Centro Euro-Mediterraneo sui Cambiamenti Climatici

Research Papers
Issue RP0165
December 2012

SCO - Scientific Computing and Operations Division

Numerical Methods for Data Assimilation: Kalman Filter

By **Luisa D'Amore**
University of Naples Federico II
luisa.damore@unina.it

Rossella Arcucci
Centro Euro-Mediterraneo sui Cambiamenti Climatici (CMCC)
rossella.arcucci@cmcc.it

Almerico Murli
Centro Euro-Mediterraneo sui Cambiamenti Climatici (CMCC)
almerico.murli@cmcc.it

SUMMARY The Kalman filter (KF) dates back to 1960, when R. E. Kalman [4] provided a recursive algorithm to compute the solution of a (linear) data filtering and prediction problem, proving to be much more efficient than the N. Wiener's approach, introduced in 1949 in [5].
Data filtering is a *simple* example of Data Assimilation problem which can be regarded as a least squares approximation problem and, more precisely, as an inverse ill-posed problem. In this paper we review and discuss KF in the context of numerical regularization methods aimed to solve ill-posed inverse problems such those arising in Data Assimilation applications.

INTRODUCTION AND HISTORY

In the past 20 years Data Assimilation (DA), used in principle only in atmospheric models, has become a main component in the development and validation of oceanographic models and, more generally, in the validation of the mathematical models used in meteorology, climatology, geophysics, geology and hydrology (often these models are referred to with the term *predictive* to underline that these are dynamical systems).
The aim of the "data assimilation scientific community" is to "*accost*" the data acquired experimentally (in vivo) to those obtained numerically (in vitro) in order to improve the understanding of the surrounding ecosystem.

Predictive models are described by using evolutionary differential equations. Predictions are obtained by running "computational simulations", hence a small perturbation on data may propagate on the solution. Since this solution is used, in turn, as initial condition of a next time prediction, propagation can lead to completely unreliable solutions, even after a few time steps. This catastrophic error amplification was already known in 1960 to Edward Lorenz (1917-2008, founder of chaos theory). Lorenz convinced himself that the models used to describe climate changes provide solutions unpredictable: variations of the initial parameters on the third or fourth (significant) digit produced enormous disruption in the solution. The so strong dependence on the initial parameters was called butterfly effect:: "*Does the flap of a butterfly's wings in Brazil set off a tornado in Texas*?" was the title of a conference at the American Association for the Advancement of Sciences held by E. Lorenz in 1979.

In this context DA corrects, periodically, the initial value of a predictive model using the information provided by the experimental measurements, or by observations of the state system acquired in the same time of the initial condition.

Assimilate the data (from the Latin "assimilate" = make similar) just means, in this context, to make as similar as possible the observed data to those provided by the simulation models. Therefore, we may introduce the **DA mathematical problem as an Approximation problem** and, in particular, as a Best Approximation problem. Let us start by giving the following definition of DA, even if in a simplified version [1]:

Definition 0.1 *(Data Assimilation)*
Given a dynamical system, an estimate of the system state at a fixed time, and an experimental measure of the system state, to compute the Best Approximation of the system state at each time.

Let x be the vector representing the system state, y the vector of experimental measurements of the system state, using the Euclidean norm to measure the distance between x and y (for details, see the Appendix), the DA problem can be formulated hereafter as:

Definition 0.2 *(Data Assimilation as Best Approximation)*
Given two vectors x and y, to compute the vector $\widetilde{x}$ obtaining the minimum distance from x and y.

In the context of DA, this solution is named "analysis".

Here after we will refine this definition of DA, and we will give a precise characterization of the "data filtering", on which R.E. Kalman began to work in 1960, in the context of **inverse and ill posed problems**.

