alpha_2 = 1$ using $\mathrm{E}(\hat{x} - x) = 0$.
2. Prove formula (1.1) for the variance.
3. Prove formula (1.2)
4. Prove that $\hat{x}$ is the minimizer of (1.3).
5. Prove (1.4).

1.4 Solutions

Complete the calculations of this chapter (§1.2):

1. Prove that $\alpha_1 + \alpha_2 = 1$ using $\mathrm{E}(\hat{x} - x) = 0$.
   Sol. $\hat{x} - x = \alpha_1 y_1 + \alpha_2 y_2 - x = (\alpha_1 + \alpha_2 - 1)x + \alpha_1 e_1 + \alpha_2 e_2$ then

$$\begin{aligned}\mathrm{E}(\hat{x} - x) &= (\alpha_1 + \alpha_2 - 1)\mathrm{E}(x) + \alpha_1 \mathrm{E}(e_1) + \alpha_2 \mathrm{E}(e_2)) \\ &= (\alpha_1 + \alpha_2 - 1)\mathrm{E}(x) = 0\end{aligned}$$

   We then have $\alpha_1 + \alpha_2 = 1$, or $\alpha_2 = 1 - \alpha_1$.

2. Prove formula (1.1) for the variance.
   Sol.

$$\begin{aligned}\mathrm{Var}(\hat{x}) &= \mathrm{E}((\hat{x} - x)^2) = \mathrm{E}((\alpha_1 e_1 + \alpha_2 e_2)^2) \\ &= \alpha_1^2 \mathrm{E}(e_1^2) + 2\alpha_1\alpha_2 \mathrm{E}(e_1 e_2) + \alpha_2^2 \mathrm{E}(e_2^2) \\ &= \alpha_1^2 \sigma_1^2 + \alpha_2^2 \sigma_2^2 \\ &= \alpha_1^2 \sigma_1^2 + (1 - \alpha_1)^2 \sigma_2^2\end{aligned}$$

3. Prove formula (1.2)
   Sol. $\mathrm{Var}(\hat{x})$ is a quadratic function of α_1. To look for a minimum we look for the critical point (zero of the derivative), and we find the expression of α_1. Then

$$\hat{x} = \frac{\frac{1}{\sigma_1^2} y_1 + \frac{1}{\sigma_2^2} y_2}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}}$$

   Remaining eqs are straightforward.
4. Prove that $\hat{x}$ is the minimizer of (1.3).
   Sol. Straightforward, compute the derivative of J.
5. Prove (1.4).
   Sol. Straightforward.

2

Elements of probability and statistics

Chapter outline

2.1 Events and probability

Random experiment. A random experiment is mathematically described by:

- the set Ω of all possible outcomes of an experiment, the result of which cannot be perfectly anticipated;
- the subsets of Ω, called events;
- a probability function, $\mathbb{P}$: a numerical expression of a state of knowledge. P is such as, for any disjoint events A and B:

$$0 \leq \mathbb{P}(A) \leq 1,$$
$$\mathbb{P}(\Omega) = 1,$$
$$\mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B)$$

Independance. Two events A and B are said to be independant if:

$$\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B)$$

Conditional probability. When the two events A and B are not independent, knowing that B has occurred changes our state of knowledge on A. This reads:

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}$$

2.2 Real random variables

The outcome of a random experiment is called a random variable. A random variable can be either an integer number (e.g., a die cast) or a real number (e.g., the lifetime of a electric light bulb), in this case we call them *discrete* or *continuous* random variables. We will denote by X a random variable and x its realization. Bold symbol will be associated to vectors.

2.2.1 Discrete variable

X is defined on a finite set $\mathcal{X}$, e.g. $\mathcal{X} = \{1, ..., N\}$ or $\mathcal{X} = \mathbb{N}$, ie the ensemble of values of X is discrete: $(x_i)_{i \in \mathcal{X}}$ and the law of X is given by the probabilities

$$\mathbb{P}(X = x_i) = p_i \in [0, 1], \text{ with } \sum p_i = 1$$

Cumulative distribution function.

$$F_X(x) = \mathbb{P}(X \leq x)$$

It is an increasing function, such that $F(-\infty) = 0$, $F(+\infty) = 1$ and we have

$$P(X \in]a, b]) = F_X(b) - F_X(a)$$

For discrete random variables, F_X is a piecewise-constant function.

Conditional probability. Let X and Y be two random variables taking values in finite sets $\mathcal{X}$ and $\mathcal{Y}$, we define the conditional law of X knowing Y by :

$$\mathbb{P}(X = x|Y = y) = \frac{\mathbb{P}(X = x, Y = y)}{\mathbb{P}(Y = y)} = \frac{\mathbb{P}(X = x, Y = y)}{\sum_{x' \in \mathcal{X}} \mathbb{P}(X = x', Y = y)}$$

Independance Two random variables X and Y are *independant* when

$$\mathbb{P}(X = x, Y = y) = \mathbb{P}(X = x)\mathbb{P}(Y = y)$$

then, we have :

$$\mathbb{P}(X = x|Y = y) = \mathbb{P}(X = x)$$

Bayes theorem From the definition of the conditional probability, we can deduce the Bayes formula :

$$\mathbb{P}(X = x|Y = y) = \frac{\mathbb{P}(Y = y|X = x)\mathbb{P}(X = x)}{\mathbb{P}(Y = y)}$$

2.2.2 Continuous variables

Here, the set $\mathcal{X}$ is no longer supposed to be finite, e.g. $\mathcal{X} = \mathbb{R}$. X is said to have a *probability density function* (pdf) if and only if there exists a unique function $p_X : \mathbb{R} \to \mathbb{R}_+$ such that

$$\forall x \in \mathbb{R}, F_X(x) = \int_{-\infty}^{x} p_X(x)\, dx$$

In this case, p_X is continuous, differentiable and we have:

$$\int_{\mathcal{X}} p_X(x)\ dx = 1 \qquad \text{and} \qquad \mathbb{P}(X \in]a, b]) = \int_a^b p_X(x)\, dx.$$

Joint, marginal and conditional pdf Let X and Y be two random variables, living in $\mathcal{X}$ and $\mathcal{Y}$.

- $p_{X,Y}(x, y)$ is called the *joint density*.
- the (density of) the *marginal law* of X is

$$p_X(x) = \int_{\mathcal{Y}} p_{X,Y}(x, y)\ dy$$

- the conditional density of X knowing $Y = y$ denoted by $p_{X|Y}(x|y)$ is defined by :

$$p_{X|Y}(x|y) = \frac{p_{X,Y}(x, y)}{p_Y(y)}$$

Bayes theorem The Bayes theorem writes :

$$p_{X|Y}(x|y) = \frac{p_{Y|X}(y|x)\ p_X(x)}{p_Y(y)} = \frac{p_{Y|X}(y|x)\ p_X(x)}{\int_{\mathcal{X}} p_{Y|X}(y|x)\ p_X(x)\ dx}$$

2.2.3 Expectation, variance and covariance

(Note: the mean is also called expected value or expectation, mathematical expectation, EV, or first moment)

A pdf is rarely known completely. Generally, only some properties are determined and handled. The two main properties are the expectation and the variance.

expectation The expectation of a random variable X is

$$\mathrm{E}(X) =< X >= \begin{cases} \displaystyle\int_{-\infty}^{+\infty} x\ p_X(x)\ dx & \text{if } X \text{ is continuous} \\ \displaystyle\sum_{i \in I} x_i\ p_i & \text{if } X \text{ is discrete} \end{cases}$$

variance The variance (also called second central moment) is

$$\mathrm{Var}(X) = \mathrm{E}\left([X - \mathrm{E}(X)]^2\right) = \begin{cases} \displaystyle\int_{-\infty}^{+\infty} (x - \mathrm{E}(x))^2\ p_X(x)\ dx & \text{if } X \text{ is continuous} \\ \displaystyle\sum_{i \in I} (x_i - E(x))^2\ p_i & \text{if } X \text{ is discrete} \end{cases}$$

We also have

$$\mathrm{Var}(X) = \mathrm{E}(X^2) - \mathrm{E}(X)^2 = \sigma^2(X)$$

The standard deviation is the square root $\sigma(X)$ of the variance.

covariance Let X and Y be two random variables, we define their covariance as

$$\mathrm{Cov}(X, Y) = \mathrm{E}\left[(X - \mathrm{E}(X)).(Y - \mathrm{E}(Y))\right]$$

and their linear correlation coefficient

$$\rho(X, Y) = \frac{\mathrm{Cov}(X, Y)}{\sigma_X \sigma_Y}$$

If X and Y are independent then $\mathrm{Cov}(X, Y) = 0$, but the converse is false in general.

2.3 Random vectors

Real random vectors (denoted by bold symbol) are vectors which components are real random variables: $\mathbf{X} = (X_1, ..., X_N)$ where X_i is a real random variable. The pdf of a vector is the joint pdf of its real components.

The expectation vector is the vector formed with the expected values of the real components:

$$\mathrm{E}(\mathbf{X}) = (\mathrm{E}(X_i))_{i=1..N}$$

The second moment of the distribution is the covariance matrix. If $\mathbf{X}$ denotes the random vector, the covariance matrix is defined by

$$\mathrm{Cov}(\mathbf{X}) = \mathrm{E}\left[(\mathbf{X} - \mathrm{E}(\mathbf{X}))\,(\mathbf{X} - \mathrm{E}(\mathbf{X}))^T\right].$$

A covariance matrix is symmetric, positive. The terms on the diagonal are the variances of the vector components. The non-diagonal terms are covariances.

2.4 Normal (Gaussian) distribution

2.4.1 The Gaussian univariate distribution.

The random variable X has a Gaussian (or normal) distribution with parameters μ and σ^2, which is noted $X \sim \mathcal{N}(X; \mu, \sigma^2)$, when

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right].$$

The Gaussian distribution possesses some very nice properties, in particular:

- it is a natural distribution for signal noises;
- the parameters μ and σ^2 of the distribution are the expectation and the variance, respectively;
- if $x_1 \sim \mathcal{N}(x_1; \mu_1, \sigma_1^2)$ and $x_2 \sim \mathcal{N}(x_2; \mu_2, \sigma_2^2)$ are two independent variables, then $x_1 + x_2$ is also Gaussian and $z = x_1 + x_2 \sim \mathcal{N}(z; \mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$;
- if a is a real number and $x \sim \mathcal{N}(x; \mu, \sigma^2)$, then $z = ax \sim \mathcal{N}(z; a\mu, a^2\sigma^2)$.

2.4.2 The Gaussian multivariate distribution.

The random vector $\mathbf{X}$ of size n has a Gaussian (or normal) distribution with parameters μ and $\mathbf{P}$, which is noted $\mathbf{X} \sim \mathcal{N}(\mathbf{X}; \mu, \mathbf{P})$, when

$$p_{\mathbf{X}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{n/2}|\mathbf{P}|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x} - \mu)^T \mathbf{P}^{-1} (\boldsymbol{x} - \mu)\right].$$

μ and $\mathbf{P}$ are the expectation and the covariance matrix of $\mathbf{X}$, respectively. $|\mathbf{P}|$ denotes the determinant of $\mathbf{P}$. The component of $\mathbf{X}$ are said to be jointly Gaussian.

We have the following results:

$$\mathbf{X} \sim \mathcal{N}(\mathbf{X}; \mu, \mathbf{P}) \quad \Rightarrow \quad \mathbf{Z} = (\mathbf{A}\mathbf{X} + \mathbf{Y}) \sim \mathcal{N}(\mathbf{Z}; \mathbf{A}\mu + \mathbf{Y}, \mathbf{A}\mathbf{P}\mathbf{A}^T)$$

$$\mathbf{X} \sim \mathcal{N}(\mathbf{X}; \mu, \mathbf{P}) \quad \Rightarrow \quad \mathbf{Z} = \mathbf{P}^{-1/2}(\mathbf{X} - \mu) \sim \mathcal{N}(\mathbf{Z}; \mathbf{0}, \mathbf{I})$$

From joint to marginal and conditional distributions

Let the vector $\mathbf{Z} = [\mathbf{X}^T \mathbf{Y}^T]^T$ be normally distributed according to

$$\mathbf{Z} = \begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{Z}; \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^T & \mathbf{B} \end{bmatrix}\right)$$

where $\mathbf{C}$ is the (non-symmetric) cross-covariance matrix between $\mathbf{X}$ and $\mathbf{Y}$ which has as many rows as the size of $\boldsymbol{x}$ and as many columns as the size of $\boldsymbol{y}$. then the marginal distributions are:

$$\mathbf{X} \sim \mathcal{N}(\mathbf{X}; \mathbf{a}, \mathbf{A})$$
$$\mathbf{Y} \sim \mathcal{N}(\mathbf{Y}; \mathbf{b}, \mathbf{B})$$

and the conditional distributions are:

$$\mathbf{X}|\mathbf{Y} \sim \mathcal{N}(\mathbf{X}; \mathbf{a} + \mathbf{C}\mathbf{B}^{-1}(\mathbf{Y} - \mathbf{b}), \mathbf{A} - \mathbf{C}\mathbf{B}^{-1}\mathbf{C}^T) \tag{2.1}$$

$$\mathbf{Y}|\mathbf{X} \sim \mathcal{N}(\mathbf{Y}; \mathbf{b} + \mathbf{C}^T\mathbf{A}^{-1}(\mathbf{X} - \mathbf{a}), \mathbf{B} - \mathbf{C}^T\mathbf{A}^{-1}\mathbf{C}) \tag{2.2}$$

From conditional to joint and marginal distributions

Let $\mathbf{X}$ and $\mathbf{Y}$ have the Gaussian distributions :

$$\mathbf{X} \sim \mathcal{N}(\mathbf{X}; \boldsymbol{m}, \mathbf{P}) \qquad \mathbf{Y}|\mathbf{X} \sim \mathcal{N}(\mathbf{Y}; \mathbf{H}\mathbf{X}, \mathbf{R})$$

the the joint distribution is :

$$\mathbf{Z} = \begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{Z}; \begin{bmatrix} \boldsymbol{m} \\ \mathbf{H}\boldsymbol{m} \end{bmatrix}, \begin{bmatrix} \mathbf{P} & \mathbf{P}\mathbf{H}^T \\ \mathbf{H}\mathbf{P} & \mathbf{H}\mathbf{P}\mathbf{H} + \mathbf{R} \end{bmatrix}\right)$$

and the marginal is :

$$\mathbf{Y} \sim \mathcal{N}(\mathbf{Y}; \mathbf{H}\boldsymbol{m}, \mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R}) \tag{2.3}$$

Product of two Gaussian functions

the product of two Gaussian functions is another Gaussian function (although no longer normalized). in particular,

$$\mathcal{N}(\mathbf{X}; \mathbf{a}, \mathbf{A})\, \mathcal{N}(\mathbf{X}; \mathbf{b}, \mathbf{B}) \propto \mathcal{N}(\mathbf{X}; \mathbf{c}, \mathbf{C}) \tag{2.4}$$

where

$$\mathbf{C} = (\mathbf{A}^{-1} + \mathbf{B}^{-1})^{-1}$$
$$\mathbf{c} = \mathbf{C}\mathbf{A}^{-1}\mathbf{a} + \mathbf{C}\mathbf{B}^{-1}\mathbf{b}$$

2.5 (Hidden) Markov chain

2.5.1 Markov chain

Let un consider a state process $\mathbf{X} = \{\mathbf{X}_k\}_{k\geq 0}$ taking its values in a continuous state space $\mathcal{X}$. We will use the following notation :

$$\mathbf{X}_{0:K} = [\mathbf{X}_0, \dots, \mathbf{X}_K]$$

The law of the process $\mathbf{X} = \{\mathbf{X}_k\}_{k\geq 0}$ is defined through the laws $p_{\mathbf{X}_{0:k}}(\boldsymbol{x}_{0:k})$ for all $k \geq 0$. We have the following general result :

$$\begin{aligned} p_{\mathbf{X}_{0:k}}(\boldsymbol{x}_{0:k}) = \quad & p_{\mathbf{X}_0}(\boldsymbol{x}_0) \times p_{\mathbf{X}_1|\mathbf{X}_0}(\boldsymbol{x}_1|\boldsymbol{x}_0) \\ & \times p_{\mathbf{X}_2|\mathbf{X}_{0:1}}(\boldsymbol{x}_2|\boldsymbol{x}_{0:1}) \dots p_{\mathbf{X}_k|\mathbf{X}_{0:k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}) \end{aligned} \tag{2.5}$$

The process is said to be Markovian when

$$p_{\mathbf{X}_k|\mathbf{X}_{0:k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}) = p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$$

meaning that the probability of each subsequent state depends only on what was the previous state. *Knowing the present, we can forget the past to predict the future.*

We then have

$$p_{\mathbf{X}_{0:k}}(\boldsymbol{x}_{0:k}) = p_{\mathbf{X}_0}(\boldsymbol{x}_0) \prod_{l=1}^{k} p_{\mathbf{X}_l|\mathbf{X}_{l-1}}(\boldsymbol{x}_l|\boldsymbol{x}_{l-1})$$

A Markov process is then entirely defined with its initial density $p_{\mathbf{X}_0}(\boldsymbol{x}_0)$ and the transition density $p_{\mathbf{X}_l|\mathbf{X}_{l-1}}(\boldsymbol{x}_l|\boldsymbol{x}_{l-1})$.

