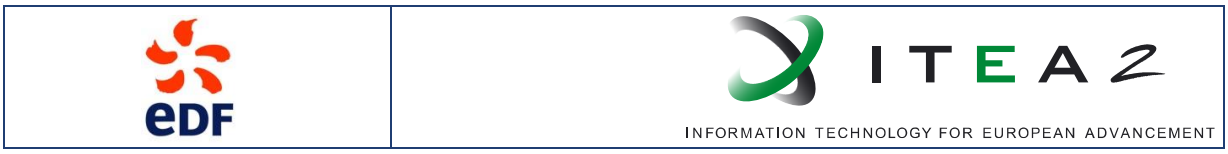

D3.1.1 – Method to extend models for system design to models for system operation

Part IIb: State estimation techniques for system diagnosis – State of the art

WP3.1 – Continuous systems
WP3 – State estimation and system monitoring

MODRIO (11004)

Version 1.0

Date 04/05/2016

Authors

Julien Lagarde	EDF
Audrey Jardin	EDF

Executive summary

The MODRIO overall objective is to extend Modeling and Simulation (M&S) tools from system design to system operation in order to cover all stages of the system lifecycle.

Extending models from system design to system operation implies the following:

1. Ability to replace design hypothesis by real system data to initialize the models from the best estimate of the system state
2. Ability to simulate the models through different operating modes

The objective of WP3.1 is to tackle the first point by working on state estimation techniques. The second point is taken care of in WP4 with the simulation of systems with multiple operating modes.

There are two main application areas that are dealt with in the scope of MODRIO regarding state estimation:

- non-linear model predictive control (NMPC)
- system diagnosis (assessing and predicting systems performance degradation in order to plan more efficiently maintenance operations).

The mathematical and algorithmic background are essentially the same for both applications, although the vocabulary may differ in some aspects. The first application uses vocabulary from the systems control community, whereas the second application uses vocabulary from the physics community, but may also use vocabulary from the systems control community as well.
The main difference between the two applications lies in the fact that the assessment of uncertainties is sought for diagnosis and not for systems control. On the other hand, real time capabilities are essential for systems control, and not for system diagnosis (where extra computing time may be used to compute the uncertainties).

For that reason, deliverable D3.1.1 is composed of two separate parts:

- Part I for "systems control" purpose (DLR contribution);
- Part II for "system diagnosis" applications (EDF contribution).

For system diagnosis, state estimation techniques are referred as data estimation techniques, which were originally developed for NWP (numerical weather prediction) since the early 1950's, which are now used in other fields such as NMPC (maybe under different terms). Data Assimilation (DA) refers to a set of mathematical algorithms such as Kalman filters, 3D-VAR, 4D-VAR, particle filters, Moving Horizon Estimator (MHE), etc. for state estimation of continuous physical systems.

The objective of D3.1.1 Part II is adapt these DA techniques to the 0D/1D modelling of physical systems such as power plants.

This document (Part IIb) is the corresponding state of the art.
Part IIa is included in D8.1.1 and proposes some tool solutions to apply in practice such DA techniques to a Modelica model.

Summary

1. Objective

This document briefly presents the main Data Assimilation (DA) methods that can be found in the literature. Its aim is not to give the theoretical basis of these methods but to analyse the assets and drawbacks of each one to help future user to choose the most appropriate algorithm according to his case study. Special care is given to precise which inputs are necessary to run each method. A few available data assimilation tools are also listed at the end of this document to give an idea on how such algorithms can be implemented and if existing codes can be reused.

1.1. Definition, notations and problem

The following definition is a translation from the definition which is given in (Bocquet, 2004-2009):
Data assimilation contains all statistical techniques that allow the improvement of the knowledge of a physical system, combining both measurements and a priori knowledge of the system.
Several notations can be found in the literature. It has been decided to use in this report the notations that can be found in (Bocquet, 2004-2009). They are presented in Table 1.