THE KALMAN PROBLEM: AN INVERSE ILL POSED PROBLEM

The term "*filtering*" originates from the operation separating the information from the noise/error:

$$filter \rightleftharpoons operator$$

$$which\ separates\ data\ from\ noise$$

Let us consider the Kalman problem [4]:

Definition 0.3 **(Kalman)**
Let us consider the two functions:

$$x_k(t) : t \in [0,T] \quad \longrightarrow x_k(t), \quad k = 1,2.$$

We assume that $x_1(t)$ describes a phenomenon that we are observing while the other, $x_2(t)$, is the additive noise. Suppose that their sum:

$$y(t_i) = x_1(t_i) + x_2(t_i), \quad t_i \in [0,T]$$

is known in a finite number, let's say m, of values $t_i, i = 1,\dots,m$. Set a value of t in the same interval of the observations, which we denote by $\widetilde{t} \in [0,T]$. Kalman poses the following:

Problem 0.1 *is it possible to calculate an estimate of $x_1(\widetilde{t})$ from this information? if so, how can it be done?*

In case of $m = 2$, let

$$y_i = y(t_i) \quad i = 1,2$$

and

$$\epsilon_i = x_2(t_i) \quad i = 1,2$$

the Kalman problem can be rewritten in this way:

Definition 0.4 **(Kalman)**
Given the points

$$(t_i, y_i), \quad i = 1,2 \quad t_i \in [0,T]$$

where

$$y_i = x_1(t_i) + \epsilon_i,$$

to compute $x_1(\widetilde{t})$, $\widetilde{t} \in [0,T]$.

Depending on the position of $\widetilde{t}$ with respect to t_1 and t_2 in [4] this problem was characterized as follows:

Definition 0.5 If

- $t_1 < \widetilde{t} < t_2$: data smoothing (fitting of data)
- $\widetilde{t} = t_i$: data filtering
- $\widetilde{t} > \{t_1, t_2\}$: data prediction (data mining)

and, in general, it was called *Data Estimation*, or *Data Assimilation.*

The Kalman problem is an identification problem [6], being in particular an approximation problem:

Definition 0.6 *(Approximation problem)*
Given the points (t_i, y_i) where $y_i = y(t_i)$, $i = 1,\dots,m$ we aim to determine a function, denoted by $x(t)$, as a fitting of the points. In particular, since the values y_i are affected by error (not negligible) we search for a function $x(t)$ which is "slightly deviated" from the values y_i.

The function x must be such that at t_i its distance from the assigned values (ie $\|\epsilon_i\| = \|x(t_i) - y_i\|$) is small or minimal, i.e.:

$$x = argmin\|x(t_i) - y_i\|$$

This is called a *denoising* problem: to reduce the noise (ϵ_i) given ($x_1(t_i)$).
Hence, the Kalman problem, given in Problem 0.1, is denoising problem: the function x_1 is the model approximating the points $(t_i, y_i), i = 1,2$. It is obtained by requiring that the distance between $x_1(t_i)$ and y_i is minimum, i.e.:

$$x_1 = argmin\|(\epsilon_1, \epsilon_2)\| =$$

$$= \min \|(y_1 - x_1(t_1), y_2 - x_1(t_2)\|.$$

REMARK: The data approximation - in the sense of least squares - goes back to *C. F. Gauss* which, although published later (in 1805), he had already used in 1795 at age 18, in his study of the orbits of the planets. Gauss states that[3]:

> "*[...] measurements are affected by errors and so are all obtained from these computations, therefore, the only way to get information about the problem at hand is to compute an approximation of the nearest and most practicable solution possible. This can be done by using a suitable combination of the experimental measurements, which must be in number than those of the unknown parameters, and starting from an approximate knowledge of the orbit (to be calculated), which will be corrected in order to describe as accurately as possible the experimental observations.*"

Gauss focused on the main ingredients needed for the computation of the solution of an approximation problem. Indeed, he refers to a solution "as close as possible" and to a calculation "affordable", through:

1. the use of experimental measurements in a number higher than that of the unknown parameters;
2. the estimation of the model linking the quantities and the known unknowns;
3. the calculation of the minimum distance between the known values and those obtained by solving the model.

It is possible to identify here some features common to the Kalman problem, as for example, the time dependence of the experimental measurements.

Careful analysis reveals that the operator mapping data (the measures y_i) to the unknown (the function $x_1(t)$) is known, being the sum of x_1 and x_2. In other words, we can say that the Kalman problem is an **inverse problem**[1] .