2.5.2 Hidden Markov chain

Let us consider a couple of processes : the state process $\mathbf{X} = \{\mathbf{X}_k\}_{k\geq 0}$, taking its value in the state space $\mathcal{X}$; and the observation process $\mathbf{Y} = \{\mathbf{Y}_k\}_{k\geq 1}$, taking its value in the observation space $\mathcal{Y}$. The couple $(\mathbf{X}, \mathbf{Y})$ is said to be a hidden Markov chain if

- $\mathbf{X}$ is a Markov chain, of initial law $p_{\mathbf{X}_0}(x_0)$ and transition law $p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(x_k|x_{k-1})$.
- knowing the state, the observation are independent (property of *conditional indenpendance*), meaning

$$p_{\mathbf{Y}_{1:K}|\mathbf{X}_{0:K}}(\boldsymbol{y}_{1:K}|\boldsymbol{x}_{0:K}) = \prod_{k=1}^{K} p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)$$

$p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)$ is called the *likelihood*.

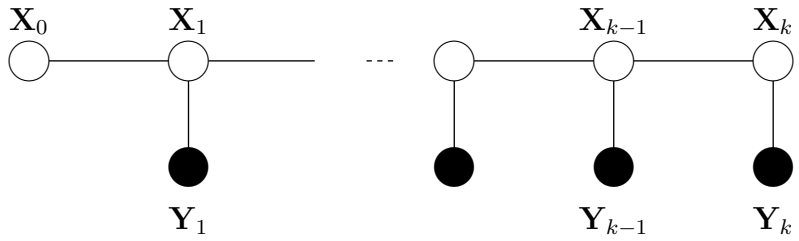


Figure 2.1: Hidden Markov chain. $\mathbf{X}$'s are states. $\mathbf{Y}$'s are observations.

Hence, a hidden Markov process in defined though :

- the state space $\mathcal{X}$
- the observation space $\mathcal{Y}$
- the initial density $p_{\mathbf{X}_0}(\boldsymbol{x}_0)$
- the transition density $p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$
- the likelihood $p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)$

We can write :

$$p_{\mathbf{X}_{0:K},\mathbf{Y}_{1:K}}(\boldsymbol{x}_{0:K},\boldsymbol{y}_{1:K}) = p_{\mathbf{X}_0}(\boldsymbol{x}_0)\prod_{k=1}^{K} p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)\; p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$$

3

Ingredients of data assimilation

Chapter outline

3.1 State, evolution and model error

Let be the state of the system we aim at estimating be modelled by a variable $\boldsymbol{x}$, living in the *state space* $\mathcal{X}$

Here are considered dynamical models, i.e. models that compute the time evolution of the simulated state. Let $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k+1}$ be the vectors modelling the state at two consecutive times, k being a time index. They are related by a causality link:

$$\boldsymbol{x}_{k+1} = \mathcal{M}_{k\to k+1}(\boldsymbol{x}_k) + \epsilon^m_{k+1} \tag{3.1}$$

For example, in oceanography we could have

$$\boldsymbol{x}_k = (U_{1,k}, U_{2,k}, ..., U_{n,k}, V_{1,k}, V_{2,k}, ..., V_{n,k}, T_{1,k}, ..., T_{n,k}, S_{1,k}, ..., S_{n,k})$$

where $U_{i,k}, V_{i,k}$ are the horizontal velocity of the oceanic currents on every point i of the spatial grid at time k, and $T_{i,k}, S_{i,k}$ are the values of temperature and salinity on these points at time k. The model $\mathcal{M}_{k\to k+1}$ would then be one time-step of the resolution of the Navier-Stokes equations (+ temperature and salinity diffusion and transport equations).

The *model error* ϵ^m_k accounts for the errors in the numerical model (e.g., misrepresentation of physical processes) and for the errors due to the discretization. The actual value of this error is not known, it is thus considered as a random variable. we suppose that this error is of zero mean $\mathrm{E}(\epsilon^m_k) = 0$, and has a covariance matrix $\mathrm{Cov}(\epsilon^m_k) = \mathbf{Q}_k$. Finally, the model errors are independent in time, meaning $\mathrm{Cov}(\epsilon^m_k \epsilon^m_l) = 0 \ \forall k \neq l$.

Finally, to calculate the evolution of the state vector by means of equation (3.1), it is essential to have an estimate $\boldsymbol{x}_b$, called *background* or *a priori estimate*, of the initial state $\boldsymbol{x}_0$. The background error ϵ^b (difference between the background and the true value of the initial state) is supposed to be of zero mean $\mathrm{E}(\epsilon^b) = 0$ and its covariance matrix is $\mathrm{Cov}(\epsilon^b) = \mathbf{P}_b$. We also suppose that the background error is independent of the model error $\mathrm{Cov}(\epsilon^b \epsilon^m_k) = 0$

In a statistical framework, the state sequence is modelled as a random sequence $\mathbf{X} = \{\mathbf{X}_k\}_{k\geq 0}$, whose realisations are denoted $\boldsymbol{x} = \{\boldsymbol{x}_k\}_{k\geq 0}$. The evolution law (3.1) is then modelled through the density $p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$. The density of the initial state is $p_{\mathbf{X}_0}(\boldsymbol{x}_0)$.

3.2 Observations, mapping and error

The system $\boldsymbol{x}$ is observed through a set of observations modelled by a variable $\boldsymbol{y}$. The observation vector live in the *observation space* $\mathcal{Y}$ .The mapping between the state space and the observation space is done thanks to the so-called *observation operator* $\mathcal{H}$.

Let us suppose that the observations are local in time (that is to say correspond to the state of the system at a given time) and are available at times coinciding with time steps of the model, i.e. $\boldsymbol{y} = \{\boldsymbol{y}_k\}_{k\geq 1}$.

The observations may be spoiled by instrumental and/or numerical error. We get the following equation:

$$\boldsymbol{y}_k = \mathcal{H}_k(\boldsymbol{x}_k) + \epsilon_k^o \tag{3.2}$$

where $\mathcal{H}_k$ is the local observation operator. ϵ_k^o accounts for the *observation error* at time k. The actual value of this error is not known, it is thus considered as a random variable. We suppose that it is of zero mean $\mathrm{E}(\epsilon_k^o) = 0$, and has a covariance matrix $\mathrm{Cov}(\epsilon_k^o) = \mathbf{R}_k$. Moreover, we make the following noise independence assumptions : $\mathrm{Cov}(\epsilon_l^o \epsilon_k^o) = 0 \ \forall k \neq l$; $\mathrm{Cov}(\epsilon_k^o \epsilon^b) = 0$; $\mathrm{Cov}(\epsilon_k^o \epsilon_k^m) = 0$.

In a statistical framework, the observation sequence is modelled as a random sequence $\mathbf{Y} = \{\mathbf{Y}_k\}_{k \geq 1t}$, whose realisations are denoted $\boldsymbol{y} = \{\boldsymbol{y}_k\}_{k \geq 1}$. The mapping between observations and states is then modelled through the likelihood $p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)$.

3.3 Two points of view

Data assimilation is the process by which observational data are fused with scientific information. At this stage, we therefore have several sources of information on the system we are interested in :

- observations $\boldsymbol{y}$ and associated mapping $\mathcal{H}$
- an evolution model $\mathcal{M}$
- a background $\boldsymbol{x}_b$ of the state of the system at an initial moment (which therefore of course allows, by propagating it by the model $\mathcal{M}$, to have a draft of the whole trajectory of the system)

From this information, data assimilation methods aim at obtaining the "best possible estimate" $\hat{\boldsymbol{x}}$ or $\boldsymbol{x}^a$ (also called analysis or analyzed value) of $\boldsymbol{x}$. All the diversity of approaches will come from the precise meaning that one will give to this expression. Thus, when one speaks of estimating the vector $\boldsymbol{x}$, is one trying to determine the state of the system as a set of physical variables whose value is sought, or more generally as an estimation of a random vector ? And in the latter case, do we look for its complete probability law, or more simply some of its characteristics (mean, mode, covariance matrix ...)? Similarly, the adjective "best" naturally refers to a notion of optimality with respect to a certain criterion. Again, many choices are possible to formalize such a criterion.

3.3.1 Variational approach

A first translation of the previous problem, namely to find the best possible estimate, consists in defining a functional quantifying a discrepancy between the state of the system $\boldsymbol{x}$ and the two available sources of information, i.e. the background $\boldsymbol{x}_b$ and the observations $\boldsymbol{y}$. We then look for the optimal state $\boldsymbol{x}^a$, also called *analysis*, which minimizes this functional. This is the so-called *variational approach* to data assimilation.

$$\boldsymbol{x}^a = \min_{\boldsymbol{x}} J(\boldsymbol{x})$$

where $J(\boldsymbol{x})$ is a cost function defined to measure the misfit between available information (background and observation) and model.

In the simple stationnary case we have:

$$J(\boldsymbol{x}) = J^b(\boldsymbol{x}) + J^o(\boldsymbol{x})$$

- $J^b(\boldsymbol{x})$ stands for the *a priori* term

$$J^b(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}}$$

  where we used the notation $\|x\|^2_A = x^T A x$ to denote the norm associated to the scalar product defined by the symetric positive definite matrix A.

- $J^o(\boldsymbol{x})$ stands for the *data* term

$$J^o(\boldsymbol{x}) = \frac{1}{2}\|\mathcal{H}(\boldsymbol{x}) - \boldsymbol{y}\|^2_{\mathbf{R}^{-1}}$$

In the general case we aim at estimating the trajectory $\boldsymbol{x} = [\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_K]$ by minimizing the cost function

$$J(\boldsymbol{x}) = J^b(\boldsymbol{x}) + J^q(\boldsymbol{x}) + J^o(\boldsymbol{x})$$

- $J^b(\boldsymbol{x})$ stands for the *a priori* term

$$J^b(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}}$$

- $J^o(\boldsymbol{x})$ stands for the *data* term

$$J^o(\boldsymbol{x}) = \frac{1}{2}\sum_{k=1}^{K} \|\mathcal{H}_k(\boldsymbol{x}_k) - \boldsymbol{y}_k\|^2_{\mathbf{R}_k^{-1}}$$

- $J^q(\boldsymbol{x})$ stands for the *model* term

$$J^q(\boldsymbol{x}) = \frac{1}{2}\sum_{k=1}^{K} \|\boldsymbol{x}_k - \mathcal{M}_{k-1\to k}(\boldsymbol{x}_{k-1})\|^2_{\mathbf{Q}_k^{-1}}$$

When the cost function has been set, then the problem is entirely defined, and so is its solution. The "physical" part lies in the definition of J, the choice of the covariance matrices, the background, etc. Finding the solution once the cost function has been defined is "only" technical work.

3.3.2 Bayesian approach

The Bayesian paradigm provides a coherent probabilistic approach for combining information, and thus is an appropriate framework for data assimilation. In that case, the problem is modelling using a hidden Markov state process $\mathbf{X} = \{\mathbf{X}_k\}_{k\geq 0}$ of transition law

$p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$. This probability density models the evolution of the state process. The set of observations $\mathbf{Y} = \{\mathbf{Y}_k\}_{k\geq 1}$, are supposed conditionally independent given the state sequence. The conditional probability $p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)$ describes the likelihood of the observations.

recap : fundamental rules of estimation theory

The two fundamental rules of estimation theory are:
Bayes' rule,

$$p_{\mathbf{X}|\mathbf{Y}}(\boldsymbol{x}|\boldsymbol{y}) = \frac{p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{y}|\boldsymbol{x})p_{\mathbf{X}}(\boldsymbol{x})}{p_{\mathbf{Y}}(\boldsymbol{y})} \tag{3.3}$$

and the marginalisation rule:

$$p_{\mathbf{Y}}(\boldsymbol{y}) = \int p_{\mathbf{X},\mathbf{Y}}(\boldsymbol{x},\boldsymbol{y})d\boldsymbol{x} = \int p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{y}|\boldsymbol{x})p_{\mathbf{X}}(\boldsymbol{x})d\boldsymbol{x} \tag{3.4}$$

needed densities

- The initial density is :

  $$\mathbf{X}_0 \sim \mathcal{N}(\mathbf{X}_0; \boldsymbol{x}_b, \mathbf{P}_b)$$

  where $\mathbf{P}_b$ is the background error covariance matrix.

- The transition law $p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ is obtained by translating the density of ϵ_k^m by $\mathcal{M}_{k-1\to k}(\boldsymbol{x}_{k-1})$. It is generally assumed that the model errors follow a white Gaussian noise, meaning that they are independent of each others, and

  $$\epsilon_k^m \sim \mathcal{N}(\epsilon_k^m; 0, \mathbf{Q}_k)$$

  In that case, we have

  $$\mathbf{X}_k|\mathbf{X}_{k-1} \sim \mathcal{N}(\mathbf{X}_k; \mathcal{M}_{k-1\to k}(\mathbf{X}_{k-1}), \mathbf{Q}_k)$$

  where $\mathbf{Q}_k$ is the model error covariance matrix.

- The likelihood $p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)$ is obtained with a translation of the density ϵ_k^o by $\mathcal{H}_k(\boldsymbol{x}_k)$. It is generally assumed that the observation errors follow a white Gaussian noise, meaning that they are independent of each others (model and observation errors are also supposed independent), and

  $$\epsilon_k^o \sim \mathcal{N}(\epsilon_k^o; 0, \mathbf{R}_k)$$

  In that case, we have

  $$\mathbf{Y}_k|\mathbf{X}_k \sim \mathcal{N}(\mathbf{Y}_k; \mathcal{H}_k(\mathbf{X}_k), \mathbf{R}_k)$$

  where $\mathbf{R}_k$ is the observation error covariance matrix.

Optimal filter The purpose of data assimilation is to estimate states based on observations. This problem can take many forms, so we will focus on the *filtering* problem only, which consists in finding the pdf of $\mathbf{X}_k$ given past and present observations, $\mathbf{Y}_{1:k}$. This conditional pdf is $p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$. A Bayesian recursive solution known as optimal filter is constituted by two interleaved steps:

$$p_{\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1}) \xrightarrow[p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})]{forecast} p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) \xrightarrow[p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)]{analysis} p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$$

- Assuming $p_{\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})$ known, the *prediction step*, also called *forecast step* relying on the dynamic equation enables making a first approximation of the next state given all available information:

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) = \int p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) \,.\, p_{\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})\, d\boldsymbol{x}_{k-1} \quad (3.5)$$

- During the *analysis (or update) step*, the introduction of the new observation $\mathbf{Y}_k = \boldsymbol{y}_k$ corrects this first approximation

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) = \frac{p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)\; p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})}{\int p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)\; p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})\; d\boldsymbol{x}_k} \quad (3.6)$$

Due to their huge dimension, a direct computation of these two integrals can not be realized in a general case. Indeed, defining a computational formulation of the two sums constitutes the key point to solve in filtering problems. As detailled in the following chapter, In the case of linear Gaussian models, the Kalman filter gives the optimal solution in terms of a recursive expression of mean and covariance of the Gaussian filtering distribution $p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$.

3.4 Exercises

Prove the equations of the optimal filter (§3.3.2):

1. Prove eq. (3.5)
2. Prove eq. (3.6)

3.5 Solutions

Prove the equations of the optimal filter (§3.3.2):

1. Prove eq. (3.5)
   Sol. Two rules are needed here :
   (a) the marginalisation rule (3.4)
   (b) the following conditional independence from the Markov property :

$$p_{\mathbf{X}_k|\mathbf{X}_{k-1},\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{y}_{1:k-1}) = p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$$

2. Prove eq. (3.6)
   Sol. Start by writing

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) = p_{\mathbf{X}_k|\mathbf{Y}_k,\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_k,\boldsymbol{y}_{1:k-1})$$

   Then three rules are needed here :
   (a) An extension of the Bayes rule :

$$P(A|B,C) = \frac{P(B|A,C)P(A|C)}{P(B|C)}$$

   (b) the marginalisation rule (3.4)
   (c) the following conditional independence from the Markov property :

$$p_{\mathbf{Y}_k|\mathbf{X}_k,\mathbf{Y}_{1:k-1}}(\boldsymbol{y}_k|\boldsymbol{x}_k,\boldsymbol{y}_{1:k-1}) = p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)$$

4

The Kalman filter

Chapter outline

4.1 Algorithm

Let us consider a hidden Markov chain $(\mathbf{X}, \mathbf{Y})$ whose evolution is described by the following model with additive Gaussian noises:

$$\begin{cases} \mathbf{X}_k = \mathcal{M}_{k-1\to k}(\mathbf{X}_{k-1}) + \epsilon_k^m \\ \mathbf{Y}_k = \mathcal{H}_k(\mathbf{X}_k) + \epsilon_k^o, \end{cases} \tag{4.1}$$

where the following assumptions are made:

- The initial state $\mathbf{X}_0$ is Gaussian, of expectation $\boldsymbol{x}_b$ (background) and covariance $\mathbf{P}_b$
- The model $\mathcal{M}_{k-1\to k}$ and the observation operator $\mathcal{H}_k$ are supposed to be linear, thus we will use the matrices notations, respectively $\mathbf{M}_{k-1,k}$ and $\mathbf{H}_k$
- model errors ϵ_k^m and observation errors ϵ_k^o are white Gaussian noises, of zero mean and of respective covariance $\mathbf{Q}_k$ et $\mathbf{R}_k$. They are supposed to be mutually independent and independent of the initial condition.