Notations	
$X \in \mathcal{M}_{n,1}(\mathbb{R})$	State vector : contains the variables of interest that should be estimated
$X_b \in \mathcal{M}_{n,1}(\mathbb{R})$	A priori estimate of the state vector / Background vector. Determined by design data, knowledge expertise or previous simulations
$P_b \in \mathcal{M}_{n,n}(\mathbb{R})$	Covariance matrix of the background error. Determined by knowledge expertise or previous simulations
$X_a \in \mathcal{M}_{n,1}(\mathbb{R})$	Analysis vector: new value of the state vector once the data assimilation is completed
$P_a \in \mathcal{M}_{n,n}(\mathbb{R})$	Analysis error covariance matrix. This matrix is computed.
$Y \in \mathcal{M}_{p,1}(\mathbb{R})$	Observation vector (contains sensor values)
$R \in \mathcal{M}_{p,p}(\mathbb{R})$	Observation error covariance matrix: contains the uncertainties of the measurements and the correlations between the measurement errors. Determined by knowledge expertise and by sensor operating instructions manual.
$M: \mathbb{R}^n \rightarrow \mathbb{R}^n$	Evolution operator: describes the time evolution of the state vector. This operator is often a complex code.
$Q \in \mathcal{M}_{n,n}(\mathbb{R})$	Evolution error covariance matrix
$H: \mathbb{R}^n \rightarrow \mathbb{R}^p$	Operator which allows going from the space of states to the space of observations. This operator is often a complex code.

Table 1: Data assimilation - Main vectors and operators – notations

The objective of data assimilation is to evaluate both X_a and P_a and thereby to get an information on the system state and on the uncertainty of the system state estimation. Data assimilation proceeds as illustrated on Figure 1.

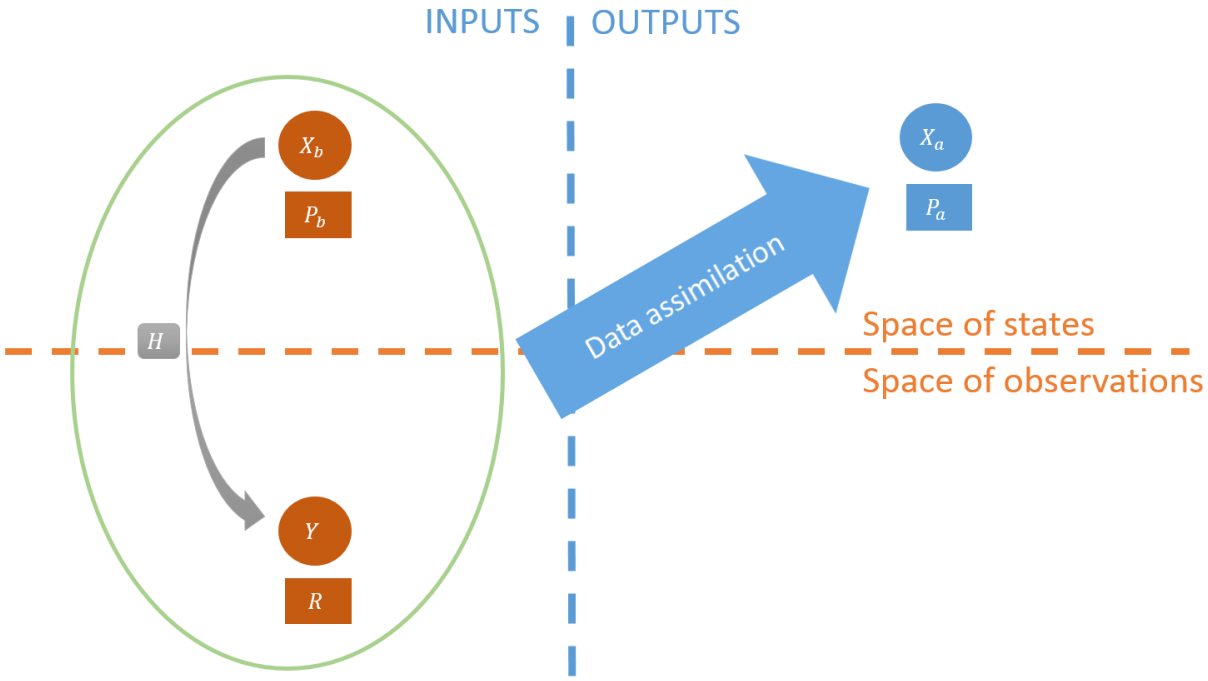

Figure 1: Data assimilation - Principle

There are several possibilities to formalize the data assimilation problem:

- ***Variational formulation***. The value of X_a has to depend on both the a priori value of the state vector X_b and the measurements Y. H is the operator which allows going from the space of states to the space of observations. The more the measurements (resp. the a priori value of the state vector) are precise, the more the influence of their value on X_a has to be high. The cost function $J(X) = \frac{1}{2}\|Y - H(X)\|_{R^{-1}} + \frac{1}{2}\|X - X_b\|_{P_b{}^{-1}}$ contains all these constraints. Its minimization gives an interesting value for the analysis vector (for the sake of simplicity, all the measurements are supposed to be done at the same time) → $\min_X(J(X))$**?**
- ***Bayesian formulation.*** One has at disposal measurements, whose uncertainty can be described by a known probability law, and an a priori idea of the value of the state vector, whose uncertainty can also be described by a known probability law. The physical knowledge of the system let one know the influence that a variation of the state vector would have on the variables measured by sensors. From the probability distributions of the a priori vector and the observations, the Bayes formula makes it possible to evaluate the probability distribution of the analysis vector → $P_{X|Y}$**?**

These two formulations of the data assimilation problem are not equivalent. They are generally equivalent if:

- The observation and evolution operators are linear;
- The probability densities of the variables contained by the state vector are all Gaussian.

1.2. Hypotheses

All the data assimilation algorithms suppose that the measurement errors are unbiased. In order to apply a data assimilation algorithm, one has to have at disposal:

- An a priori value of the system state and to be able to evaluate the uncertainty of this a priori estimation;
- Measurements coming from sensors and their uncertainties;

- An operator that link the value of the state of the system with the values measured by the sensors (or knowledge of the impact of a state variation on the variables estimated by sensors). This operator can be a numeric code.

Depending on the data assimilation algorithm, some hypotheses (linearity for instance) and some other material can be required (the adjoint of the observation operator, the evolution operator, its adjoint, its derivative operator…). The hypotheses made by each of the data assimilation algorithms presented in this document will be specified in the section where they are described.

2. Data assimilation most usual (or well-known) methods

2.1. BLUE: Best Linear Unbiased Estimator

BLUE stands for Best Linear Unbiased Estimator which can be viewed asthe easiest of data assimilation methods. This data assimilation method is dedicated to evaluating a state vector when the model that link the measurements and the state vector is linear, and when there is no temporal dimension. If the variables of the state vector are measured by sensors, applying the BLUE algorithm is nothing more than averaging the measurement values and the a priori values with weights depending on their uncertainties.

Hypotheses:

- Linear observation operator
- Unbiased a priori error
- Invertible a priori error covariance matrix
- Measurement errors and a priori errors not correlated

The state analysis vector can be written (thanks to the linearity of the observation operator):

$$X^a = LX^b + KY$$

And in order to get an estimation with an unbiased error:

$$X^a = X^b + K(Y - HX^b)$$

The vector $Y - HX^b$ is called the innovation vector.
The analysis error covariance matrix is given by:

$$P^a = (I - KH)P^b(I - KH)^T + KRK^T$$

The minimization of $Tr(P^a)$) gives the BLUE estimation:

$$P^a = (I - K^*H)P^b$$

With:

$$K^* = P^b H^T (R + HP^b H^T)^{-1}$$

More details are available in (Bocquet, 2004-2009).