Definition 0.7 *(*Kalman *problem as inverse problem)*
Let S*:*

$$S: \quad y_1(t) \to x_1(t) + \epsilon(t)$$

the Problem *0.1 can be written in the equivalent form:*

ϵ is known as *error function*

Problem 0.2

$$P: \quad x_1(t) \to y_1(t) = S(x_1)$$

whence

$$x_1(t) = S^{-1}[y_1(t)]$$

So, the computation of x_1 *requires inverting the operator* S*.*

So, the computation of $x_1(t_i)$, only knowing $y_1(t_i)$, admits infinite solutions (one equation and two unknowns). This means that the operator S that relates y_i to $x_1(t_i)$ is not invertible, and the Kalman problem, as given in

[1] It is worth noting that the denoising problem can indeed be regarded as an inverse problem where the operator is the identity

Problem 0.2, is an **ill-posed inverse problem**.

REMARK: The characterization of ill-posed mathematical problems, dates back to the early years of the last century (J. Hadamard, 1902) and reflects the belief of the mathematicians of that time to be able to describe in a unique and complete way each physics problem. As a result, a problem was ill posed when, from the mathematical point of view, it presents anomalies and for this reason it could certainly not correspond to a physical event. Therefore, for some years, ill-posed problems were not taken into consideration by mathematicians. The first comprehensive treatment of ill-posed problems, it is due to A. N. Tikhonov, in 1965, which described the concept of solution for ill-posed problem and introduced the regularization methods [10]. A regularization method computes as (approximation of) the solution of an ill-posed problem, the best possible. This solution is obtained by solving a best approximation problem by adding to the minimization of the error, resulting from the problem, one or more constraints on the desired solution, arising from additional information. In other words, a regularization method replaces the problem with one another, well posed, whose solution, under certain assumptions, should be close to the ''ideal'' solution to the initial problem.

THE KALMAN FILTER: A NUMERICAL METHOD TO SOLVE THE DA INVERSE PROBLEM

Hereafter we will discuss KF as an inverse ill posed problem, through some case studies, each obtained as a refinement of the one posed by Kalman in 1960.

CASE STUDY 1:

Let $\widehat{x}$ be the solution of the normal equations arising from the least-squares problem treated by CF Gauss (see Appendix, (20)):

$$\widehat{x} = (A^T A)^{-1} A^T b$$

Suppose we have already solved the system (20) and that we have computed the solution $\widehat{x}$. The following problem provides a **first example of Data Assimilation** which we aim to solve appling the Kalman Filter:

Problem 0.3 *We consider the linear system (20) with the addition of a new equation:*

$$a^T x = \widetilde{b}$$

So, the system (20) becomes:

$$Mz = p \tag{1}$$

where

$$M = \begin{bmatrix} A \\ a^T \end{bmatrix} \in \Re^{(m+1)\times n} \tag{2}$$

and

$$p = \begin{bmatrix} b \\ \widetilde{b} \end{bmatrix} \in \Re^{(m+1)\times 1} \tag{3}$$

The system (10) is over-determined (because $m+1 > n$), then we aim to calculate the least square solution.

The least square solution of the system (10) is such that:

$$z = argmin_z \|Mz - p\|_2$$

and it is obtained by solving the normal equations:

$$M^T M z = M^T p \tag{4}$$

or else:

$$z = (M^T M)^{-1} M^T p$$

Observe that this formulation, although formally correct, leads to an inefficient computational approach because it does not use the vector $\widehat{x}$ which was already computed. We analyze in details the time complexity needed for solving system (4).

Let T_{normal} denote the **algorithm complexity** requested for solving system (4). It is due to the:

1. construction of normal equations (4):
   (a) calculation of $M^T M$: $T_{costr}(M^T M) = O((m+1) \times n^2)$ flop,
   (b) calculation of $M^T p$: $T_{costr}(M^T p) = O(n \times (m+1))$ flop
2. solution of system (4): $T^{sol} = O(n^3)$ flop

for an amount of:

$$T_{normal} = T_{costr} + T_{sol} =$$
$$= O(\underbrace{(m+1) \times n^2 + n \times (m+1)}_{passo1} + \underbrace{n^3}_{passo2})$$

Computing T_{normal} we observe that:

- (a) Since

$$M^T M = A^T A + aa^T$$

it is:

$$T_{costr}(M^T M) = O((m+1) \times n^2) =$$
$$= O(\underbrace{m \times n^2}_{A^T A} + \underbrace{n^2}_{aa^T}) \quad flop$$