Then this system can be written as : :

$$\begin{cases} \mathbf{X}_0 \sim \mathcal{N}(\mathbf{X}_0; \boldsymbol{x}_b, \mathbf{P}_b) \\ \mathbf{X}_k|\mathbf{X}_{k-1} \sim \mathcal{N}(\mathbf{X}_k; \mathbf{M}_{k-1,k}\ \mathbf{X}_{k-1}, \mathbf{Q}_k) \\ \mathbf{Y}_k|\mathbf{X}_k \sim \mathcal{N}(\mathbf{Y}_k; \mathbf{H}_k\ \mathbf{X}_k, \mathbf{R}_k) \end{cases} \tag{4.2}$$

Because of the linearity of the model and observation equations, and as the initial condition is supposed to be Gaussian, the filtering distribution $p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$ is also Gaussian. The Kalman filter (Kalman and Bucy, 1961) gives the equation to compute the optimal filter seen in the previous chapter. It is sequential and decomposed in 2 steps: *forecast* and *analysis* (or observational update)

$$p_{\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1}) = \mathcal{N}(\boldsymbol{x}_{k-1}; \boldsymbol{x}_{k-1}^a, \mathbf{P}_{k-1}^a) \tag{4.3}$$

$$\text{forecast} \quad \Big\downarrow$$

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) = \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{x}_k^f, \mathbf{P}_k^f) \tag{4.4}$$

$$\text{analysis} \quad \Big\downarrow$$

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) = \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{x}_k^a, \mathbf{P}_k^a) \tag{4.5}$$

4.1.1 Forecast step

Let us suppose that the Gaussian pdf $p_{\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})$ is known through the mean $\boldsymbol{x}_{k-1}^a$ and the covariance matrix $\mathbf{P}_{k-1}^a$. The forecast step provides $p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})$ (i.e. expression of $\boldsymbol{x}_k^f$ and $\mathbf{P}_k^f$) using the first step of the optimal filter, eq. (3.5).

We can show that

$$\mathbf{X}_k|\mathbf{Y}_{1:k-1} \sim \mathcal{N}(\mathbf{X}_k \ ; \ \mathbf{M}_{k-1,k} \ \boldsymbol{x}^a_{k-1} \ , \ \mathbf{M}_{k-1,k}\mathbf{P}^a_{k-1}\mathbf{M}^T_{k-1,k} + \mathbf{Q}_{k-1})$$

then,

$$\begin{aligned}
\mathbf{X}_k|\mathbf{Y}_{1:k-1} &\sim \mathcal{N}(\mathbf{X}_k \ ; \ \boldsymbol{x}^f_k \ , \ \mathbf{P}^f_k) \\
\boldsymbol{x}^f_k &= \mathrm{E}[\mathbf{X}_k|\mathbf{Y}_{1:k-1} = \boldsymbol{y}_{1:k-1}] = \mathbf{M}_{k-1,k} \ \boldsymbol{x}^a_{k-1} \\
\mathbf{P}^f_k &= \mathrm{E}[(\mathbf{X}_k - \boldsymbol{x}^f_k)(\mathbf{X}_k - \boldsymbol{x}^f_k)^T|\mathbf{Y}_{0:k-1} = \boldsymbol{y}_{0:k-1}] = \mathbf{M}_{k-1,k}\mathbf{P}^a_{k-1}\mathbf{M}^T_{k-1,k} + \mathbf{Q}_{k-1}
\end{aligned}$$

4.1.2 Analysis step

At time k, $p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})$ is known through the mean $\boldsymbol{x}^f_k$, the covariance matrix $\mathbf{P}^f{}_k$, and the assumption of a Gaussian distribution. The analysis step consists in updating this pdf using the observation $\boldsymbol{y}_k$ available at time k, and find $p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$. This is done using Bayes' rule. Notation f in superscript is used because this comes from a previous forecast, as shown in the previous section. Since both state and observation are from time k here, the time index is dropped for conciseness in this section.

Starting with:

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{:k-1})\frac{1}{(2\pi)^{n/2}|\mathbf{P}^f_k|^{1/2}}\exp\left[-\frac{1}{2}(\boldsymbol{x}_k - \boldsymbol{x}^f_k)^T \ \mathbf{P}^{f\,-1}_k \ (\boldsymbol{x}_k - \boldsymbol{x}^f_k)\right]$$

$$p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k) = \frac{1}{(2\pi)^{n/2}|\mathbf{R}_k|^{1/2}}\exp\left[-\frac{1}{2}(\boldsymbol{y}_k - \mathbf{H}_k\boldsymbol{x}_k)^T \ \mathbf{R}_k{}^{-1} \ (\boldsymbol{y}_k - \mathbf{H}_k\boldsymbol{x}_k)\right]$$

Then Bayes'rule provides the posterior pdf with (3.6). Using (2.4) (here normalized), we have

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) = \mathcal{N}(\boldsymbol{x}_k \ ; \ \boldsymbol{x}^a_k \ , \ \mathbf{P}^a_k)$$

with

$$\begin{aligned}
\boldsymbol{x}^a_k &= \mathrm{E}[\mathbf{X}_k|\mathbf{Y}_{1:k} = \boldsymbol{y}_{1:k}] = \mathbf{P}^a_k\left[\mathbf{P}^{f\,-1}_k\boldsymbol{x}^f_k + \mathbf{H}^T_k\mathbf{R}^{-1}_k\boldsymbol{y}_k\right] \\
\mathbf{P}^a_k &= \mathrm{E}[(\mathbf{X}_k - \boldsymbol{x}^a_k)(\mathbf{X}_k - \boldsymbol{x}^a_k)^T|\mathbf{Y}_{0:k} = \boldsymbol{y}_{0:k}] = \left[\mathbf{P}^{f\,-1}_k + \mathbf{H}^T_k\mathbf{R}^{-1}_k\mathbf{H}_k\right]^{-1}
\end{aligned}$$

With the help of the Sherman-Morrison-Woodbury (SMW) formula:

$$[\mathbf{A} + \mathbf{UDV}]^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}[\mathbf{D}^{-1} + \mathbf{VA}^{-1}\mathbf{U}]^{-1}\mathbf{VA}^{-1}$$

we can also write :

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{P}^f_k \ \mathbf{H}^T_k(\mathbf{R}_k + \mathbf{H}_k\mathbf{P}^f_k\mathbf{H}^T_k)^{-1} \\
\mathbf{P}^a_k &= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}^f_k \\
\boldsymbol{x}^a_k &= \boldsymbol{x}^f_k + \mathbf{K}_k(\boldsymbol{y}_k - \mathbf{H}_k\boldsymbol{x}^f_k)
\end{aligned}$$

4.2 Synthesis

The Kalman filter is initialized with a background state vector $\boldsymbol{x}_b$ and the associated error covariance matrix $\mathbf{P}_b$. The assimilation sequence is performed according to the Kalman filter equations:

Initialization: $\boldsymbol{x}_b$ and $\mathbf{P}_b$

Forecast step:

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) = \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{x}_k^f, \mathbf{P}_k^f) \tag{4.9}$$

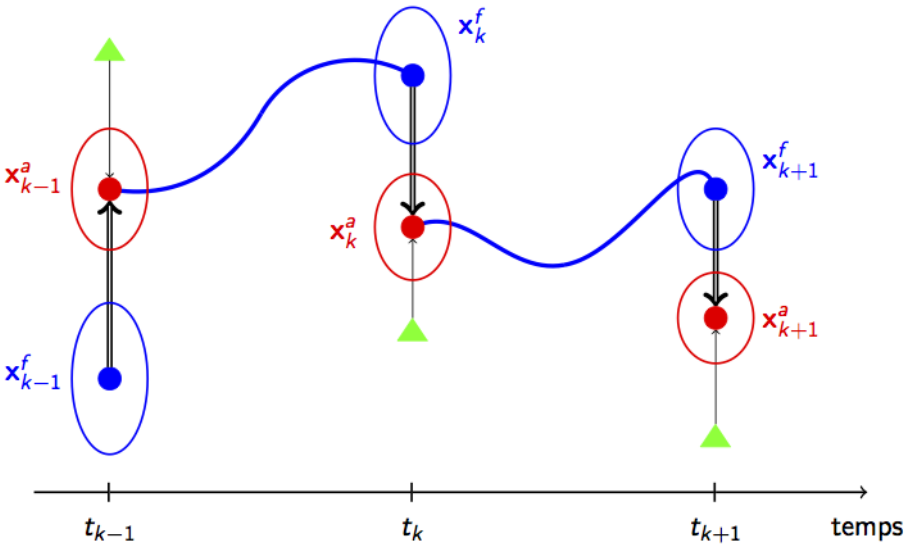
$$\begin{aligned} \boldsymbol{x}_k^f &= \mathbf{M}_{k-1,k}\ \boldsymbol{x}_{k-1}^a & (4.10a)\\ \mathbf{P}_k^f &= \mathbf{M}_{k-1,k}\ \mathbf{P}_{k-1}^a\ \mathbf{M}_{k-1,k}^T + \mathbf{Q}_{k-1} & (4.10b) \end{aligned}$$

Analysis step:

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) = \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{x}_k^a, \mathbf{P}_k^a) \tag{4.11}$$

$$\begin{aligned} \mathbf{K}_k &= \mathbf{P}_k^f\ \mathbf{H}_k^T(\mathbf{R}_k + \mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T)^{-1} & (4.12a)\\ \mathbf{P}_k^a &= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^f & (4.12b)\\ \boldsymbol{x}_k^a &= \boldsymbol{x}_k^f + \mathbf{K}_k(\boldsymbol{y}_k - \mathbf{H}_k\boldsymbol{x}_k^f) & (4.12c) \end{aligned}$$

where exponents ^{f} and ^{a} mean respectively *forecast* and *analysis*. $\mathbf{K}_k$ is called the *Kalman gain* and $(\boldsymbol{y}_k - \mathbf{H}_k\boldsymbol{x}_k^f)$ is the *innovation*.



4.3 Exercices

4.3.1 Proof - first version

1. Prove the forecast step (4.9), using result (2.3)
2. Prove the analysis step (4.11), using results (3.6) and (2.4).

4.3.2 Innovation process

The innovation process denotes the pieces of information added by $\mathbf{Y}_k$, not contained in $\mathbf{Y}_{1:k-1}$:

$$\mathbf{I}_k = \mathbf{Y}_k - \mathrm{E}[\mathbf{Y}_k | \mathbf{Y}_{1:k-1} = \boldsymbol{y}_{1:k-1}]$$

1. prove that $\mathbf{I}_k = \mathbf{Y}_k - \mathbf{H}_k \, \boldsymbol{x}_k^f$
2. prove that $\mathbf{I}_k = \mathbf{H}_k(\mathbf{X}_k - \boldsymbol{x}_k^f) + \epsilon_k^o$
3. deduce that $\mathrm{E}(\mathbf{I}_k) = 0$ and $\mathrm{Cov}(\mathbf{I}_k) = \mathbf{H}_k \, \mathbf{P}_k^f \, \mathbf{H}_k^T + \mathbf{R}_k$
4. deduce from item 2 that $\mathrm{E}[(\mathbf{X}_k - \boldsymbol{x}_k^f)\mathbf{I}_k^T] = \mathbf{P}_k^f \, \mathbf{H}_k^T$

4.3.3 Proof - second version

We can prove the Kalman equations, without using the results (2.3) and (2.4). As we are in a Gaussian case, we need to prove the recursive equations to calculate the expectations and covariances $\boldsymbol{x}_k^f = \mathrm{E}[\mathbf{X}_k | \mathbf{Y}_{0:k-1} = \boldsymbol{y}_{0:k-1}]$, $\mathbf{P}_k^f = \mathrm{E}[(\mathbf{X}_k - \boldsymbol{x}_k^f)(\mathbf{X}_k - \boldsymbol{x}_k^f)^T | \mathbf{Y}_{0:k-1} = \boldsymbol{y}_{0:k-1}]$, $\boldsymbol{x}_k^a = \mathrm{E}[\mathbf{X}_k | \mathbf{Y}_{0:k} = \boldsymbol{y}_{0:k}]$, $\mathbf{P}_k^a = \mathrm{E}[(\mathbf{X}_k - \boldsymbol{x}_k^a)(\mathbf{X}_k - \boldsymbol{x}_k^a)^T | \mathbf{Y}_{0:k} = \boldsymbol{y}_{0:k}]$.

1. Prove (4.10a)
2. Prove (4.10b)
3. Prove (4.12a) , (4.12b) , (4.12c) using results from the Innovation process exercice. The proof is not straightforward at all. It is enough if you make sure that you understand the solution.

4.3.4 Special cases

Write the Kalman filter algorithm for the following special cases:

1. There is no observation. Comment.
2. Observations are perfect: we assume that their quality is perfect ($\mathbf{R}_k = 0$) and that $\mathbf{H}_k$ is invertible.
3. $\mathbf{x}$ is a scalar, model $\mathbf{M_k}$ is the identity matrix, $\mathbf{H}_k$ as well, and the error statistics $\mathbf{R_k}$ and $\mathbf{Q_k}$ are constant over time. We denote $\mathbf{R} = r^2$ the scalar 1-d matrix of observation error covariance, and similarly $\mathbf{P}_k^f = \sigma_k^{f\,2}$, $\mathbf{P}_k^a = \sigma_k^{a\,2}$, and $\mathbf{Q} = q^2$.

(a) give the equations of the Kalman filter in that situation
(b) study also the asymptotic behaviour of the analysis error variance $\sigma_k^{a\,2}$ for $k \to +\infty$ (use a fix-point).
(c) what is then the asymptotics for x^a?

4.4 Limitations of the Kalman filter

4.4.1 Definition of covariance matrices, filter divergence

If the input statistical information is mis-specified, the filtering system may come to underestimate the state error variances $\mathbf{P}$. For exemple, if $\mathbf{Q}$ is underestimated, too much confidence is then given to the state estimation and the effects of the analyses are then minimized. In the extreme case, observations are simply rejected. This is a filter divergence.
Very often filter divergence is quite easy to diagnose: state error variances are small and the time sequence of innovations is biased. But it is not always simple to correct. The main rule to follow is not to underestimate model errors. If possible, it is better to use an adaptive scheme to tune them online.

4.4.2 Nonlinearity and large dimension

One limitation of the straightforward implementation of the Kalman filter is the problem dimension. In oceanography or meteorology, models generally involve several millions (very often tens of millions, even hundreds of millions sometimes) of variables. Let us call n the number of variables. A state covariance matrix is then $n \times n$. With the dimensions considered, the storage of such matrix is obviously impossible. One standard solution is *rank reduction*. Another strategy is to rely on the Ensemble Kalman filter that is described next chapter.

Nonlinearity of the dynamics or the mapping observation operator poses two problems to the Kalman filter. First, the transposed models are not defined. Then, nonlinearity destroys gaussianity of statistics. On way to proceed with nonlinearity is given by the extended Kalman filter detailed here ($\mathbf{M}_{k-1,k}$ and $\mathbf{H}_k$ denote the tangent linear models of $\mathcal{M}_{k-1\to k}$ and $\mathcal{H}_k$ respectively, applied to respectively $\boldsymbol{x}_{k-1}^a$ and $\boldsymbol{x}_k^f$):

Initialization: $\boldsymbol{x}_b$ and $\mathbf{P}_b$

Forecast step:

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) = \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{x}_k^f, \mathbf{P}_k^f)$$

$$\begin{aligned} \boldsymbol{x}_k^f &= \mathcal{M}_{k-1\to k}(\boldsymbol{x}_{k-1}^a) \\ \mathbf{P}_k^f &= \mathbf{M}_{k-1,k}\, \mathbf{P}_{k-1}^a\, \mathbf{M}_{k-1,k}^T + \mathbf{Q}_{k-1} \end{aligned}$$

Analysis step:

$$p_{\mathbf{X}_k|\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) = \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{x}_k^a, \mathbf{P}_k^a)$$

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{P}_k^f \, \mathbf{H}_k^T (\mathbf{R}_k + \mathbf{H}_k \mathbf{P}_k^f \mathbf{H}_k^T)^{-1} \\
\mathbf{P}_k^a &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^f \\
\boldsymbol{x}_k^a &= \boldsymbol{x}_k^f + \mathbf{K}_k (\boldsymbol{y}_k - \mathcal{H}_k(\boldsymbol{x}_k^f))
\end{aligned}$$

However, this algorithm is valid only for weakly nonlinear models. Otherwise, one may rely on the Ensemble Kalman filter (of only the dynamics is non linear and the gaussianity assumptions holds) or on the particle filter (if the dimension is not too large)

4.5 Exercice - Extended Kalman Filter

Let us consider the Lorentz model

$$\begin{aligned}
\frac{dx}{dt} &= \sigma(y - x) \\
\frac{dy}{dt} &= \rho x - y - xz \\
\frac{dz}{dt} &= -\beta z + xy
\end{aligned} \tag{4.15}$$

This system is a non-linear differential system of order 1. Using the following parameters, $\sigma = 10$, $\beta = \frac{8}{3}$, $\rho = 28.0$, it gives rise to a chaotic system. the solution is seen to orbit around two equilibrium points giving two 'regimes'.

We aim at performing data assimilation. We assume our model is inaccurate, and the initial condition is not perfectly known. To mimic this, we assimilate observations extracted from a reference simulation into an unperfect model (with $\rho = 29$) initialized with unperfect conditions : the background is set to $(3, -3, 21)$. We consider the case where all variables (x, y and z) are observed at all times. Write the Extended Kalman filter algorithm for that data assimilation problem. Write it again, in the case where only variable x is observed.