2.2. 3D-Var: Three-dimensional variational method

2.2.1. Linear observation operator

This algorithm is equivalent to the BLUE algorithm if the observation operator is linear. It corresponds to the minimum value of the following cost function:

$$J(X) = \frac{1}{2}\|Y - H(X)\|_{R^{-1}} + \frac{1}{2}\|X - X_b\|_{P_b{}^{-1}}$$

The minimization gives:

$$X^a = X^b + K^*(Y - HX^b)$$

With:

$$K^* = \left(P^b{}^{-1} + H^T R^{-1} H\right)^{-1} H^T R^{-1}$$

This algorithm can be extended to problems with non-linear observation operators. This approach is also algorithmically more efficient when the number of iterations is not too large: contrarily to the BLUE algorithm, it minimizes a cost function. At each iteration, the invert matrices of R and P_b are multiplied by a matrix, which is less expensive than inverting: $R + HP^b H^T$.
The 3D-Var has been used in meteorology during many years (before being replaced by the 4D-Var).

2.2.2. Non-linear observation operator

The same method can be used with a non-linear observation operator. The cost function is replaced by:

$$J(X) = \frac{1}{2}\|Y - H[X]\|_{R^{-1}} + \frac{1}{2}\|X - X_b\|_{P_b{}^{-1}}$$

Where H is the first order term of the Taylor development of H.

2.3. Kalman filter

Contrarily to the two previous methods, the Kalman filter is a sequential interpolation method. Such a system can be described by the following system of stochastic equations:

$$\begin{cases} x_{k+1} = M_{k+1}[x_k] + w_k \\ y_k = H_k[x_k] + v_k \end{cases}$$

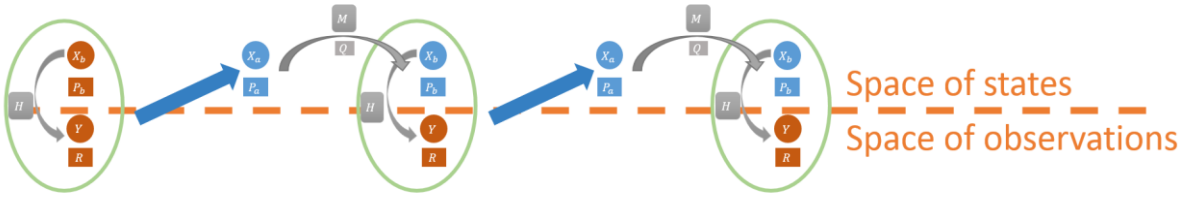

Figure 2: Estimation method of the Kalman filter

Hypothesis: all the operators are supposed to be linear operators.
If the estimator associated to the analysis vector is the minimum of the analysis error, the Kalman filter is a succession of BLUE analyses.

2.4. Extended Kalman filter

This algorithm is used if the observation operator or if the evolution operator is not linear. The operators are linearized. The Kalman gain is:

$$K^* = P^b H^T (R + H P^b H^T)^{-1}$$

The optimal estimator is:

$$X^a = X^b + K^*(Y - H[X^b])$$

The first term of the Taylor development of H is in the Kalman gain but not in the innovation vector.

2.5. 4D-Var: Four-dimensional variational data assimilation

With some assumptions, a variational formalism can be associated to the Kalman filter. 4D-Var is completely equivalent to the Kalman filter only if the observation and evolution operators are linear operators.

2.5.1. Deterministic evolution

The temporal generalization of the 3D-Var cost function is:

$$J(X) = \frac{1}{2}\sum \|Y_k - H_k(X_k)\|_{R^{-1}} + \frac{1}{2}\left\|X_0 - X_0^b\right\|_{P_0^{b-1}}$$

With:

$$\forall k, X_{k+1} = M_{k+1}(X_k)$$

The adjoint operator is necessary to apply this method.
More information is available in (Bocquet, 2004-2009).

If the observation and/or the evolution operator are not linear, they have to be (like in a 3D-Var

approach when the operators are not linear) linearized. If the operators are given by a code, the code has to be differentiated.

2.6. Bayesian approach of the data assimilation problem

2.6.1. Bayes formula

The Bayes Formula:

$$p_{A|B}p_B = p_{B|A}p_A$$

allows to estimate the probability of different causes to be responsible of an observed phenomenon.