	Standard
T_{costr}	$(m+1) \times n^2 + n \times (m+1)$
T_{sol}	n^3

Table 1: *Algorithm complexity of the construction and resolution of the normal equations (4).*

- (b) Since

$$M^T p = A^T b + a\widehat{b}$$

we have:

$$T_{costr}[M^T p] = O(n \times (m+1)) =$$
$$= O(\underbrace{n \times m}_{A^T b} + \underbrace{n}_{a\widehat{b}}) \quad flop$$

So, assuming that the normal equations $A^T Ax = A^T b$ have already been constructed and solved, it follows that:

$$T_{constr} = O((m+1) \times n^2 + n \times (m+1)) =$$
$$= O(\underbrace{m \times n^2}_{A^T A} + n^2 + \underbrace{n \times m}_{A^T b} + n)$$

whence

$$T_{constr} = O(n^2 + n) \quad flop$$

As shown in Table 2, we have still assumed that the solution of the normal equations has a time complexity of $O(n^3)$ flops. To evaluate the savings in solving the normal equations (4) it is necessary to analyze how to calculate $z = (M^T M)^{-1} M^T p$ using $\widehat{x} = (A^T A)^{-1} A^T b$.
To this end, we observe that system (4) derives from the problem:

$$P: \quad z = argmin_x \{\|Ax - b\|_2 + \|a^T x - \widehat{b}\|_2\} \tag{5}$$

	standard	updating
T_{costr}	$(m+1) \times n^2 + n \times (m+1)$	$n^2 + n$
T_{sol}	n^3	n^3

Table 2: *Algorithm complexity of forming the normal equations (4) in the standard formulation (standard) and avoiding to calculate A^TA and A^Tb (updating).*

in which the part relating to the matrix A is **separated** from that related to the new equation. In other words, we are looking for a solution that, in addition to be solution of (20), minimizes the residual of the new equation, too, as stated in the following:

Problem 0.4 *Let $\widehat{x} = (A^TA)^{-1}A^Tb$ be the least squares solution of the system $Ax = b$. Calculate the least squares solution of the system:*

$$Mz = p \tag{6}$$

with

$$M = \begin{bmatrix} A \\ a^T \end{bmatrix} \in \Re^{(m+1)\times n} \tag{7}$$

and

$$p = \begin{bmatrix} b \\ \widetilde{b} \end{bmatrix} \in \Re^{(m+1)\times 1} \tag{8}$$

means to solve the following constrained least squares problem:

$$P: \quad z = argmin_x\{\|Ax - b\|_2 + \|a^Tx - \widehat{b}\|_2\} \tag{9}$$

Problem P expresses the system (7) as a **constrained least-squares approximation** problem. The credit of R. E. Kalman was to have identified such relation and to have proposed a numerical method for the computation of z - solution of the system (7) - as a solution of the problem (9):

Propostion 0.1 (Kalman Filter) *Let*

$$Mz = p$$

a linear system, where

$$M = \begin{bmatrix} A \\ a^T \end{bmatrix} \in \Re^{(m+1)\times n} \tag{10}$$

and

$$p = \begin{bmatrix} b \\ \widehat{b} \end{bmatrix} \in \Re^{(m+1)\times 1} \tag{11}$$

Let $\widehat{x} = (A^TA)^{-1}A^Tb$ be the least squares solution of the system:

$$Ax = b$$

and let $\mathcal{F}$ be the **operator corresponding to the Kalman Filter**, *defined as:*

$$\mathcal{F} : x \mapsto \mathcal{F}(x) = \widehat{x} + k(\widehat{b} - a^Tx)$$

where $k \in \Re^{n\times 1}$ is:

$$k = \frac{1}{1 + a^T(A^TA)^{-1}a}(A^TA)^{-1}a.$$

The least squares solution of the system $Mz = p$, ie the vector

$$\widetilde{z} = (M^TM)^{-1}M^Tp$$

is:

$$\widetilde{z} = \mathcal{F}(\widehat{x}) \tag{12}$$

In addition:

$$(M^TM)^{-1} = [I - ka^T](A^TA)^{-1} \tag{13}$$

Before proving the proposition we give the following:

Lemma: *Let I be the identity matrix and c, d two vectors then:*

$$[I + dc^T]^{-1} = I + kdc^T$$

where

$$k = -(1 + c^T d)^{-1}$$

Dim.:
To prove the result, we show that

$$[I + dc^T][I + kdc^T] = I$$

assuming

$$k = -(1 + c^T d)^{-1}$$

We have:

$$[I+dc^T][I+kdc^T] = I+kdc^T+dc^T+dc^T kdc^T =$$

$$= I+kdc^T+dc^t+kd(c^T d)c^T = I+[k(1+c^T d)+1]dc^T$$

The latter expression is equal to the identity matrix I because

$$k = -(1 + c^T d)^{-1}$$

♣

Corollary: *Let B be an invertible matrix, we have:*

$$B^{-1}[I + dc^T]^{-1} = B^{-1} + kB^{-1}dc^T$$

♣
Also, if you put $e^T = c^T B$ it follows:

$$[B + dc^T]^{-1} = B^{-1} + kB^{-1}de^T B^{-1}$$

We are now in a position to prove Proposition 2.3:

proof Proposition 2.3: Compute

$$\widehat{z} = (M^T M)^{-1} M^T p \quad .$$

From the solution of $M^T M$ and $M^T p$ we have:

$$M^T M = A^T A + aa^T$$

and

$$M^T p = A^T b + a\widehat{b}$$

so:

$$z = (A^T A + aa^T)^{-1}(A^T b + a\widehat{b})$$

We apply Lemma with $d = e = a$ and $B = A^T A$:

$$(A^T A+aa^T)^{-1} = [(A^T A)^{-1}+k(A^T A)^{-1}aa^T(A^T A)^{-1}]$$

where:

$$k = k = \frac{1}{1 + a^T(A^T A)^{-1}a}(A^T A)^{-1}a$$

performing products and considering that

$$x = (A^T A)^{-1} A^T b$$

follows the thesis. ♣

In conclusion, KF aims to calculate the solution of system (4), performing the following steps:

$$k = \frac{1}{1 + a^T(A^T A)^{-1}a}(A^T A)^{-1}a \qquad (14)$$

$$\widetilde{x} = x + k(\widehat{b} - a^T x) \qquad (15)$$

We compute the time complexity needed for the calculation of $\widetilde{x}$ in (15):

1. computation of $B = (A^T A)$ (BLAS3: product of matrices): $O(n^2 m)$ flop
2. computation of B^{-1} (BLAS3: matrix inversion): $O(n^3)$ flop
3. computation of $a' = Ba$ (BLAS2: matrix vector product): $O(n^2)$ flop
4. computation of $a'' = a^T a'$ (BLAS1: product between vectors): $O(n)$ flop
5. computation of $1+a''$(BLAS1: sum of vectors): $O(n)$ flop
6. computation of $p = \frac{1}{1+a''}$ (BLAS1: constant $\times$ vector product): $O(n)$ flop
7. computation of $k = p \cdot a'$ (BLAS1: constant $\times$ vector product): $O(n)$ flop

obtaining a total of:

$$T_{costr\,k}(n) = n^3 + n^2 m + n^2 + 4n$$

For the computation of k in (14):

1. computation of $u = a^T \widehat{x}$ (product between vectors): $O(n)$ flop
2. computation of $u' = \widehat{b} - u$ (difference between two constants): $O(1)$ flop
3. computation of ku' (product between constant): $O(1)$ flop
4. computation of $\widetilde{x} = x + ku'$ (constant vector summ): $O(n)$ flop

obtaining a total of

$$T_{\widetilde{x}} = 2n$$

Therefore, we get:

$$T_{sol\,kalman}(n) = T_{costr\,k} + T_{\widetilde{x}} =$$

$$= O(n^3 + n^2 m + n^2 + 6n)$$

If $m >> n$, the performance gain is significant.