4.6 Solutions

4.6.1 Proof - first version

1. result given by (2.3) and identification ($\mathbf{Y}$ is $\boldsymbol{x}_k$, $\mathbf{X}$ is $\boldsymbol{x}_{k-1}$)
2. remarking that :

$$\begin{aligned}(\boldsymbol{y}-\mathbf{H}\boldsymbol{x})^T\mathbf{R}^{-1}(\boldsymbol{y}-\mathbf{H}\boldsymbol{x}) &= (\boldsymbol{x}^T\mathbf{H}^T-\boldsymbol{y}^T)\mathbf{R}^{-1}(\boldsymbol{y}-\mathbf{H}\boldsymbol{x})\\ &= (\boldsymbol{x}^T\mathbf{H}^T-\boldsymbol{y}^T)\mathbf{H}^{T^{-1}}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{H}^{-1}(\mathbf{H}\boldsymbol{x}-\boldsymbol{y})\\ &= (\boldsymbol{x}^T-\boldsymbol{y}^T\mathbf{H}^{T^{-1}})\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}(\boldsymbol{x}-\mathbf{H}^{-1}\boldsymbol{y})\\ &= (\boldsymbol{x}-\mathbf{H}^{-1}\boldsymbol{y})^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}(\boldsymbol{x}-\mathbf{H}^{-1}\boldsymbol{y})\end{aligned}$$

   then we proceed by identification with :

$$\mathbf{a}=\mathbf{H}^{-1}\boldsymbol{y} \qquad \mathbf{A}=\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}$$

$$\mathbf{b}=\boldsymbol{x}_k^f \qquad \mathbf{B}=\mathbf{P}_k^f$$

4.6.2 Innovation process

1.

$$\begin{aligned}\mathbf{I}_k &= \mathbf{Y}_k-\mathrm{E}[\mathbf{Y}_k|\mathbf{Y}_{1:k-1}=\boldsymbol{y}_{1:k-1}]\\ &= \mathbf{Y}_k-\mathrm{E}[\mathbf{H}_k\mathbf{X}_k+\epsilon_k^o|\mathbf{Y}_{1:k-1}=\boldsymbol{y}_{1:k-1}]\\ &= \mathbf{Y}_k-\mathbf{H}_k\boldsymbol{x}_k^f\end{aligned}$$

2.

$$\begin{aligned}\mathbf{I}_k &= \mathbf{Y}_k-\mathbf{H}_k\boldsymbol{x}_k^f\\ &= \mathbf{H}_k\mathbf{X}_k+\epsilon_k^o-\mathbf{H}_k\boldsymbol{x}_k^f\end{aligned}$$

3.

$$\begin{aligned}\mathrm{Cov}(\mathbf{I}_k) &= \mathrm{E}[\mathbf{I}_k\mathbf{I}_k^T]\\ &= \mathrm{E}[(\mathbf{H}_k(\mathbf{X}_k-\boldsymbol{x}_k^f)+\epsilon_k^o)(\mathbf{H}_k(\mathbf{X}_k-\boldsymbol{x}_k^f)+\epsilon_k^o)^T]\\ \text{then we develop:} &= \mathbf{H}_k\,\mathbf{P}_k^f\,\mathbf{H}_k^T+\mathbf{R}_k\end{aligned}$$

   we use the hypothesis that the forecast error and the observation error are decorrelated

4.

$$\begin{aligned}\mathrm{E}[(\mathbf{X}_k-\boldsymbol{x}_k^f)\mathbf{I}_k^T] &= \mathrm{E}[(\mathbf{X}_k-\boldsymbol{x}_k^f)(\mathbf{H}_k(\mathbf{X}_k-\boldsymbol{x}_k^f)+\epsilon_k^o)^T]\\ &= \mathbf{P}_k^f\,\mathbf{H}_k^T\end{aligned}$$

   by developping and using decorrelated rules.

4.6.3 Proof - second version

il faut re-ecrire la preuve en ajoutant le conditionnement sur les mesures dans les covariances - ca devrait marcher exactement pareil

1. (4.10a): forecast from the model and the previous analysis.

$$\begin{aligned} \boldsymbol{x}_k^f &= \mathrm{E}[\mathbf{X}_k|\mathbf{Y}_{1:k-1} = \boldsymbol{y}_{1:k-1}] \\ &= \mathrm{E}[\mathbf{M}_{k-1,k}\ \mathbf{X}_{k-1} + \epsilon_k^m|\mathbf{Y}_{1:k-1} = \boldsymbol{y}_{1:k-1}] \\ &= \mathbf{M}_{k-1,k}\ \mathrm{E}[\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1} = \boldsymbol{y}_{1:k-1}] + \mathrm{E}[\epsilon_{k-1}^m|\mathbf{Y}_{1:k-1} = \boldsymbol{y}_{1:k-1}] \\ &= \mathbf{M}_{k-1,k}\ \boldsymbol{x}_{k-1}^a \end{aligned}$$

2. (4.10b):

$$\begin{aligned} \mathbf{P}_k^f &= \mathrm{E}[(\mathbf{X}_k - \boldsymbol{x}_k^f)(\mathbf{X}_k - \boldsymbol{x}_k^f)^T] \\ &= \mathrm{E}[(\mathbf{M}_{k-1,k}(\mathbf{X}_{k-1} - \boldsymbol{x}_{k-1}^a) + \epsilon_{k-1}^m)\ (')^T] \\ \text{then we develop:} \quad &= \mathbf{M}_{k-1,k}\mathbf{P}_{k-1}^a\mathbf{M}_{k-1,k}^T + \mathrm{E}[\epsilon_{k-1}^m\epsilon_{k-1}^{m\ T}] \\ &+ \mathbf{M}_{k-1,k}\mathrm{E}[(\mathbf{X}_{k-1} - \boldsymbol{x}_{k-1}^a)\epsilon_{k-1}^{m\ T}] + \mathrm{E}[\epsilon_{k-1}^m(\mathbf{X}_{k-1} - \boldsymbol{x}_{k-1}^a)^T]\mathbf{M}_{k-1,k}^T \end{aligned}$$

   we use the hypothesis that the analysis error and the model error are decorrelated

$$\mathbf{P}_k^f = \mathbf{M}_{k-1,k}\mathbf{P}_{k-1}^a\mathbf{M}_{k-1,k}^T + \mathbf{Q}_{k-1}$$

3.

$$\begin{aligned} \boldsymbol{x}_k^a &= \mathrm{E}[\mathbf{X}_k|\mathbf{Y}_{1:k} = \boldsymbol{y}_{1:k}] \\ &= \boldsymbol{x}_k^f + \mathrm{E}[\mathbf{X}_k - \boldsymbol{x}_k^f|\mathbf{Y}_{1:k} = \boldsymbol{y}_{1:k}] \\ &= \boldsymbol{x}_k^f + \mathrm{E}[\mathbf{X}_k - \boldsymbol{x}_k^f|\mathbf{Y}_{1:k-1} = \boldsymbol{y}_{1:k-1}, \mathbf{I}_k = \boldsymbol{i}_k] \\ &= \boldsymbol{x}_k^f + \mathrm{E}[\mathbf{X}_k - \boldsymbol{x}_k^f|\mathbf{I}_k = \boldsymbol{i}_k] \end{aligned}$$

   then

$$\begin{aligned} \mathbf{X}_k - \boldsymbol{x}_k^a &= (\mathbf{X}_k - \boldsymbol{x}_k^f) - (\boldsymbol{x}_k^a - \boldsymbol{x}_k^f) \\ &= (\mathbf{X}_k - \boldsymbol{x}_k^f) - \mathrm{E}[\mathbf{X}_k - \boldsymbol{x}_k^f|\mathbf{I}_k = \boldsymbol{i}_k] \end{aligned}$$

   so that

$$\begin{aligned} \mathbf{P}_k^a &= \mathrm{E}[(\mathbf{X}_k - \boldsymbol{x}_k^a)(\mathbf{X}_k - \boldsymbol{x}_k^a)^T] \\ &= \mathrm{E}[((\mathbf{X}_k - \boldsymbol{x}_k^f) - \mathrm{E}[\mathbf{X}_k - \boldsymbol{x}_k^f|\mathbf{I}_k = \boldsymbol{i}_k])(')^T] \end{aligned}$$

   We therefore need to calculate the expectation and variance of $\mathbf{X}_k - \boldsymbol{x}_k^f|\mathbf{I}_k = \boldsymbol{i}_k$. Using results from the exercice on the innovation process, the vector $(\mathbf{X}_k - \boldsymbol{x}_k^f, \mathbf{I}_k)$ is Gaussian of mean 0 and covariance:

$$\begin{pmatrix} \mathbf{P}_k^f & \mathbf{P}_k^f\ \mathbf{H}_k^T \\ \mathbf{H}_k\ \mathbf{P}_k^{fT} & \mathbf{H}_k\ \mathbf{P}_k^f\ \mathbf{H}_k^T + \mathbf{R}_k \end{pmatrix}$$

   then using (2.1), we can deduce

$$\mathbf{P}_k^a = \mathbf{P}_k^f - \mathbf{P}_k^f\ \mathbf{H}_k^T(\mathbf{H}_k\ \mathbf{P}_k^f\ \mathbf{H}_k^T + \mathbf{R}_k)^{-1}\mathbf{H}_k\ \mathbf{P}_k^{fT}$$

which proves (4.12a) and (4.12b). and finally

$$\begin{aligned} \boldsymbol{x}_k^a &= \boldsymbol{x}_k^f + \mathrm{E}[\mathbf{X}_k - \boldsymbol{x}_k^f | \mathbf{I}_k = \boldsymbol{i}_k] \\ &= \boldsymbol{x}_k^f + \mathbf{P}_k^f \, \mathbf{H}_k^T (\mathbf{H}_k \, \mathbf{P}_k^f \, \mathbf{H}_k^T + \mathbf{R}_k)^{-1} (\boldsymbol{y}_k - \mathbf{H}_k \boldsymbol{x}_k^f) \end{aligned}$$

which proves (4.12c). Youhou, it's done !!

4.6.4 Special cases

4. Without observations, the Kalman filter consists only of the forecast equation (4.10a) (4.10b), In particular (except very special cases), the error always increases.

5. With perfect obs the equation for $\mathbf{K}_k$ is simpler: $\mathbf{K}_k = \mathbf{H}_k^{-1}$, then we find $\mathbf{P}_k^a = 0$, and finally $\boldsymbol{x}_k^a = \mathbf{H}_k^{-1} \boldsymbol{y}_k$, $\mathbf{P}_k^f = \mathbf{Q}_k$. We trust the data completely, the forecast error is just the model error.

6. (a) We denote $\mathbf{R} = r^2$ the scalar 1-d matrix of observation error covariance, and similarly $\mathbf{P}_k^f = {\sigma_k^f}^2$, $\mathbf{P}_k^a = \sigma_k^{a2}$, and $\mathbf{Q} = q^2$. Then, as $\mathbf{M} = \mathbf{H} = id.$ we have :

   - (4.10a) : $x_k^f = x_{k-1}^a$
   - (4.10b) : ${\sigma_k^f}^2 = {\sigma_{k-1}^a}^2 + q^2$
   - the kalman gain equation (4.12a) is $k_k = \frac{{\sigma_k^f}^2}{({\sigma_k^f}^2 + r^2)}$
   - (4.12b) : $\sigma_k^{a2} = \frac{{\sigma_k^f}^2 r^2}{{\sigma_k^f}^2 + r^2} = k_k r^2$
   - (4.12c) : $x_k^a = x_k^f + k_k(y_k - x_k^f)$

   (b) asymptotic for σ_k^{a2}: we have the recursive equation

$$\sigma_k^{a2} = \frac{{\sigma_k^f}^2 r^2}{{\sigma_k^f}^2 + r^2}$$

   using (4.10b) we get

$$\sigma_k^{a2} = \frac{({\sigma_{k-1}^a}^2 + q^2) r^2}{{\sigma_{k-1}^a}^2 + q^2 + r^2}$$

   for $k \to \infty$ we have $\sigma_k^a \to \sigma_\infty^a$ where ${\sigma_\infty^a}^2 = X$ is the solution of the fixed point equation

$$X = \frac{(X + q^2) r^2}{(X + q^2 + r^2}$$

   equivalent to $X^2 + q^2 X - q^2 r^2 = 0$.
   whose positive solution is $X = {\sigma_\infty^a}^2 = \frac{-q^2 + \sqrt{q^4 + 4q^2 r^2}}{2}$.

(c) for x^a, using $k_k = \frac{\sigma_k^{a\,2}}{r^2}$ and $x_k^f = x_{k-1}^a$ we then have $x_k^a = x_{k-1}^a + \frac{\sigma_k^{a\,2}}{r^2}(y_k - x_{k-1}^a)$, and for large time we have

$$x_k^a = (1 - \frac{\sigma_\infty^{a\ 2}}{r^2})x_{k-1}^a + \frac{\sigma_\infty^{a\ 2}}{r^2}y_k$$

5

Ensemble algorithms : Ensemble Kalman filter and Particle filter

Chapter outline

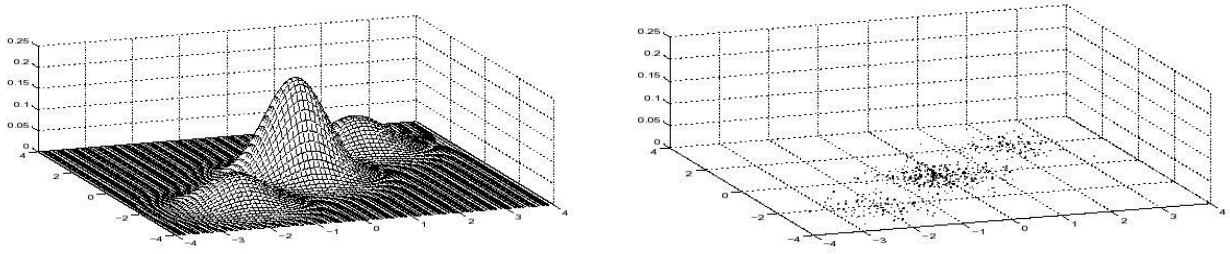
As seen in the previous chapter, the Kalman filter solves the optimal filter by given explicit recursive expressions of the two first moments of the pdfs. This is optimal only in the linear Gaussian case. What happens if the models are nonlinear and / or the pdfs non Gaussian? The Kalman filter is no more optimal and, more importantly, can easily fail the estimation process. Other approaches must be used, relying on sampling strategies. Then come the Ensemble Kalman filter and the particle filter. They work sequentially in the spirit of the Kalman filter, but unlike the latter, they handle an ensemble of states (members or particles) which distribution approximates the pdf of the true state, eventually Gaussian if one is using the ensemble Kalman filter.

5.1 Monte Carlo principle and sampling[1]

The idea of Monte Carlo simulation is to draw an i.i.d. set of samples $\{\boldsymbol{x}^{(i)}\}_{i=1...N}$ from a target density $p_{\mathbf{X}}(\boldsymbol{x})$ defined on a high-dimensional space. These N samples can de used to approximate the target density with the following empirical point mass function

$$p_{\mathbf{X}}(\boldsymbol{x}) \approx \frac{1}{N} \sum_{i=1}^{N} \delta_{\boldsymbol{x}^{(i)}}(\boldsymbol{x}), \tag{5.1}$$

where $\delta_{\boldsymbol{x}^{(i)}}(\boldsymbol{x})$ denotes the delta-Dirac mass located at $\delta_{\boldsymbol{x}^{(i)}}$.



Then we will be able to approximate integrals such that:

$$I[f(\boldsymbol{x})] = \int_{\mathcal{X}} f(\boldsymbol{x})\ p_{\mathbf{X}}(\boldsymbol{x}) \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{x}^{(i)})$$

When $p_{\mathbf{X}}(\boldsymbol{x})$ has standard form, e.g Gaussian, it is straightforward to sample from it using available routines. However, when this is not the case, we need to introduce more sophisticated techniques. Most of the sampling methods to sample from $p_{\mathbf{X}}(\boldsymbol{x})$ are based on the following principles :

1. choose a distribution $q_{\mathbf{X}}(\boldsymbol{x})$ from which it is easy to obtain samples. This distribution is called the *importance distribution* or *proposal*
2. compensate the errors due to the sampling from the wrong distribution
3. The sampling quality increases with the number of samples (the ideal is $N \to \infty$)

[1]This section is largely inspired from An Introduction to MCMC for Machine Learning, C Andrieu, N de Freitas, A Doucet, M. I. Jordan

5.1.1 Rejection sampling

Rejection sampling is based on the previous principles.The following assumptions are made :

- the easy-to-sample proposal distribution $q_{\mathbf{X}}(\boldsymbol{x})$ satisfies $p_{\mathbf{X}}(\boldsymbol{x}) \leq M\ q_{\mathbf{X}}(\boldsymbol{x})$ with $M < \infty$
- we are able to evaluate $p_{\mathbf{X}}(\boldsymbol{x})$ for all $\boldsymbol{x}$

`rejection sampling : generate N i.i.d. samples` $\{\boldsymbol{x}^{(i)}\}_{1=1...N}$ `de` $p_{\mathbf{X}}(\boldsymbol{x})$

- $i = 0$
- `while` $i \neq N$ `do`
  1. `generate a realisation` $\boldsymbol{x}^{(i)}$ `from` $q_{\mathbf{X}}(\boldsymbol{x})$
  2. `generate a realisation` u `from` $\mathcal{U}_{(0,1)}$
  3. `if`

$$u < \frac{p_{\mathbf{X}}(\boldsymbol{x}^{(i)})}{M\ q_{\mathbf{X}}(\boldsymbol{x}^{(i)})}$$

  `then accept` $\boldsymbol{x}^{(i)}$ `and increment` i`; otherwise reject the sample.`

The drawbacks of this method are the following. First, It is not always possible to find a value M that is valid for the whole space $\mathcal{X}$. Second, the acceptance probability of a sample is proportional to $1/M$ thus the method is intractable in high dimension when M is large.