Let us consider the following example:

- Two machines (M1, M2) produce exactly the same number of screws. At the end of the day, all the screws are put together in a container. If a screw is randomly selected in the container, there is a probability of 50% that it has been created by M1 and 50% that it has been created by M2.
- A screw made by M1 has a probability of 3% to be declared bad by a test on the produced screw, while a screw made by M2 has a probability of 10% to be declared bad.
- I choose randomly a screw in the container. It is a "bad" screw → what is the probability that this screw comes from M1?

$$\begin{cases} P(M1) = 0.5 \\ P(\overline{M1}) = 0.5 \\ P(B|M1) = 0.03 \\ P(B|\overline{M1}) = 0.1 \end{cases} \quad \rightarrow \quad P(M1|B) \ ?$$

Bayes formula gives:

$$P(M1|B) = \frac{P(B|M1)P(M1)}{P(B)} = \frac{P(B|M1)P(M1)}{P(B|M1)P(M1) + P(B|\overline{M1})P(\overline{M1})} = \frac{0.03 \times 0.5}{0.03 \times 0.5 + 0.1 \times 0.5} = 0.23$$

The probability that a screw is from M1 if it is known that the screw is a bad one is 23%: Bayes theorem makes it possible to invert cause and consequence!

The Bayesian approach is illustrated by the following figure:

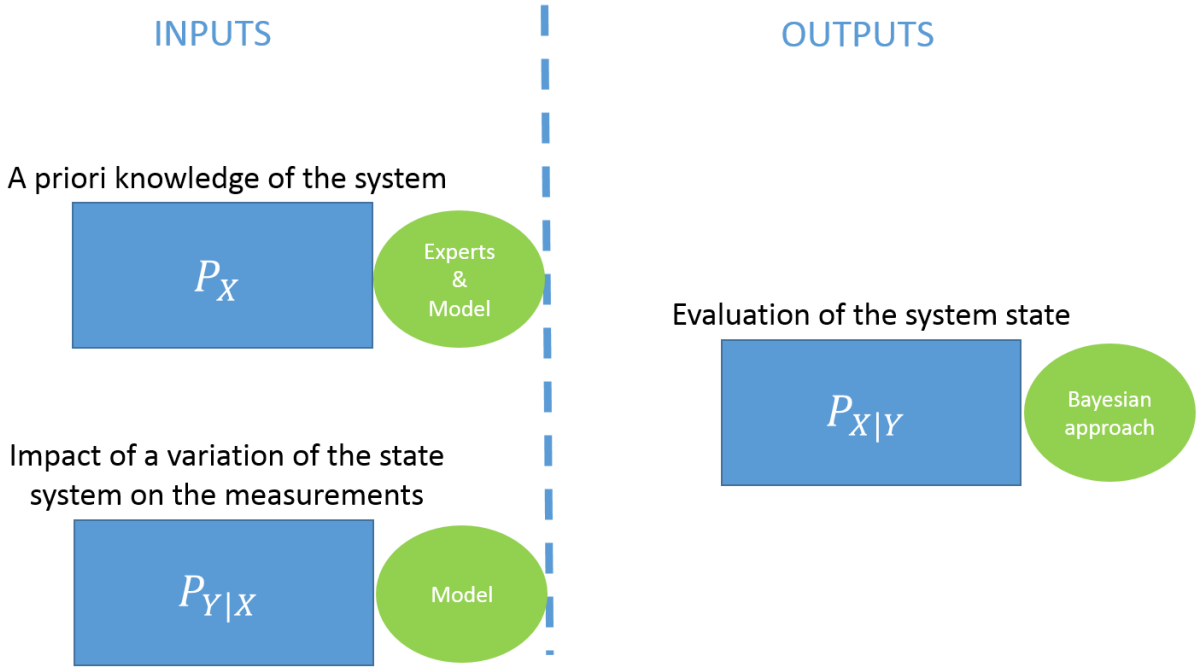

Figure 3: Bayesian estimation approach

The Bayesian approach of the data assimilation problem is more general than the approaches presented in this document so far. The other algorithms, based on the variational formulation of the data assimilation problem, do indeed rely on the propagation of the two first moments of a probability density function. On the contrary, the Bayesian formulation of the data assimilation problem uses directly the probability densities, which contains of course more information and by far than the mean value and the standard deviation of a probability density.