5.1.2 Importance sampling

Importance sampling (IS) is an alternative to rejection sampling. The good thing is that the hypothese on the existence of M is removed. Again, we suppose that we know how to generate samples from a importance function $q_{\mathbf{X}}(\boldsymbol{x})$ such that $p_{\mathbf{X}}(\boldsymbol{x}) > 0 \Rightarrow q_{\mathbf{X}}(\boldsymbol{x}) > 0$

By remarking the following equality :

$$I[f(\boldsymbol{x})] = \int_{\mathbf{X}} f(\boldsymbol{x})\ p_{\mathbf{X}}(\boldsymbol{x})\ d\boldsymbol{x} = \int_{\mathbf{X}} f(\boldsymbol{x})\ \frac{p_{\mathbf{X}}(\boldsymbol{x})}{q_{\mathbf{X}}(\boldsymbol{x})}\ q_{\mathbf{X}}(\boldsymbol{x})\ d\boldsymbol{x}$$

we can deduce that we can sample $\boldsymbol{x}^{(i)}$ from $q_{\mathbf{X}}$ and use the ration $\frac{p_{\mathbf{X}}(\boldsymbol{x}^{(i)})}{q_{\mathbf{X}}(\boldsymbol{x}^{(i)})}$ to compensate the error made. and

$$p_{\mathbf{X}}(\boldsymbol{x}) \approx \frac{1}{N} \sum_{i=1}^{N} \widetilde{w}^{(i)} \delta_{\boldsymbol{x}^{(i)}}(\boldsymbol{x}), \tag{5.2}$$

IS : generate N i.i.d. weighted samples $\{\boldsymbol{x}^{(i)}, \widetilde{w}^{(i)}\}_{1=1...N}$ approximating $p_{\mathbf{X}}(\boldsymbol{x}^{(i)})$

Importance sampling

- Generate N samples $\{\boldsymbol{x}^{(i)}\}_{i=1:N}$ from $q_{\mathbf{X}}(\boldsymbol{x})$
- Associate to each sample its normalised importance weight

$$\widetilde{w}^{(i)} = \frac{w^{(i)}}{\sum_{j=1}^{N} w^{(j)}} \text{ whith } w^{(i)} = \frac{p_{\mathbf{X}}(\boldsymbol{x}^{(i)})}{q_{\mathbf{X}}(\boldsymbol{x}^{(i)})}$$

5.2 The Ensemble Kalman Filter (EnKF)

Let us start with the Ensemble Kalman Filter (EnKF). As we have seen in the previous chapter, when the model and/or observation operators are non linear, one may rely on the Extended Kalman Filter. However, in high dimensional applications, the EKF cannot be implemented due to the high cost associated with the construction of the evolved covariance matrix. Besides, the implementation of the EKF relies on the local linear tangent which leads to neglect the nonlinear effects. Efficient ensemble techniques have been devised specifically for that purpose. They are mainly defined through replacing the forecast mean and covariance matrix by an empirical expression of the ensemble mean and covariance matrix.

The structure of the EnKF is the same as the Kalman filter, i.e. it works in two phases, forecast and analysis. But, here, the pdfs are represented by a sample (called the ensemble) of states (the members). In the forecast phase, the ensemble is propagated by the nonlinear dynamics. This ensemble of samples allows computing an approximation of the error covariance matrix. The analysis step is computed in Kalman's fashion: the analysis equations are applied to each member of the ensemble. For consistency with the observation error covariance matrix, the observations used need to be noised accordingly. All the other statistical information necessary to the BLUE is calculated from the ensemble.

The algorithm version presented here is called the stochastic Ensemble Kalman filter. Let N be the number of members in the ensemble, and i a member index running from 1 to N.

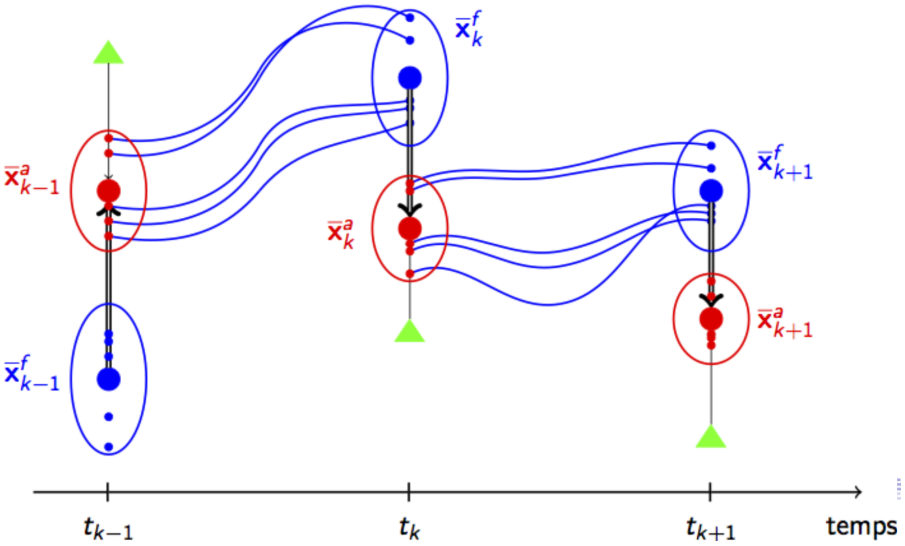
Initialization: $\boldsymbol{x}_0^{a(i)} = \boldsymbol{x}_b + \epsilon^b$ with $\epsilon^b \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_b)$

Forecast step:

$$\begin{aligned}
\boldsymbol{x}_k^{f(i)} &= \mathcal{M}_{k-1\to k}(\boldsymbol{x}_{k-1}^{a(i)}) \\
\bar{\boldsymbol{x}}^f &= \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_k^{f(i)} \\
\mathbf{P}_k^f &= \frac{1}{N-1}\sum_{i=1}^{N} \left(\boldsymbol{x}_k^{f(i)} - \bar{\boldsymbol{x}}^f\right)\left(\boldsymbol{x}_k^{f(i)} - \bar{\boldsymbol{x}}^f\right)^T
\end{aligned}$$

Analysis step:

$$\begin{aligned}
\mathbf{P}_k^f\mathbf{H}_k^T &= \frac{1}{N-1}\sum_{i=1}^{N} \left(\boldsymbol{x}_k^{f(i)} - \bar{\boldsymbol{x}}^f\right)\left(\mathcal{H}_k(\boldsymbol{x}_k^{f(i)}) - \mathcal{H}_k(\bar{\boldsymbol{x}}^f)\right)^T \\
\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k{}^T &= \frac{1}{N-1}\sum_{i=1}^{m} \left(\mathcal{H}_k(\boldsymbol{x}_k^{f(i)}) - \mathcal{H}_k(\bar{\boldsymbol{x}}^f)\right)\left(\mathcal{H}_k(\boldsymbol{x}_k^{f(i)}) - \mathcal{H}_k(\bar{\boldsymbol{x}}^f)\right)^T \\
\mathbf{K}_k &= \mathbf{P}_k^f\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \\
\boldsymbol{y}_k^{(i)} &= \boldsymbol{y}_k + \epsilon^{o(i)} \text{ with } \epsilon^{o(i)} \sim \mathcal{N}(0, \mathbf{R}_k) \\
\boldsymbol{x}_k^{a(i)} &= \boldsymbol{x}_k^{f(i)} + \mathbf{K}_k(\boldsymbol{y}_k^{(i)} - \mathcal{H}_k(\boldsymbol{x}_k^{f(i)})
\end{aligned}$$



In this version of the algorithm, no model error is assumed. If the model is imperfect, the integration scheme needs to be adjusted to account for the stochastic noise : the cloud of possible states is generated from a randomization of the dynamics or of its parameters.

5.2.1 How to make it works in very high dimensions ?

Let us suppose that we have a data assimilation problem coming from geophysics, where the state to be estimed is represented by thousands of unknowns living on a discretisation grid (exemple fig 5.1)

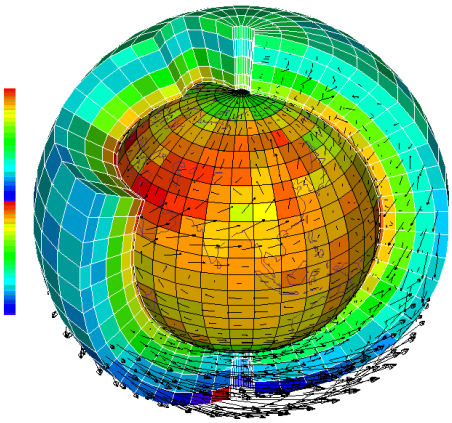


Figure 5.1: exemple of a discretisation grid for an atmospheric model

5.2.1.1 Localization

With the presented EnKF, the problem of storing the state covariance matrix, mentionned in section 4.4.2, is solved. "Only" N state vectors are stored. In the standard EnKF, the inversion of the innovation error covariance matrix is still required to compute the Kalman gain. The dimension of this matrix is $s \times s$, s being the number of observations. In real problems, s may easily become of the order of a few hundred, which makes the inversion prohibitive. The usual strategy to tackle this problem is to localize the analysis, i.e., to consider, for the correction at one grid point, only the observations present within a limited region in the close environment. Thus, the Kalman gain is different and must be recomputed for each grid point. But the local innovation error covariance matrix is of low dimension and its inversion is possible. Localization is a very important aspect in high dimensional Kalman filtering.

Localization is not only useful to compute the Kalman gain in the EnKF. It also prevents corrections due to distant observations. Such corrections are due to significant correlations between distant grid points. But these correlations are very often due more to the effect of subsampling rather than real physical and statistical reasons.

5.2.1.2 Inflation

Even when the analysis is made local, the error covariance matrices are still evaluated with an ensemble of limited size. This often leads to sampling errors, that can accumulate in time and lead to the filter divergence. One way around is to inflate the error covariance matrix by a given factor:

$$\mathbf{P}^a = \lambda^2 \mathbf{P}^a$$

Such a trick can also be applied to $\mathbf{P}^f$

Many more elements on Ensemble Kalman filters can be found in chapter 6 of the book *Data assimilation - Methods, Algorithms, and Applications*, M. Asch, M. Bocquet, M. Nodet, ed. Siam, 2016.

5.3 Particle filter

As the EnKF computes the two first moments of the distributions, it still relies on Gaussian assumptions. What happens if both models are nonlinear and the pdfs are non-Gaussian ? Then the Kalman filter and extensions are non longer optimal and can easily fail the estimation process. Another approach must be used. A promising candidate is the particle filter.

The idea behind particle filtering is very simple. These techniques propose to implement recursively an approximation of the complete density $p_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$ by N samples $\boldsymbol{x}_{0:k}^{(i)}$, following Importance sampling strategy (see section 5.1.2). This approximation consists in a finite weighted sum of N Diracs centered on hypothesized locations in the state space – called particles, (members of the ensemble in the EnKF jargon) – of the initial system $\mathbf{x}_0$. At each particle $\boldsymbol{x}_{0:k}^{(i)}$ $(i = 1 : N)$ is assigned a weight $w_k^{(i)}$ describing its relevance. This approximation can be formulated with the following expression:

$$p_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k}) \approx \sum_{i=1:N} \widetilde{w}_k^{(i)} \delta_{\boldsymbol{x}_{0:k}^{(i)}}(\boldsymbol{x}_{0:k})$$

The IS principle tells us that if the samples $\boldsymbol{x}_{0:k}^{(i)}$ were drawn from an importance density $q_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$, the weightw are defined by :

$$\frac{w^{(i)}}{\sum_{j=1}^{N} w^{(j)}} \quad \text{with} \quad w_k^{(i)} = \frac{p_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}^{(i)}|\boldsymbol{y}_{1:k})}{q_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}^{(i)}|\boldsymbol{y}_{1:k})} \tag{5.3}$$

5.3.1 Towards a sequential algorithm

Assuming that the approximation of $p_{\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1})$ is known, we aim at designing a recursive algorithm to approximate $p_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$. To do so, we have to make the following assumption on the importance density :

$$q_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k}) = q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1},\boldsymbol{y}_{1:k})\; q_{\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1})$$

Then one can obtain samples $\boldsymbol{x}_{0:k}^{(i)} \sim q_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k})$ by augmenting each of the existing samples $\boldsymbol{x}_{0:k-1}^{(i)} \sim q_{\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1})$ with the new state $\boldsymbol{x}_k^{(i)} \sim q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1},\boldsymbol{y}_{1:k})$. To maintain a consistent sample, the importance weights are updated according to a recursive evaluation as the new measurement $\boldsymbol{y}_k$ becomes available :

$$\widetilde{w}_k^{(i)} \propto \widetilde{w}_{k-1}^{(i)} \frac{p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k^{(i)})\; p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(i)})}{q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{0:k-1}^{(i)},\boldsymbol{y}_{1:k})}, \qquad \sum_{i=1:N} \widetilde{w}_k^{(i)} = 1. \tag{5.4}$$

Limiting ourself to these two steps for updating the swarm of particles induces an increase over time of the weight variance. In practice, this *degeneracy problem* makes the number of significant particles decreases dramatically over time implying an impoverishment of the estimate. From time to time, it is thus necessary to perform a resampling

step. This procedure aims at removing particles with weak normalized weights, and multiplying particles associated to strong weights, as soon as the number of significant particles is too small. Consequently, resampled particles tend to be concentrated in areas where important features exist. The decision to perform or not a resampling step may be depending on a value called the *effective sample size* ESS, whose value i slow if the sample is impoverished :

$$ESS = \frac{N}{\sum_i (\widetilde{w}_k^{(i)})^2}$$

These three main steps (sampling / calculation of the importance weights / resampling) constitute the general framework of particle filtering. Let N be the number of particules, and i a member index running from 1 to N.

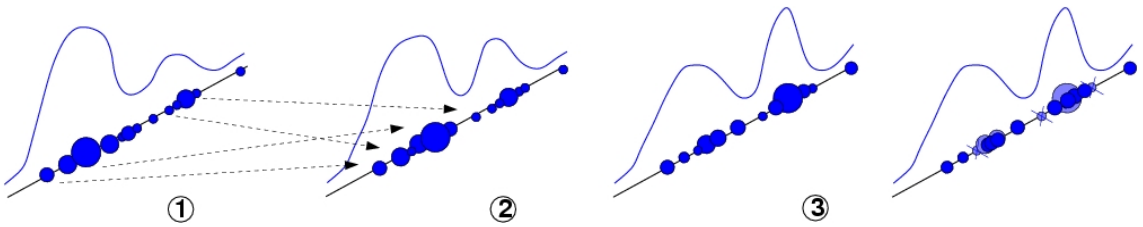
Initialisation Generate N samples $\{\boldsymbol{x}_0^{(i)}\}_{i=1...N}$ and set $w_0^{(i)} = \frac{1}{N}$

Importance sampling

- ① Generate N samples $\{\boldsymbol{x}_k^{(i)}\}_{i=1...N}$ from $q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^{(i)}, \boldsymbol{y}_{1:k})$
- Set $\boldsymbol{x}_{0:k}^{(i)} = (\boldsymbol{x}_{0:k-1}^{(i)} \boldsymbol{x}_k^{(i)})$
- ② Associate to each sample its normalised importance weight $\{\tilde{w}_k^{(i)}\}$ using (5.4)

Resampling ③ [If $\frac{N}{\sum_i (\tilde{w}_k^(i))^2} < \epsilon$]

- Generate N new samples among $\{\boldsymbol{x}_{0:k}^{(i)}\}$ depending on their probabilities given by the weights $\{\tilde{w}_k^{(i)}\}$
- Set $\tilde{w}_k^{(i)} = \frac{1}{N}$



5.3.2 Choice of the importance density

Historically, the first proposed particle filter including a resampling step has been built with the following rules:

(a) to set the importance function to the evolution law, i.e.

$$q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^{(i)}, \boldsymbol{y}_{1:k}) = p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^{(i)})$$

(b) to proceed the resampling step at each iteration.

This scheme corresponds to the Bootstrap particle filter.

It is clear that adding a resampling procedure improves the quality of the estimates by reducing the degeneracy problem. However, unnecessary resampling may introduce

its own challenge as samples with higher probability may be oversampled, and regions corresponding to secondary modes of the filtering distribution may be not well explored. Another strategy whose aim is to reduce the degeneracy problem consists in using an *optimal importance function* which minimizes the variance of the weights conditioned upon $\mathbf{x}_{0:k-1}$ and $\mathbf{z}_{1:k}$. It is then possible to prove that choosing:

$$q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{x}^{(i)}_{0:k-1},\boldsymbol{y}_{1:k}) = p_{\mathbf{X}_k|\mathbf{X}_{k-1},\mathbf{Y}_k}(\boldsymbol{x}_k|\boldsymbol{x}^{(i)}_{k-1},\boldsymbol{y}_k)$$

corresponds to this optimal choice (which is rarely possible in practice).

The choice of the importance function is of crucial importance for the quality of the particle filter estimates. As a consequence, the goal of more recent approaches is to design efficient importance functions approximating as closely as possible the optimal one, and to guide the particles in high likelihood areas. These approaches aim also at introducing the measurements into the sampling step.

5.4 Particle filter and high dimension

The particle filter is very efficient for highly nonlinear models but with low dimensionality. The number of required particles typically increases with the system state space dimension. Designing particle filters for data assimilation in high dimension space is still an active research topic.

5.5 Exercices

1. Prove the weight update equation (5.4) from (5.3)
2. What is the weight update in the case of Bootstrap particle filter ?
3. What is the weight update in the case of optimal importance function ?

5.6 Solution

1. Prove the weight update equation (5.4) from (5.3)
   Let us remark that :

$$\begin{aligned}
& p_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k}) \\
= \; & p_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k-1},\boldsymbol{y}_k) \\
= \; & \frac{p_{\mathbf{X}_{0:k},\mathbf{Y}_k|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k},\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1})}{p_{\mathbf{Y}_k|\mathbf{Y}_{k-1}}(\boldsymbol{y}_k|\boldsymbol{y}_{k-1})} \\
\propto \; & p_{\mathbf{Y}_k|\mathbf{X}_{0:k},\mathbf{Y}_{1:k-1}}(\boldsymbol{y}_k|\boldsymbol{x}_{0:k},\boldsymbol{y}_{1:k-1})\; p_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{1:k-1}) \\
\propto \; & p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)\; p_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1},\boldsymbol{y}_{1:k-1})\; p_{\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1}) \\
\propto \; & p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k)\; p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})\; p_{\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{1:k-1})
\end{aligned}$$

then, we have :

$$\begin{aligned} w_k^{(i)} &= \frac{p_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}^{(i)}|\boldsymbol{y}_{1:k})}{q_{\mathbf{X}_{0:k}|\mathbf{Y}_{1:k}}(\boldsymbol{x}_{0:k}^{(i)}|\boldsymbol{y}_{1:k})} \\ &\propto \frac{p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k^{(i)})\ p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(i)})\ p_{\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k-1}^{(i)}|\boldsymbol{y}_{1:k-1})}{q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^{(i)},\boldsymbol{y}_{1:k})\ q_{\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1}}(\boldsymbol{x}_{0:k-1}^{(i)}|\boldsymbol{y}_{1:k-1})} \\ &\propto \widetilde{w}_{k-1}^{(i)}\ \frac{p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k^{(i)})\ p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(i)})}{q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{0:k-1}^{(i)},\boldsymbol{y}_{1:k})} \end{aligned}$$

2. What is the weight update in the case of Bootstrap particle filter ?

$$q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^{(i)},\boldsymbol{y}_{1:k}) = p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^{(i)})$$

$$\begin{aligned} w_k^{(i)} &\propto \widetilde{w}_{k-1}^{(i)}\ \frac{p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k^{(i)})\ p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(i)})}{q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{0:k-1}^{(i)},\boldsymbol{y}_{1:k})} \\ &\propto \widetilde{w}_{k-1}^{(i)} p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k^{(i)}) \end{aligned}$$

as a resampling step is performed at each iteration, $\widetilde{w}_{k-1}^{(i)} = \frac{1}{N} \forall i$ so

$$w_k^{(i)} \propto p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k^{(i)})$$

3. What is the weight update in the case of optimal importance function ?

$$q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^{(i)},\boldsymbol{y}_{1:k}) = p_{\mathbf{X}_k|\mathbf{X}_{k-1},\mathbf{Y}_k}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^{(i)},\boldsymbol{y}_k)$$

We have :

$$p_{\mathbf{X}_k|\mathbf{X}_{k-1},\mathbf{Y}_k}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^{(i)},\boldsymbol{y}_k) = \frac{p_{\mathbf{Y}_k,\mathbf{X}_k,\mathbf{X}_{k-1}}(\boldsymbol{y}_k,\boldsymbol{x}_k^{(i)},\boldsymbol{x}_{k-1}^{(i)})}{p_{\mathbf{Y}_k,\mathbf{X}_{k-1}}(\boldsymbol{y}_k,\boldsymbol{x}_{k-1}^{(i)})}$$

and

$$p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k^{(i)})\ p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(i)}) = \frac{p_{\mathbf{Y}_k,\mathbf{X}_k,\mathbf{X}_{k-1}}(\boldsymbol{y}_k,\boldsymbol{x}_k^{(i)},\boldsymbol{x}_{k-1}^{(i)})}{p_{\mathbf{X}_{k-1}}(\boldsymbol{x}_{k-1}^{(i)})}$$

$$\begin{aligned} w_k^{(i)} &\propto \widetilde{w}_{k-1}^{(i)}\ \frac{p_{\mathbf{Y}_k|\mathbf{X}_k}(\boldsymbol{y}_k|\boldsymbol{x}_k^{(i)})\ p_{\mathbf{X}_k|\mathbf{X}_{k-1}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{k-1}^{(i)})}{q_{\mathbf{X}_k|\mathbf{X}_{0:k-1},\mathbf{Y}_{1:k}}(\boldsymbol{x}_k^{(i)}|\boldsymbol{x}_{0:k-1}^{(i)},\boldsymbol{y}_{1:k})} \\ &\propto \widetilde{w}_{k-1}^{(i)} \frac{p_{\mathbf{Y}_k,\mathbf{X}_{k-1}}(\boldsymbol{y}_k,\boldsymbol{x}_{k-1}^{(i)})}{p_{\mathbf{X}_{k-1}}(\boldsymbol{x}_{k-1}^{(i)})} \\ &\propto \widetilde{w}_{k-1}^{(i)}\ p_{\mathbf{Y}_k|\mathbf{X}_{k-1}}(\boldsymbol{y}_k|\boldsymbol{x}_{k-1}^{(i)}) \end{aligned}$$

6

Elements of optimisation

Chapter outline

6.1 Differential calculus

6.1.1 Directional derivative

Definition 6.1 *Let $f : E \to \mathbb{R}$. We call* Gateaux derivative, *or* directional derivative, *of f at point $\mathbf{a} \in E$ in direction $\mathbf{d} \in E$*

$$\frac{\partial f}{\partial \mathbf{d}}(\mathbf{a}) = \widehat{f}[\mathbf{a}](\mathbf{d}) = \lim_{\alpha \to 0} \frac{f(\mathbf{a} + \alpha \mathbf{d}) - f(\mathbf{a})}{\alpha}$$

Examples:

- partial derivatives $\dfrac{\partial f}{\partial x_i}$ are directional derivatives in the direction of the members of the canonical basis ($d = e_i$)

$$\frac{\partial f}{\partial x_i}(\mathbf{a}) = \lim_{\alpha \to 0} \frac{f(\mathbf{a} + \alpha \mathbf{e}_i) - f(\mathbf{a})}{\alpha}$$

- Let $f(x, y) = 2x^2 + xy + y^2$ and $\mathbf{d} = (1, 2)$.

$$\frac{\partial f}{\partial \mathbf{d}}(x, y) = \lim_{\alpha \to 0} \frac{f(x + \alpha, y + 2\alpha) - f(x, y)}{\alpha} = \lim_{\alpha \to 0} 6x + 5y + 8\alpha = 6x + 5y$$

6.1.2 Fréchet derivative, gradient, jacobian

Definition 6.2 *Let E be a Hilbert space. Let f be a function from E to $\mathbb{R}$. We say that f is* Fréchet differentiable *at point $\mathbf{a} \in E$ if there exists $\mathbf{p_a} \in E$ such that*

$$f(\mathbf{a} + \mathbf{d}) = f(\mathbf{a}) + \langle \mathbf{p_a}, \mathbf{d} \rangle + o(\|\mathbf{d}\|) \quad \forall \mathbf{d} \in E \tag{6.1}$$

Then $\mathbf{p_a}$ is called the derivative *or the* gradient *of f in $\mathbf{a}$, denoted $\nabla f(\mathbf{a})$.*

Definition 6.3 *The function $\mathbf{d} \to \langle \nabla f(\mathbf{a}), \mathbf{d} \rangle$ is a* **linear** *application called* differential function *or* tangent linear function *of f at point $\mathbf{a}$.*

Examples:

- In finite dimension ($E = \mathbb{R}^n$), the gradient of f in $\mathbf{a}$ is simply

$$\nabla f(\mathbf{a}) = \begin{pmatrix} \dfrac{\partial f}{\partial x_1}(\mathbf{a}) \\ \vdots \\ \dfrac{\partial f}{\partial x_n}(\mathbf{a}) \end{pmatrix}$$

- For the previous exemple $f(x, y) = 2x^2 + xy + y^2$, we have

$$\nabla f(x, y) = \begin{pmatrix} 4x + y \\ x + 2y \end{pmatrix}$$

Very important property:
if the gradient exists, then the directional derivative in direction $\mathbf{d}$ is

$$\frac{\partial f}{\partial \mathbf{d}}(\mathbf{a}) = \widehat{f}[\mathbf{a}](\mathbf{d}) = \langle \nabla f(\mathbf{a}), \mathbf{d} \rangle \tag{6.2}$$

Examples:

- Going back to $f(x,y) = 2x^2 + xy + y^2$. Its directional derivative in direction $\mathbf{d} = (1,2)$ is $6x + 5y$ and its gradient is $\nabla f(x,y) = \begin{pmatrix} 4x+y \\ x+2y \end{pmatrix}$ so we can verify that

$$\langle \nabla f(x,y), \mathbf{d} \rangle = \begin{pmatrix} 4x+y \\ x+2y \end{pmatrix} . \begin{pmatrix} 1 \\ 2 \end{pmatrix} = (4x+y) + 2(x+2y) = 6x + 5y = \frac{\partial f}{\partial \mathbf{d}}(x,y)$$

Definition 6.4 *Let E and F be two Hilbert spaces. Let f be a function from E to F (not necessary $\mathbb{R}$). We call differential of f at point $\mathbf{a} \in E$ the linear function $\mathbf{F}_{[\mathbf{a}]}$, defined by:*

$$f(\mathbf{a} + \mathbf{d}) = f(\mathbf{a}) + \mathbf{F}_{[\mathbf{a}]}(\mathbf{d}) + o(\|\mathbf{d}\|) \quad \forall \mathbf{d} \in E \tag{6.3}$$

When E and F are of finite dimension, for exemple f from $\mathbb{R}^n$ to $\mathbb{R}^p$, $\mathbf{F}_{[\mathbf{a}]}(\mathbf{d}) = \mathbf{F}_{[\mathbf{a}]}\mathbf{d}$, where $\mathbf{F}_{[\mathbf{a}]}$ is the Jacobian matrix of f at point $\mathbf{a}$ (also called linear tangent operator).

$$f : \begin{array}{l} \mathbb{R}^n \longrightarrow \mathbb{R}^p \\ \mathbf{x} \longrightarrow \begin{pmatrix} f_1(x_1, \ldots, x_n) \\ \vdots \\ f_p(x_1, \ldots, x_n) \end{pmatrix} \end{array} \qquad \mathbf{F}_{[\mathbf{a}]} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{a}) & \ldots & \frac{\partial f_1}{\partial x_n}(\mathbf{a}) \\ \vdots & & \vdots \\ \frac{\partial f_p}{\partial x_1}(\mathbf{a}) & \ldots & \frac{\partial f_p}{\partial x_n}(\mathbf{a}) \end{pmatrix}$$

.

6.2 Optimisation algorithm

We consider the following problem:

Problem 6.5 *Find the minimum $\boldsymbol{x}^*$:*

$$J(\boldsymbol{x}^*) = \min_{\boldsymbol{x} \in \mathbb{R}^n} J(\boldsymbol{x})$$

6.2.1 When J has a quadratic form

Let $\mathbf{A}$ be a $p \times n$ matrix, of rank n ($p \geq n$), and $\mathbf{b}$ a vector of $\mathbb{R}^p$. Let $\mathbf{N}$ be a symmetric positive definite matrix of size $p \times p$. Let the function J of $\mathbb{R}^n$ in $\mathbb{R}$ defined by

$$J(\boldsymbol{x}) = \|\mathbf{A}\boldsymbol{x} - \mathbf{b}\|^2_{\mathbf{N}} = (\mathbf{A}\boldsymbol{x} - \mathbf{b})^T \mathbf{N} (\mathbf{A}\boldsymbol{x} - \mathbf{b})$$

J has a quadratic form. It is minimum for

$$\boldsymbol{x}^* = (\mathbf{A}^T \mathbf{N}\, \mathbf{A})^{-1} \mathbf{A}^T \mathbf{N} \mathbf{b} \tag{6.4}$$

6.2.2 Descent algorithms

For more complex function J, descent algorithms can be used for minimisation. The minimisation algorithms start at a point $\boldsymbol{x}_{(0)}$ and build a sequence of points $\boldsymbol{x}_{(k)}$ which is meant to converge to a local minimum. $\boldsymbol{x}_{(0)}$ must be in the basin of attraction of the local minimum. At step k of the algorithm, we determine a direction $\boldsymbol{d}_k$ which is characteristic of the method. This direction is used to define the next point of the sequence

$$\boldsymbol{x}_{(k+1)} = \boldsymbol{x}_{(k)} + \alpha_k \boldsymbol{d}_k \text{ such that } J(\boldsymbol{x}_{(k+1)}) < J(\boldsymbol{x}_{(k)})$$

- $\boldsymbol{d}_k \in \mathbb{R}^n$ is the direction descent at iteration k,
- $\alpha_k \in \mathbb{R}$ is the descent step at iteration k.

There exist many methods, each one corresponding to a specific choice of α_k ands $\boldsymbol{d}_k$, for exemple :

- gradient method: $\boldsymbol{d}_k = -\nabla J(\boldsymbol{x}_{(k)})$
- Newton method: $\boldsymbol{d}_k = \left[\nabla^2 J(\boldsymbol{x}_k)\right]^{-1} \nabla J(\boldsymbol{x}_k)$
  where $\nabla^2 J(x_k)$ is the Hessian matrix of J.
- conjugate gradient: $\boldsymbol{d}_k = -\nabla J(\boldsymbol{x}_{(k)}) + \boldsymbol{d}_{k-1} \dfrac{\|\nabla J(\boldsymbol{x}_{(k)})\|^2}{\|\nabla J(\boldsymbol{x}_{(k-1)})\|^2}$
- etc.

One important point here is to notice that they all need an estimation of the gradient value $\nabla J(\boldsymbol{x}_{(k)})$. This is detailled next section

6.2.3 Getting the gradient is not always obvious

if the size N of the state variable is very small (< 10), ∇J can be easily estimated by the computation of growth rates:

$$\nabla J(\boldsymbol{x}) = \begin{pmatrix} \dfrac{\partial J}{\partial x_1}(\boldsymbol{x}) \\ \vdots \\ \dfrac{\partial J}{\partial x_N}(\boldsymbol{x}) \end{pmatrix} \simeq \begin{pmatrix} \left[J(\boldsymbol{x} + \alpha\, \mathbf{e}_1) - J(\boldsymbol{x})\right]/\alpha \\ \vdots \\ \left[J(\boldsymbol{x} + \alpha\, \mathbf{e}_N) - J(\boldsymbol{x})\right]/\alpha \end{pmatrix} \tag{6.5}$$

This calculation requires $N + 1$ run of the model, that can be prohibitif in actual applications like meteorology / oceanography where $N = [\mathbf{u}] = \mathcal{O}(10^6 - 10^9)$. Alternatively, the adjoint method provides a very efficient way to compute ∇J. It will be introduced in the next chapter ... but let's do some exercices first.

6.3 Exercices

exercice 1

Let $\mathbf{A}$ be a $p \times n$ matrix, of rank n ($p \geq n$), and $\mathbf{b}$ a vector of $\mathbb{R}^p$. Let the function J of $\mathbb{R}^n$ in $\mathbb{R}$ defined by

$$J(\boldsymbol{x}) = \|\mathbf{A}\boldsymbol{x} - \mathbf{b}\|^2 = (\mathbf{A}\boldsymbol{x} - \mathbf{b})^T(\mathbf{A}\boldsymbol{x} - \mathbf{b})$$

1. Compute the directional derivative of J in $\boldsymbol{x}$ in direction $\boldsymbol{d} = \delta\boldsymbol{x}$
2. Deduce the gradient of J
3. Deduce the expression of $\boldsymbol{x}^* = \min_{\boldsymbol{x}} J(\boldsymbol{x})$

exercice 2

Let $\mathbf{A}$ be a $p \times n$ matrix, of rank n ($p \geq n$), and $\mathbf{b}$ a vector of $\mathbb{R}^p$. Let $\mathbf{N}$ be a symmetric positive definite matrix of size $p \times p$. Let the function J of $\mathbb{R}^n$ in $\mathbb{R}$ defined by

$$J_1(\boldsymbol{x}) = \|\mathbf{A}\boldsymbol{x} - \mathbf{b}\|^2_{\mathbf{N}} = (\mathbf{A}\boldsymbol{x} - \mathbf{b})^T\mathbf{N}(\mathbf{A}\boldsymbol{x} - \mathbf{b})$$

1. Deduce the gradient of J_1 from the result of exercice 1.
2. Deduce the expression of $\boldsymbol{x}^* = \min_{\boldsymbol{x}} J_1(\boldsymbol{x})$

exercice 3

Let X and Y be two Hilbert spaces and $\langle ., . \rangle_X$, $\langle ., . \rangle_Y$ the associated scalar products. Let $\boldsymbol{x} \in X$, $\boldsymbol{y} \in Y$ and $\boldsymbol{d} \in X$. Let g an application from X to Y differentiable. We define f from X to $\mathbb{R}$ by

$$f(\boldsymbol{x}) = \|g(\boldsymbol{x}) - \boldsymbol{y}\|^2$$

1. Compute the directional derivative of f in $\boldsymbol{x}$ in direction $\boldsymbol{d} = \delta\boldsymbol{x}$
2. Deduce its gradient.