2.6.2. Particular filter

It is possible to build solutions that converge towards the solution of the Bayesian data assimilation problem. They are based on Monte-Carlo methods. The particular filter for instance replaces the probability density by a sample on which the Bayes formula is applied. The statistical properties of the sample are computed and give the a posteriori value of the state vector.
Figure 4 presents the best-known Monte-Carlo algorithm that solves the Bayesian problem: the bootstrap particular filter. The orange items of the graph represent the inputs: the a priori probability density of the state vector, the observations, and the impact of a variation of the system state on a measurement. A sample of particles is created from the a priori probability density of the system state. All the particles are given the same weight W_i. The probability density of the sample is given by:

$$P(X) = \sum W_i \delta(X - X_i)$$

These particles are propagated with the evolution operator M(which is supposed deterministic for the sake of simplicity). Using the measurements and the knowledge of the impact of a system state variation on the measurements, the weight of the particles is updated with the Bayes formula:

$$W_i^{t+1} = W_i^t P(Y_j^{t+1} | X_i^{t+1})$$

The weight of some particles can grow a lot, which make the sample unable to approach the real probability density function. It is therefore necessary to resample. Resampling methods are not detailed here.
This data assimilation method is very efficient on systems in which the number of variables to estimate is low.

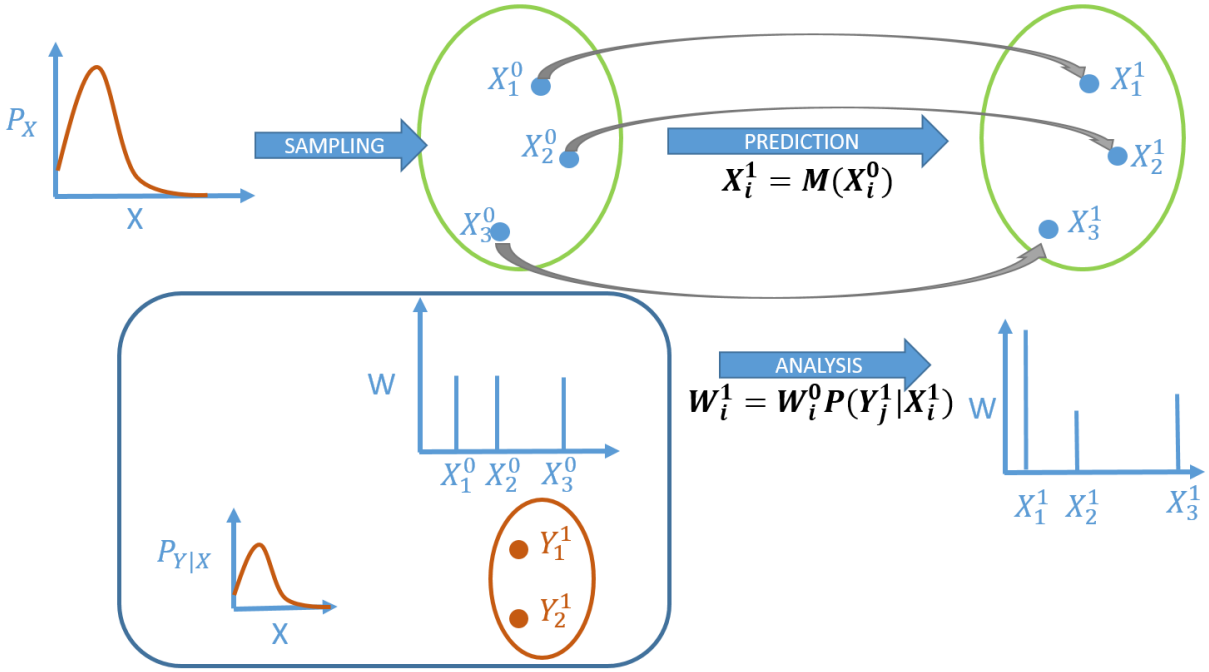

Figure 4: Bootstrap particular filter

2.6.3. Ensemble Kalman filter

The Ensemble Kalman filter is not a particular filter. It does only propagate two moments of a