6.4 Solutions

exercice 1

$$\begin{aligned} J(\boldsymbol{x} + \alpha\,\delta\boldsymbol{x}) - J(\boldsymbol{x}) &= (\mathbf{A}(\boldsymbol{x} + \alpha\,\delta\boldsymbol{x}) - \mathbf{b})^T(\mathbf{A}(\boldsymbol{x} + \alpha\,\delta\boldsymbol{x}) - \mathbf{b}) - (\mathbf{A}\boldsymbol{x} - \mathbf{b})^T(\mathbf{A}\boldsymbol{x} - \mathbf{b}) \\ &= \alpha\,\left((\mathbf{A}\,\delta\boldsymbol{x})^T(\mathbf{A}\boldsymbol{x} - \mathbf{b}) + (\mathbf{A}\boldsymbol{x} - \mathbf{b})^T\mathbf{A}\,\delta\boldsymbol{x}\right) + \alpha^2\,(\mathbf{A}\,\delta\boldsymbol{x})^T\mathbf{A}\,\delta\boldsymbol{x} \end{aligned}$$

Then $\lim\limits_{\alpha\to 0} \dfrac{J(\boldsymbol{x} + \alpha\,\delta\boldsymbol{x}) - J(\boldsymbol{x})}{\alpha} = (\mathbf{A}\,\delta\boldsymbol{x})^T(\mathbf{A}\boldsymbol{x} - \mathbf{b}) + (\mathbf{A}\boldsymbol{x} - \mathbf{b})^T\mathbf{A}\,\delta\boldsymbol{x}$.
The 2 terms in the sum are equal (they are real numbers), so :

$$\begin{aligned} \widehat{J}[\boldsymbol{x}](\delta\boldsymbol{x}) &= 2\,(\mathbf{A}\boldsymbol{x} - \mathbf{b})^T\mathbf{A}\,\delta\boldsymbol{x} \\ &= 2\,\langle \mathbf{A}\boldsymbol{x} - \mathbf{b}, \mathbf{A}\,\delta\boldsymbol{x}\rangle \\ &= 2\,\left\langle \mathbf{A}^T(\mathbf{A}\boldsymbol{x} - \mathbf{b}), \delta\boldsymbol{x}\right\rangle \end{aligned}$$

And, using (6.2), $\widehat{J}[\boldsymbol{x}](\delta\boldsymbol{x}) = \langle \nabla J(\boldsymbol{x}), \delta\boldsymbol{x} \rangle$ we can deduce:

$$\nabla J(\boldsymbol{x}) = 2\,\mathbf{A}^T(\mathbf{A}\boldsymbol{x} - \mathbf{b}) \tag{6.6}$$

J is quadratic, and admits a unique minimum $\boldsymbol{x}^*$, such that $\nabla J(\boldsymbol{x}^*) = 0$.

$$\nabla J(\boldsymbol{x}^*) = 0 \Leftrightarrow \mathbf{A}^T\mathbf{A}\boldsymbol{x}^* = \mathbf{A}^T\mathbf{b}$$

$\mathbf{A}^T\mathbf{A}$ is a square matrix of size n and rank n and is inversible. Therefore

$$\boldsymbol{x}^* = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} \tag{6.7}$$

exercice 2

$\mathbf{N}$ is a symmetric positive definite matrix, so we can write $\mathbf{N} = \mathbf{N}^{1/2}\mathbf{N}^{1/2}$ where $\mathbf{N}^{1/2}$ is also a symmetric positive definite matrix of size $p \times p$. Then

$$\begin{aligned} J_1(\mathbf{x}) &= (\mathbf{A}\boldsymbol{x} - \mathbf{b})^T\mathbf{N}(\mathbf{A}\boldsymbol{x} - \mathbf{b}) \\ &= (\mathbf{A}\boldsymbol{x} - \mathbf{b})^T\mathbf{N}^{1/2}\mathbf{N}^{1/2}(\mathbf{A}\boldsymbol{x} - \mathbf{b}) \\ &= (\mathbf{N}^{1/2}\mathbf{A}\boldsymbol{x} - \mathbf{N}^{1/2}\mathbf{b})^T(\mathbf{N}^{1/2}\mathbf{A}\boldsymbol{x} - \mathbf{N}^{1/2}\mathbf{b}) \end{aligned}$$

We are brought back to (7.5), by replacing $\mathbf{A}$ by $\mathbf{N}^{1/2}\mathbf{A}$ and $\mathbf{b}$ by $\mathbf{N}^{1/2}\mathbf{b}$. Then:

$$\begin{aligned} \nabla J_1(\boldsymbol{x}) &= 2\,(\mathbf{N}^{1/2}\mathbf{A})^T(\mathbf{N}^{1/2}\mathbf{A}\boldsymbol{x} - \mathbf{N}^{1/2}\mathbf{b}) \\ &= 2\,\mathbf{A}^T\mathbf{N}^{1/2}\mathbf{N}^{1/2}(\mathbf{A}\boldsymbol{x} - \mathbf{b}) \\ &= 2\,\mathbf{A}^T\mathbf{N}(\mathbf{A}\boldsymbol{x} - \mathbf{b}) \end{aligned}$$

and (6.7) becomes

$$\boldsymbol{x}^* = (\mathbf{A}^T\mathbf{N}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{N}\mathbf{b}$$

exercice 3

We use (6.3)

$$g(\boldsymbol{x} + \alpha\delta\boldsymbol{x}) = g(\boldsymbol{x}) + \alpha\mathbf{G}_{[\boldsymbol{x}]}\delta\boldsymbol{x} + o(\alpha)$$

where $\mathbf{G}_{[\boldsymbol{x}]}$ is the jacobian of g in $\boldsymbol{x}$, that is a linear application from X to Y. Then it is similar as exercice 1 and the result is $\nabla f(\boldsymbol{x}) = 2\mathbf{G}^T_{[\boldsymbol{x}]}(g(\boldsymbol{x}) - \boldsymbol{y})$.

7

Variational approaches

Chapter outline

Let us recall that Data assimilation is the process by which observational data are fused with scientific information. Several sources of information may be available on the system we are interested in :

- observations $\boldsymbol{y}$ and associated mapping $\mathcal{H}$
- an evolution model $\mathcal{M}$
- a background $\boldsymbol{x}_b$ of the state of the system at an initial moment (which therefore of course allows, by propagating it by the model $\mathcal{M}$, to have a draft of the whole trajectory of the system)
- all associated errors

In the two previous chapter, we have been interested in the Bayesian approach to the problem, deducing from the optimal filter the Kalman filter, the Ensemble Kalman filter and the particle filter.

Let us know turn to the variational approach. As explained in §3.3.1 finding the best possible estimate can be done through the definition of a functional quantifying a discrepancy between the state of the system $\boldsymbol{x}$ and the two available sources of information, i.e. the background $\boldsymbol{x}_b$ and the observations $\boldsymbol{y}$. We then look for the optimal state $\boldsymbol{x}^a$, also called *analysis*, which minimizes this functional.

$$\boldsymbol{x}^a = \min_{\boldsymbol{x}} J(\boldsymbol{x})$$

where $J(\boldsymbol{x})$ is a cost function defined to measure the misfit between available information (background and observation) and model.

7.1 Expression of the cost function J

In the general case we aim at estimating the trajectory $\boldsymbol{x} = [\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_K]$ by minimizing the cost function

$$J(\boldsymbol{x}) = J^b(\boldsymbol{x}) + J^q(\boldsymbol{x}) + J^o(\boldsymbol{x})$$

- $J^b(\boldsymbol{x})$ stands for the *a priori* term,

$$J^b(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}}$$

  The notation $\|x\|^2_A = x^T A x$ is used to denote the norm associated to the scalar product defined by the symetric positive definite matrix A.
- $J^o(\boldsymbol{x})$ stands for the *data* term

$$J^o(\boldsymbol{x}) = \frac{1}{2}\sum_{k=1}^{K} \|\mathcal{H}_k(\boldsymbol{x}_k) - \boldsymbol{y}_k\|^2_{\mathbf{R}_k^{-1}}$$

- $J^q(\boldsymbol{x})$ stands for the *model* term

$$J^q(\boldsymbol{x}) = \frac{1}{2}\sum_{k=1}^{K} \|\boldsymbol{x}_k - \mathcal{M}_{k-1\to k}(\boldsymbol{x}_{k-1})\|^2_{\mathbf{Q}_k^{-1}}$$

When the cost function has been set, then the problem is entirely defined, and so is its solution. The "physical" part lies in the definition of J, the choice of the covariance matrices, the background, etc. Finding the solution once the cost function has been defined is "only" technical work.

With a perfect model In practice, in the variational framework, the model is almost always supposed to be perfect. In that case, the term J^q is equal to 0 ; and the problem of estimating $\boldsymbol{x}$ is brought back to a proper estimation of the initial condition $\boldsymbol{x}_0$, with

$$J(\boldsymbol{x}_0) = J^b(\boldsymbol{x}_0) + J^o(\boldsymbol{x}_0) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\sum_{k=1}^{K}\|\mathcal{H}_k(\mathcal{M}_{0\to k}(\boldsymbol{x}_0)) - \boldsymbol{y}_k\|^2_{\mathbf{R}_k^{-1}} \tag{7.1}$$

In the simple stationary case we have:

$$J(\boldsymbol{x}_0) = J^b(\boldsymbol{x}_0) + J^o(\boldsymbol{x}_0) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\|\mathcal{H}(\boldsymbol{x}_0) - \boldsymbol{y}\|^2_{\mathbf{R}^{-1}}$$

7.1.1 link with the Bayesian approach

Finding the minimum of the cost function $J(\boldsymbol{x}_0)$ as in (7.1) (i.e. with a perfect model) is equivalent to find the maximum of the probability distribution $p_{\mathbf{X}_0|\mathbf{Y}_{1:k}}(\boldsymbol{x}_0|y_{1:k})$:

$$p_{\mathbf{X}_0|\mathbf{Y}_{1:k}}(\boldsymbol{x}_0|\boldsymbol{y}_{1:k}) \propto p_{\mathbf{Y}_{1:k}|\mathbf{X}_0}(\boldsymbol{y}_{1:k}|\boldsymbol{x}_0)\ p_{\mathbf{X}_0}(\boldsymbol{x}_0)$$

with

$$\begin{aligned} p_{\mathbf{Y}_{1:k}|\mathbf{X}_0}(\boldsymbol{y}_{1:k}|\boldsymbol{x}_0) &= \prod_{k=1}^{K} p_{\mathbf{Y}_k|\mathbf{X}_0}(\boldsymbol{y}_k|\boldsymbol{x}_0) \\ &= \prod_{k=1}^{K} p_{\mathbf{Y}_k|\mathbf{X}_n}(\boldsymbol{y}_k|\boldsymbol{x}_n = \mathcal{M}_{0\to k}(\boldsymbol{x}_0)) \end{aligned} \tag{7.2}$$

Now by making the assumptions that $\mathbf{X}_0 \sim \mathcal{N}(\mathbf{X}_0; \boldsymbol{x}_b, \mathbf{P}_b)$ and $\mathbf{Y}_k|\mathbf{X}_k \sim \mathcal{N}(\mathbf{Y}_k; \mathcal{H}_k\,\mathbf{X}_k, \mathbf{R}_k)$, we get:

$$p_{\mathbf{X}_0|\mathbf{Y}_{1:k}}(\boldsymbol{x}_0|y_{1:k}) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\sum_{k=1}^{K}\|\mathcal{H}_k(\mathcal{M}_{0\to k}(\boldsymbol{x}_0)) - \boldsymbol{y}_k\|^2_{\mathbf{R}_k^{-1}}\right)$$

i.e. $p_{\mathbf{X}_0|\mathbf{Y}_{1:k}}(\boldsymbol{x}_0|y_{1:k}) \propto \exp -J(\boldsymbol{x}_0)$.

7.2 Minimising J

7.2.1 One minimum, several minima ?

We will consider the case where the cost function is composed of 2 terms, J^b and J^o. J^b has the good property of being convex ; whereas the nature of J^o depends on the nature of the operator $\mathcal{H}$ and $\mathcal{M}$.

- If $\mathcal{H}$ and $\mathcal{M}$ are linear. Considering only J^o, the optimisation problem can be underdetermined, if the number of observations is lower than the size of the unknown. Adding J^b leads the problem to be well posed. The cost function is quadratic.
- If $\mathcal{H}$ and $\mathcal{M}$ are non both linear, J^o is no longer convex, and may admit several local minima. Adding J^b ease the problem by regularising it but without fundamentaly changing it.

7.2.2 Introduction to the adjoint method

Previously seen optimisation methods require the computation of ∇J. If the dependance of J on the control variable $\boldsymbol{x}$ is complex and/or indirect, this computation can be complicated. Numerically, we can still use growth rate approximations. But this allows only to compute one directional derivative, and we would need to do this for each component of the gradient. If the dimension of the state is large, it will be costful. In practical applications, it is usually impossible to do so, e.g., in oceanography and meteorology, the initial state space dimension is larger than 10^7.

Let us take the case for J has the following form :

$$J(\boldsymbol{x}) = \frac{1}{2}\sum_{k=1}^{K} \|\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k\|^2_{R_k^{-1}} = \frac{1}{2}\sum_{k=1}^{K} \left(\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k\right)^T \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k\right)$$

Let us give the expression of the directionnal derivative of J in $\boldsymbol{x}$. Denoting by $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \alpha\,\delta\boldsymbol{x}$, we have

$$\begin{aligned}
\frac{J(\boldsymbol{x} + \alpha\,\delta\boldsymbol{x}) - J(\boldsymbol{x})}{\alpha} &= \frac{1}{2\alpha}\sum_{k=1}^{K} \left(\mathcal{G}_k(\widetilde{\boldsymbol{x}}) - \boldsymbol{y}_k\right)^T \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\widetilde{\boldsymbol{x}}) - \boldsymbol{y}_k\right) \\
&\qquad -\left(\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k\right)^T \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k\right) \\
&= \frac{1}{2}\sum_{k=1}^{K} \left(\frac{\mathcal{G}_k(\widetilde{\boldsymbol{x}}) - \mathcal{G}_k(\boldsymbol{x})}{\alpha}\right)^T \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\widetilde{\boldsymbol{x}}) - \boldsymbol{y}_k\right) \\
&\quad + \frac{1}{2}\sum_{k=1}^{K} \left(\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k\right)^T \mathbf{R}_k^{-1} \left(\frac{\mathcal{G}_k(\widetilde{\boldsymbol{x}}) - \mathcal{G}_k(\boldsymbol{x})}{\alpha}\right) \\
&= \sum_{k=1}^{K} \left(\frac{\mathcal{G}_k(\widetilde{\boldsymbol{x}}) - \mathcal{G}_k(\boldsymbol{x})}{\alpha}\right)^T \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\widetilde{\boldsymbol{x}}) - \boldsymbol{y}_k\right)
\end{aligned}$$

Using (6.3), we have

$$\mathcal{G}_k(\widetilde{\boldsymbol{x}}) = \mathcal{G}_k(\boldsymbol{x}) + \alpha\ \mathbf{G}_{k[\boldsymbol{x}]}\ \delta\boldsymbol{x} + o\left(\|\delta\boldsymbol{x}\|^2\right)$$

where $\mathbf{G}_{k[\boldsymbol{x}]}$ is the jacobian matrix of $\mathcal{G}_k$ at point $\boldsymbol{x}$. Then,

$$\lim_{\alpha \to 0} \frac{\mathcal{G}_k(\widetilde{\boldsymbol{x}}) - \mathcal{G}_k(\boldsymbol{x})}{\alpha} = \mathbf{G}_{k[\boldsymbol{x}]} \delta \boldsymbol{x} \tag{7.3}$$

and

$$\widehat{J}[\boldsymbol{x}](\delta \boldsymbol{x}) = \sum_{k=1}^{K} (\mathbf{G}_{k[\boldsymbol{x}]} \delta \boldsymbol{x})^T \, \mathbf{R}_k^{-1} \left[\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k \right] \tag{7.4}$$

$$\begin{aligned} \widehat{J}[\boldsymbol{x}](\delta \boldsymbol{x}) &= \delta \boldsymbol{x}^T \textstyle\sum_{k=1}^{K} \mathbf{G}_{k[\boldsymbol{x}]}^T \, \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k \right) \\ &= < \delta \boldsymbol{x} \, , \sum_{k=1}^{K} \mathbf{G}_{k[\boldsymbol{x}]}^T \, \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k \right) > \end{aligned}$$

And, using (6.2), $\widehat{J}[\boldsymbol{x}](\delta \boldsymbol{x}) = \langle \nabla J(\boldsymbol{x}), \delta \boldsymbol{x} \rangle = \langle \delta \boldsymbol{x}, \nabla J(\boldsymbol{x}) \rangle$ we can deduce:

$$\nabla J(\boldsymbol{x}) = \sum_{k=1}^{K} \mathbf{G}_{k[\boldsymbol{x}]}^T \, \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\boldsymbol{x}) - \boldsymbol{y}_k \right) \tag{7.5}$$

$\mathbf{G}_{k[\boldsymbol{x}]}^T$ is called the adjoint operator, and corresponds to the transpose of the Jacobian of $\mathcal{G}_k$ at point $\boldsymbol{x}$.