probability density distribution, but it does not propagate the uncertainties through error covariance matrices: it does it through a sample of particles. It is a Gaussian filter and does not solve the Bayesian equations, even with an infinite number of particles. The analysis step is the same as this of a normal Kalman filter, except that a noise is added to the observations (which prevent the sample from becoming “poor”).

2.6.4. Unscented Kalman filter

The Unscented Kalman Filter is a refinement of the Ensemble Kalman Filter. It gives better results (right at the second order, while the Ensemble Kalman filter gives results that are right only at the first order). It is based on a Choleski decomposition which can be expensive in terms of computational costs.

3. Assets and drawbacks of each method

Method	Inputs	Hypotheses	Comments
All methods	X_b,Y	Unbiased measurement errors Background and measurement errors decorrelated	
BLUE (Best Linear Unbiased Estimator)	P_b,R,H	Invertibility of P_b and R Linear observation operator H	Easiness Optimal estimator
3D-Var (Variational formulation of statistical interpolation)	P_b,R <u>Linear observation operator:</u> H <u>Non-linear observation operator:</u> H and its derivative at X_b	Invertibility of P_b and R The a priori error and the measurement error abide by a Gaussian law	Quick if there are few iterations to converge Can be used with a non-linear operator H The cost function can be minimized in the observation space (which is usually smaller) This method has no temporal dimension Optimal estimator if the observation operator is linear
Kalman filter	P_b, R, H, M, Q	Invertibility of P_b, R and Q Linear observation and evolution operators H and M The a priori error and the	Temporal version of the BLUE algorithm Easiness (at each step, the BLUE algorithm is used)

		measurement error abide by a Gaussian law	Optimal estimator
Extended Kalman filter (non-linear version of Kalman filter)	P_b, R, H, M, Q The derivatives of H and M if they are not linear	Invertibility of P_b, R and Q The a priori error and the measurement error abide by a Gaussian law	The derivative operator of H is used to compute the Kalman gain but not the innovative vector $y_k - H[x_k^f]$ Disadvantage : not an optimal estimator (because of the linearization) Standard in navigation systems and GPS Temporal method
4D-Var (Variational version of the Kalman filter)	P_b, R, H, M, Q Adjoint operators of M and H Derivatives of M and H	Invertibility of P_b, R and Q The a priori error and the measurement error abide by a Gaussian law	Acausal (can use both information from the past and from the future) while the Kalman filter uses only information from the past Equivalent to Kalman filter if H and M are linear There is software to find the adjoint operator (TAPENADE, TAF) and the differential form of a numeric code. Correspond to the maximum a posteriori estimator for the fully Bayesian posterior distribution if the operators are linear
Particular filter	P_X, $P_{Y\|X}$, Y, M		Converge towards the solution of the Bayesian data assimilation problem The number of required iterations to converge can be very high

			(depending on the dimension of the state space)
Ensemble Kalman filter (Kalman filter for large problems)	P_b, R, H, M, Q	The a priori error and the measurement error abide by a Gaussian law	Monte-Carlo implementation of the Bayesian problem More efficient than the particle filter when it is applicable Temporal method Does not converge towards the solution of the Bayesian formulation of the DA problem Gaussian filter.
Unscented Kalman filter	P_b, R, H, M, Q		Gives a better estimation than the Ensemble Kalman Filter Is based on a Cholesky decomposition (which can be expensive in terms of computational costs)

4. Existing implementation and tools

Several data assimilation tools have been developed and can be found in the literature. Some are listed here:

Tool	**Description/Comments**	**Number of downloads in the last month**	**Source**
Python data assimilation package: pyda	The setup.py file does not seem to be done as it is supposed to be. Create a .egg but no .egg-info. It is not possible to test the examples. Too few users. I'll not use this library.	55	(Pyda 1.0, 2014)
OpenDA data assimilation toolbox	Lots of data assimilation methods are implemented Language interfaces: C/C++, Java,	161/week	(OpenDA data-assimilation toolbox, s.d.)

	Fortran 77/90		
Kalman library	Python library Too few users. I'll not use this library.	153	(Luong, 2015)
Pykalman library	Python library	4125	(Duckworth, 2013)a
filterpy	Python library that provides Kalman filtering and various related optimal and non-optimal filtering software written in Python. It contains Kalman filters, Extended Kalman filters, Unscented Kalman filters, Kalman smoothers, Least Squares filters, fading memory filters, g-h filters, discrete Bayes, and more. This library has been developed in conjunction with the book *Kalman Filters and Random Signals in Python* Recent software: library uploaded on pypi on 2016/01/24.	3334	(Labbe, 2016)
PyMC library	Versions on both Python 2 and Python 3 PyMC is a python module that implements Bayesian statistical models and fitting algorithms, including Markov chain Monte Carlo. It includes methods for summarizing output, plotting, goodness-of-fit and convergence diagnostics. It should work but it uses a g77 compiler and g77 compiler has been declared obsolete. I will try another way round. Actively developed on Git (some posts have a few days).	4148	(Patil, Huard, & Fonnesbeck, 2010) (Fonnesbeck, Patil, & Huard, s.d.)
PyBayes 0.3	Too few users for being really considered as a reliable and stable implementation	85	(Laitl, 2011)
Emcee	Pure-Python implementation of Goodman & Weare's Affine Invariant Markov chain Monte Carlo (MCMC) Ensemble sampler. Has	No information	(Foreman-Mackey, 2015)

	been used in astrophysical literature. Mostly developed by Dan Foreman-Mackey (NYU) between 2011 and 2015. No further development these last months.		
PDAF: the parallel Data Assimilation Framework	Implemented in Fortran 90. Standard interface supports models that are written in other languages like C or C++		(PDAF, s.d.)
ADAO module	Developed by EDF	No information	

Table 2: Data assimilation tools

In accordance to the first experimentations made in deliverable D8.1.1 an emphasis has been put on Python libraries. Among them, the most promising libraries are:

- PyMC: lots of users, still in development, Bayesian methods implemented.
- Filterpy: still in development, lots of users, many Kalman filters implemented.
- ADAO: still in development, many different algorithms implemented, in-house developers with who it should be easier to interact with and tightly collaborate.

5. References

Bocquet, M. (2004-2009). *Introduction aux principes et méthodes de l'assimilation de données en géophysique.*

Duckworth, D. (2013). *pykalman 0.9.5*. https://pypi.python.org/pypi/pykalman

EDF, E. P. (2016). *OpenTURNS*. http://www.openturns.org/?subwindow=home

Fonnesbeck, C., Patil, A., & Huard, D. (s.d.). *pymc 2.3.6*. https://pypi.python.org/pypi/pymc

Foreman-Mackey, D. (2015). *EMCEE*. http://dan.iel.fm/emcee/current/

Labbe, R. (2016). *filterpy 0.1.2*. https://pypi.python.org/pypi/filterpy

Laitl, M. (2011). *PyBayes*. https://pypi.python.org/pypi/PyBayes

Law, K., Stuart, A., & Zygalakis, K. (2015). *Data assimilation, A Mathematical Introduction.*

Luong, V. (2015). *Kalman 0.1.3*. https://pypi.python.org/pypi/Kalman

OpenDA data-assimilation toolbox. (s.d.). http://www.openda.org

Patil, A., Huard, D., & Fonnesbeck, C. (2010). *PyMC: Bayesian Stochastic Modelling in Python*. http://www.map.ox.ac.uk/media/PDF/Patil_et_al_2010.pdf

PDAF. (s.d.). the Parallel Data Assimilation Framework: http://pdaf.awi.de/trac/wiki

Pyda 1.0. (2014). PyPI - the Python Package Index: https://pypi.python.org/pypi/pyda