7.2.3 The adjoint operator

Let us take as exemple :

$$J(\boldsymbol{x}_0) = \frac{1}{2} \sum_{k=1}^{K} \| \mathcal{G}_k(\boldsymbol{x}_0) - \boldsymbol{y}_k \|_{R_k^{-1}}^2$$

with

$$\mathcal{G}_k(\boldsymbol{x}_0) = \mathcal{H}_k(\mathcal{M}_{0 \to k}(\boldsymbol{x}_0)) = \mathcal{H}_k \circ \mathcal{M}_{0 \to k}(\boldsymbol{x}_0) = \mathcal{H}_k \circ \mathcal{M}_{k-1 \to k} \circ \cdots \circ \mathcal{M}_{0 \to 1}(\boldsymbol{x}_0)$$

Let us notice that each application of $\mathcal{M}_{k-1 \to k}$ corresponds to the integration of the model, wich can be a big peace of code for some application, e.g. oceanography or meteorology. Then

$$\nabla J(\boldsymbol{x}_0) = \sum_{k=1}^{K} \mathbf{G}_{k[\boldsymbol{x}_0]}^T \, \mathbf{R}_k^{-1} \left(\mathcal{G}_k(\boldsymbol{x}_0) - \boldsymbol{y}_k \right) \tag{7.6}$$

Denoting by $\mathbf{H}_k$, respectively $\mathbf{M}_{k-1,k}$ the jacobian of $\mathcal{H}_k$, respectively $\mathcal{M}_{k-1 \to k}$, for all k, we have

$$\mathbf{G}_{k[\boldsymbol{x}_0]} = \mathbf{H}_{k[\boldsymbol{x}_k]} \; \mathbf{M}_{k-1,k[\boldsymbol{x}_{k-1}]} \ldots \mathbf{M}_{0,1[\boldsymbol{x}_0]}$$

where $\boldsymbol{x}_k = \mathcal{M}_{0 \to k}(\boldsymbol{x}_0)$ results from the model integration. Then the adjoint operator writes

$$\mathbf{G}_{k[\boldsymbol{x}_0]}^T = \mathbf{M}_{0,1[\boldsymbol{x}_0]}^T \ldots \mathbf{M}_{k-1,k[\boldsymbol{x}_{k-1}]}^T \; \mathbf{H}_{k[\boldsymbol{x}_k]}^T$$

It is important to notice that the adjoint operator implies a reverse operation in time. At this point, it seems that an adjoint operator has to be apply for each elements of the sum in (7.6). However, the interest in using this gradient formulae is the fact that

computations can be factorised. Let us take the example of $K = 3$, and 3 observations $\boldsymbol{y}_1$, $\boldsymbol{y}_2$ and $\boldsymbol{y}_3$:

$$\begin{aligned}
\nabla J(\boldsymbol{x}_0) &= \mathbf{G}_{3[\boldsymbol{x}_0]}^T \mathbf{R}_3^{-1} \left(\mathcal{G}_3(\boldsymbol{x}_0) - \boldsymbol{y}_3\right) + \mathbf{G}_{2[\boldsymbol{x}_0]}^T \mathbf{R}_2^{-1} \left(\mathcal{G}_2(\boldsymbol{x}_0) - \boldsymbol{y}_2\right) + \mathbf{G}_{1[\boldsymbol{x}_0]}^T \mathbf{R}_1^{-1} \left(\mathcal{G}_1(\boldsymbol{x}_0) - \boldsymbol{y}_1\right) \\
&= \mathbf{M}_{0,1[\boldsymbol{x}_0]}^T \mathbf{M}_{1,2[\boldsymbol{x}_1]}^T \mathbf{M}_{2,3[\boldsymbol{x}_2]}^T \mathbf{H}_{3[\boldsymbol{x}_3]}^T \mathbf{R}_3^{-1} \left(\mathcal{G}_3(\boldsymbol{x}_0) - \boldsymbol{y}_3\right) \\
&\quad + \mathbf{M}_{0,1[\boldsymbol{x}_0]}^T \mathbf{M}_{1,2[\boldsymbol{x}_1]}^T \mathbf{H}_{2[\boldsymbol{x}_2]}^T \mathbf{R}_2^{-1} \left(\mathcal{G}_2(\boldsymbol{x}_0) - \boldsymbol{y}_2\right) \\
&\quad + \mathbf{M}_{0,1[\boldsymbol{x}_0]}^T \mathbf{H}_{1[\boldsymbol{x}_1]}^T \mathbf{R}_1^{-1} \left(\mathcal{G}_1(\boldsymbol{x}_0) - \boldsymbol{y}_1\right)
\end{aligned}$$

$$\begin{aligned}
\nabla J(\boldsymbol{x}_0) &= \mathbf{M}_{0,1[\boldsymbol{x}_0]}^T \\
&\quad \Big[\mathbf{M}_{1,2[\boldsymbol{x}_1]}^T \\
&\quad \left\{\mathbf{M}_{2,3[\boldsymbol{x}_2]}^T \mathbf{H}_{3[\boldsymbol{x}_3]}^T \mathbf{R}_3^{-1} \left(\mathcal{G}_3(\boldsymbol{x}_0) - \boldsymbol{y}_3\right) + \mathbf{H}_{2[\boldsymbol{x}_2]}^T \mathbf{R}_2^{-1} \left(\mathcal{G}_2(\boldsymbol{x}_0) - \boldsymbol{y}_2\right)\right\} \\
&\quad + \mathbf{H}_{1[\boldsymbol{x}_1]}^T \mathbf{R}_1^{-1} \left(\mathcal{G}_1(\boldsymbol{x}_0) - \boldsymbol{y}_1\right)\Big]
\end{aligned}$$

In order to ease the tangent and adjoint coding, several automatic differentiation tools are available. They take as input your direct model code and create the tangent and adjoint model (TAM) codes.

7.3 Algorithms in practice - Stationary case

Let us make the assumption that the model is stationary. Then $\boldsymbol{x} = \boldsymbol{x}_0$ and the cost function is :

$$J(\boldsymbol{x}_0) = J^b(\boldsymbol{x}_0) + J^o(\boldsymbol{x}_0) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\|\mathcal{H}(\boldsymbol{x}_0) - \boldsymbol{y}\|^2_{\mathbf{R}^{-1}}$$

7.3.1 Linear stationary case

As seen before, when $\mathcal{H}$ is linear, J has a quadratic form. Minimising J is a least-square problem, for which we know the exact solution. the cost function is

$$J(\boldsymbol{x}_0) = J^b(\boldsymbol{x}_0) + J^o(\boldsymbol{x}_0) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\|\mathbf{H}\boldsymbol{x}_0 - \boldsymbol{y}\|^2_{\mathbf{R}^{-1}}$$

by replacing $\mathbf{A}, \mathbf{b}$ and $\mathbf{N}$ in (6.4) such as

$$\mathbf{A} = \begin{pmatrix} \mathbf{I}_n \\ \mathbf{H} \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} \boldsymbol{x}^b \\ \boldsymbol{y} \end{pmatrix} \qquad \mathbf{N} = \begin{pmatrix} \mathbf{P}_b^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}^{-1} \end{pmatrix} \tag{7.7}$$

we get

$$\boldsymbol{x}^a = \boldsymbol{x}^b + (\mathbf{P}_b^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1}(\boldsymbol{y} - \mathbf{H}\boldsymbol{x}^b) \tag{7.8}$$

With the help of the Sherman-Morrison-Woodbury formula, we can recognise the Kalman filter update equation.

Let us also remark that in high dimensional problem, one may need the expression of the gradient of the of J which is given by

$$\nabla J(\boldsymbol{x}_0) = \mathbf{P}_b^{-1}(\boldsymbol{x}_0 - \boldsymbol{x}^b) + \mathbf{H}^T\mathbf{R}^{-1}(\mathbf{H}\boldsymbol{x}_0 - \boldsymbol{y}) \tag{7.9}$$

7.3.2 Non linear stationary case: 3D-Var algorithm

Let us know consider the case when $\mathcal{H}$ is non-linear. For time depending problems, the 3D-Var algorithm is a simplification of the *full* variational data assimilation scheme, making the assumption that the model is stationary over the assimilation window. This algorithm is more dedicated to stationary problems, however it has been used for long (and still is) for non stationary problems but with large dimension. In that case, the observations $\boldsymbol{y}$ although depending on time, are considered as observation of the initial time.

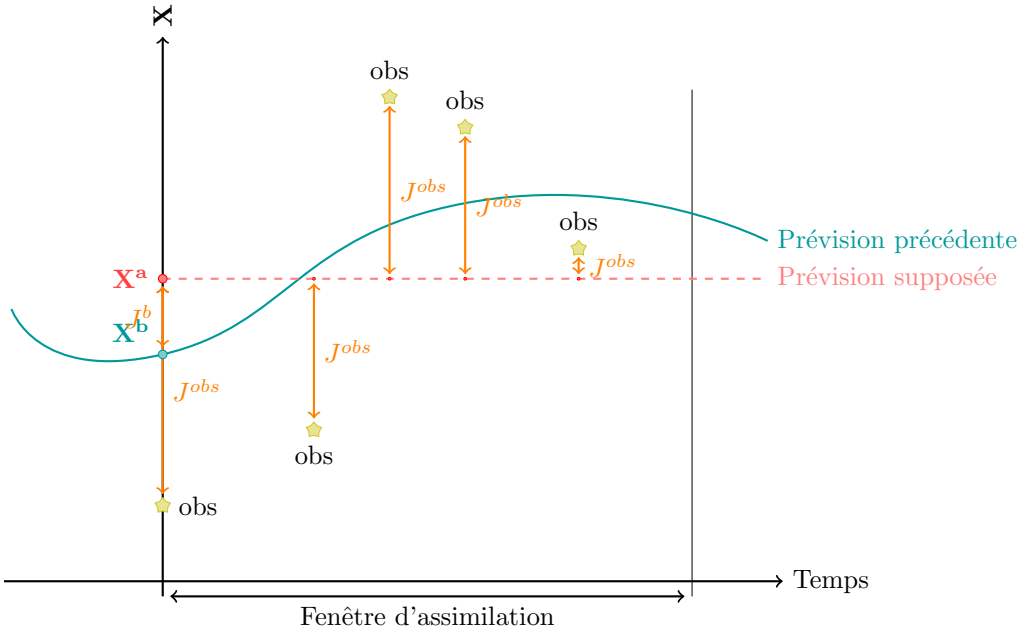


Figure 7.1: 3D-var

The cost function of 3D-Var is:

$$\begin{aligned} J(\boldsymbol{x}_0) &= J^b(\boldsymbol{x}_0) + J^o(\boldsymbol{x}_0) \\ &= \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\|\mathcal{H}(\boldsymbol{x}_0) - \boldsymbol{y}\|^2_{\mathbf{R}^{-1}} \end{aligned}$$

The gradient of J is given by

$$\nabla J(\boldsymbol{x}_0) = \mathbf{P}_b^{-1}(\boldsymbol{x}_0 - \boldsymbol{x}^b) + \mathbf{H}^T_{[\boldsymbol{x}_0]}\mathbf{R}^{-1}(\mathcal{H}(\boldsymbol{x}_0) - \boldsymbol{y})$$

where $\mathbf{H}_{[\boldsymbol{x}_0]}$ is the Jacobian of $\mathcal{H}$ at $\boldsymbol{x}_0$.

The minimization algorithm is a gradient-descent iterative algorithm. It uses the fact that ∇J is small enough as stopping criterion, in general a maximal number of iterations i_max is also given:

3D-Var algorithm

- Initialisation : $\boldsymbol{x}_0 = \boldsymbol{x}^b$, $i = 0$
- While $\|\nabla J\| > \varepsilon$ or $i \leq i_{\max}$, do :
  1. Compute J
  2. Compute ∇J
  3. Descent and update of $\boldsymbol{x}_0$
  4. $i = i + 1$

This type of analysis has been used operationally in meteorological weather services in the 1990's. In the 2000's it has been replaced in many centres by the 4D-Var, a generalisation of the 3D-Var that we shall discuss later.

7.4 Algorithms in practice - Non-stationary case

Let us now consider the case described in §7.1, with the model supposed to be perfect. Then, the cost function is

$$J(\boldsymbol{x}_0) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\sum_{k=1}^{K}\|\mathcal{H}_k(\mathcal{M}_{0\to k}(\boldsymbol{x}_0)) - \boldsymbol{y}_k\|^2_{\mathbf{R}_k^{-1}}$$

7.4.1 Linear non-stationary case

We first study the case where $\mathcal{H}_k$ and $\mathcal{M}$ are linear. Then :

$$\mathcal{M}_{0\to k}(\boldsymbol{x}_0) = \mathbf{M}_{0,k}\ \boldsymbol{x}_0 = \mathbf{M}_{k-1,k}\ldots\mathbf{M}_{1,2}\ \mathbf{M}_{0,1}\ \boldsymbol{x}_0$$

$$J(\boldsymbol{x}_0) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\sum_{k=1}^{K}\|\mathbf{H}_k\ \mathbf{M}_{k-1,k}\ldots\mathbf{M}_{1,2}\mathbf{M}_{0,1}\boldsymbol{x}_0 - \boldsymbol{y}_k\|^2_{\mathbf{R}_k^{-1}}$$

By using the following expressions in (7.7)

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_1\mathbf{M}_{0,1} \\ \vdots \\ \mathbf{H}_K\mathbf{M}_{0,K} \end{pmatrix} \quad \boldsymbol{y} = \begin{pmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_K \end{pmatrix} \quad \mathbf{R}^{-1} = \begin{pmatrix} \mathbf{R}_1^{-1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \ddots & & \vdots \\ \vdots & & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{R}_K^{-1} \end{pmatrix}$$

we can use the previous result of the stationary case, and the expression of the minimum of J is :

$$\boldsymbol{x}^a = \boldsymbol{x}^b + \left[\mathbf{P}_b^{-1} + \sum_{k=1}^{K}\mathbf{M}_{0,k}^T\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{M}_{0,k}\right]^{-1}\sum_{k=1}^{K}\mathbf{M}_{0,k}^T\mathbf{H}_k^T\mathbf{R}_k^{-1}(\boldsymbol{y}_k - \mathbf{H}_k\mathbf{M}_{0,k}\boldsymbol{x}^b)$$

7.4.2 Non linear non stationary case: 4D-Var algorithm

4D-Var is a generalisation of 3D-Var to time-depending problems, with observations spanned over a time-window. The model is supposed to be perfect so that the cost function is written as a function of the initial state $\boldsymbol{x}_0$. The cost function is again (7.1).

$$J(\boldsymbol{x}_0) = \frac{1}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_b\|^2_{\mathbf{P}_b^{-1}} + \frac{1}{2}\sum_{k=1}^{K} \|\mathcal{H}_k(\mathcal{M}_{0\to k}(\boldsymbol{x}_0)) - \boldsymbol{y}_k\|^2_{\mathbf{R}_k^{-1}}$$

The gradient of J is given by:

$$\nabla J(\boldsymbol{x}_0) = \mathbf{P}_b^{-1}(\boldsymbol{x}_0 - \boldsymbol{x}^b) + \sum_{k=1}^{K} \mathbf{M}^T_{0,1[\boldsymbol{x}_0]} \dots \mathbf{M}^T_{k-1,k[\boldsymbol{x}_{k-1}]}\ \mathbf{H}^T_{k[\boldsymbol{x}_k]}\mathbf{R}_k^{-1}(\mathcal{H}_k(\mathcal{M}_{0\to k}(\boldsymbol{x}_0) - \boldsymbol{y}_k)$$

We will denote $\boldsymbol{d}_k$ the *innovation vector*: $\boldsymbol{d}_k = \boldsymbol{y}_k - \mathcal{H}_k(\mathcal{M}_{0\to k}(\boldsymbol{x}_0))$

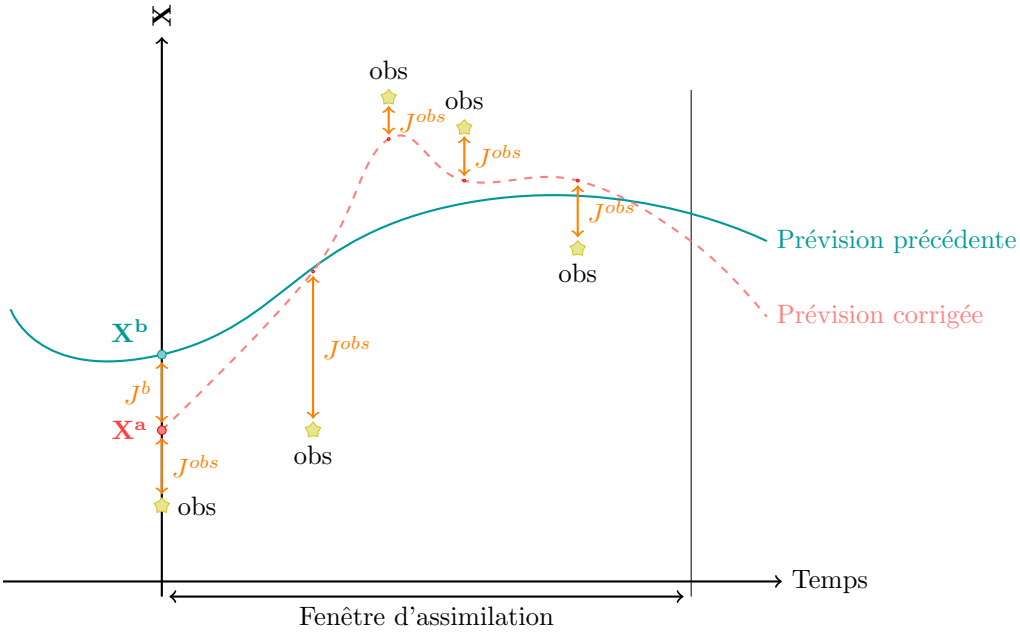


Figure 7.2: 4D-var

4D-Var algorithm

- Initialisation: $\boldsymbol{x}_0 = \boldsymbol{x}^b \;\; i = 0$
- Do while $\|\nabla J\| > \varepsilon$ or $i \leq i_{\max}$:
  1. Compute J and $\boldsymbol{d}_k$ thanks to the model and the observation operator :
  $$\boldsymbol{d}_k = \boldsymbol{y}_k - \mathcal{H}_k(\mathcal{M}_{0\to k}(\boldsymbol{x}_0))$$
  $$J(\boldsymbol{x}_0) = (\boldsymbol{x}_0 - \boldsymbol{x}^b)^T \mathbf{P}_b^{-1}(\boldsymbol{x}_0 - \boldsymbol{x}^b) + \sum_{k=1}^{K} \boldsymbol{d}_k^T R_k^{-1} \boldsymbol{d}_k$$
  2. Compute ∇J thanks to the adjoint models $\mathbf{M}^T$ and the adjoint observation operators $\mathbf{H}^T$ in reverse mode:
  $$\begin{aligned} \nabla J(\boldsymbol{x}_0) \;\; = \;\; & \mathbf{P}_b^{-1}(\boldsymbol{x}_0 - \boldsymbol{x}^b) \\ & - \textstyle\sum_{k=1}^{K} \mathbf{M}^T_{0,1[\boldsymbol{x}_0]} \dots \mathbf{M}^T_{k-1,k[\boldsymbol{x}_{k-1}]} \;\; \mathbf{H}^T_{k[\boldsymbol{x}_k]} \mathbf{R}_k^{-1} \boldsymbol{d}_k \end{aligned}$$
  3. Update $\boldsymbol{x}_0$ (descent step)
  4. $i = i + 1$

7.5 Exercices

1. prove equation (7.8)
2. prove equation (7.9)

7.6 Solutions

1. prove equation (7.8)
   We use the facts that :
   $$\mathbf{A}^T\mathbf{N}\mathbf{A} = \mathbf{P}_b^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}$$
   and
   $$\mathbf{A}^T\mathbf{N}\mathbf{b} = \mathbf{P}_b^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\boldsymbol{y} = (\mathbf{P}_b^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} - \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})\boldsymbol{x}_b + \mathbf{H}^T\mathbf{R}^{-1}\boldsymbol{y}$$
2. prove equation (7.9) Use the solutions of last week