CASE STUDY 2:

We consider the problem:

$$P: \quad z = argmin_x\{\|Ax - b\|_2 + \|a^T x - \widehat{b}\|_2\} \tag{16}$$

Assuming that we add to the system $Ax = b$, s equations where $s > 1$, the problem (16) is well defined if we assume that:

1. $\widehat{b} \equiv d \in \Re^s$,
2. $a^T \equiv V \in \Re^{s \times n}$,

	Standard
T_{costr}	$(m+1) \times n^2 + n \times (m+1)$
T_{sol}	n^3

	Kalman Complete
T_{costr}	$n^3 + n^2 m + n^2 + 4n$
T_{sol}	$2n$

	Kalman update
T_{costr}	$n^2 + 4n$
T_{sol}	$2n$

Table 3: *Time complexity of the construction and resolution of the system of normal equations (4) in a standard formulation, with KF in which you need to recalculate all operators (Kalman Complete) and Kalman in which you are only updating the solution (Kalman update).*

The least square problem (16) becomes:

$$P: \quad z = argmin_x\{\|Ax - b\|_2 + \|Vx - d\|_2\} \tag{17}$$

This is a DA problem which consists of a **model + constraint**, where the model is expressed by the system $Ax = b$ and the constraint equations by the system $Vx = d$.

The following proposition applies KF to the DA problem in the form (17),

Propostion 0.2 *Let us consider the DA problem:*

$$P: \quad z = argmin_x\{\|Ax - b\|_2 + \|Vx - d\|_2\} \tag{18}$$

where $A \in \Re^{m \times n}$, $b \in \Re^m$, $V \in \Re^{s \times n}$, $d \in \Re^s$, let $\widehat{x} = (A^T A)^{-1} Ab$ be the least square solution of the system $Ax = b$. The solution of the problem (18)

$$\widehat{z} = (A^T A + V^T V)^{-1}(A^T b + V^T d)$$

is:

$$\widehat{z} = \mathcal{F}(\widehat{x})$$

with

$$\mathcal{F} : z \mapsto \widehat{x} + K(Vx - d)$$

and

$$K = \frac{1}{I + V(A^TA)^{-1}V}(A^TA)^{-1}V^T$$

.

Proof: The proof follows the steps shown for the Proposition 2.3 with the difference that $V \in \Re^{s\times n}$ and $d \in \Re^s$. ♣

CASE STUDY 3:

Observation: *To compute the vector minimizing the residual $r = Ax - b$ is equivalent to compute the vector of minimum distance from $\widehat{x}$, or:*

$$\min \|Ax - b\|_2 \Leftrightarrow \min \|x - \widehat{x}\|_2$$

with $\widehat{x} = (A^TA)^{-1}A^Tb$ (just replace the expression of $\widehat{x}$ in the Euclidean norm and multiply vectors $x-\widehat{x}$ and $(A^TA)^{-1}A^T$), so problem (18) is equivalent to that described below:

$$P: \quad z = argmin_x\{\|x - \widehat{x}\|_2 + \|Vx - d\|_2\}$$

Although, from a mathematical point of view, the problem is the same, this formulation highlights the fact that the vector $\widehat{x}$ is available or has been calculated. In other words, it appears more evident the characteristic of the least squares approximation problem that we are solving using KF: to compute z as an upgrade of the solution $\widehat{x}$.
With this formulation, the derivation of the KF is straightforward:

Propostion 0.3 *Given the following problem:*

$$P: \quad z = argmin_x\{\|x - \widehat{x}\|_2 + \|Vx - d\|_2\}$$

where $\widehat{x} = (A^TA)^{-1}A^Tb$, then we have:

$$z = \widehat{x} + K(V\widehat{x} - d)$$

and

$$K = V^T(I + V^TV)^{-1}$$

Proof: Explaining the euclidean norm, we have:

$$\{\|x - \widehat{x}\|_2 + \|Vx - d\|_2\} =$$

$$= (x - \widehat{x})^T(x - \widehat{x}) + (Vx - d)^T(Vx - d)$$

Performing products, it follows that:

$$(x - \widehat{x})^T + (Vx - d)^T(Vx - d) =$$

$$= x^Tx - x^T\widehat{x} - \widehat{x}^Tx - \widehat{x}^T\widehat{x} +$$

$$+x^TV^TVx - x^TV^Td - d^TVx + d^Td$$

computing the derivative with respect to x, we get:

$$2x - 2\widehat{x} + 2V^TVx - V^Td + d^TV;$$

by requiring that this derivative is zero, we obtain z:

$$2(V^TV + I)z = 2V^Td + 2\widehat{x}$$

Adding and subtracting $V^TV\widehat{x}$, it follows:

$$(V^TV + I)x = V^Td + \widehat{x} + V^TV\widehat{x} - V^TV\widehat{x}$$

or:

$$(V^TV + I)x = (I + V^TV)\widehat{x} + V^T(V\widehat{x} - d),$$

multipling $(V^TV + I)$ at the left and at the right:

$$z = \widehat{x} + (I + V^TV)^{-1}[V^T(V\widehat{x} - d)]$$

or, posed:

$$K = (I + V^TV)^{-1}V^T$$

we finally have:

$$z = \widehat{x} + K(V\widehat{x} - d)$$

♣

In this form, the calculation of z is the update of the vector $\widehat{x}$.

CASE STUDY 4:

We consider hereafter the following:

Propostion 0.4 *Given the least squares problem:*

$$P: \quad z = argmin_x\{\|x-\widehat{x}\|_B + \|Vx-d\|_R\}$$

where instead of the standard Euclidean norm we use the weighted Euclidean norm:

$$\|z\|_S = z^T S z$$

(norm induced by the Mahalanobis distance (cfr. Appendix)) where as the weight matrix S we used R and B,two symmetric and positive definite matrices. If $\widehat{x} = (A^T B A)^{-1} A^T b$, then:

$$z = \widehat{x} + K(V\widehat{x} - d)$$

with:

$$K = RV^T(B + V^T RV)^{-1}$$

Proof: Explaining the weighted euclidean norm, we have:

$$\{\|x-\widehat{x}\|_B + \|Vx-d\|_R\} =$$

$$= (x-\widehat{x})^T B(x-\widehat{x}) + (Vx-d)^T R(Vx-d).$$

Performing products, it follows that:

$$(x-\widehat{x})^T B(x-\widehat{x}) + (Vx-d)^T R(Vx-d) =$$

$$= x^T Bx - x^T B\widehat{x} - \widehat{x}^T Bx - \widehat{x}^T B\widehat{x} +$$

$$+x^T RV^T Vx - x^T RV^T d - d^T RVx + d^T Rd.$$

By computing the derivative with respect to x, it follows:

$$2Bx - 2B\widehat{x} + 2V^R Vx - V^T Rd + d^T RV$$

and by requiring that this derivative is zero, so we obtain z:

$$2(V^T RV + B)z = 2V^T Rd + 2B\widehat{x}.$$

Adding and subtracting $V^T RV\widehat{x}$:

$$(V^T RV+B)x = V^T Rd+B\widehat{x}+V^T RV\widehat{x}-V^T RV\widehat{x}$$

or:

$$(V^T RV+B)x = (B+V^T RV)\widehat{x}+V^T R(V\widehat{x}-d)$$

multipling $(V^T RV + B)$ at the left and at the right:

$$z = \widehat{x} + (B + V^T RV)^{-1}[RV^T(V\widehat{x} - d)]$$

or, posed:

$$K = (B + V^T RV)^{-1} RV^T$$

we finally have:

$$z = \widehat{x} + K(V\widehat{x} - d)$$

♣

CASE STUDY 5:

KF is also referred to as **predictor-corrector** or **predict** (using an appropriate model) - **correct** (with measures). Here after we will show where does this terminology illustrating an application, which is the classic application where this operator is used:

Definition 0.8 *Let $t_i \in [0,T]$, $i = 1,\ldots,n$, we want to compute a vector of n components $x(t_i) \in \Re^n$ such that:*

$$x(t_i) = \Phi(t_{i-1}, t_i)x(t_{i-1}) + u(t_i) \quad (model)$$

and such that:

$$y(t_i) = V(t_i)x(t_i) + v(t_i) \quad (misure)$$

where, for all fixed values of i:

1. *Φ is a known matrix of dimension $n \times n$, said transition matrix,*
2. *$u(t_i), v(t_i) \in \Re^n$ are unknown vectors of dimension n, of random variables with Gaussian distribution with covariance matrix U and R respectively.*

3. *$V(t_i)$ is a known matrix of dimension $p \times n$, with $p \ll n$,*

4. *$y(t_i)$ is a known vector of dimension p, with$p \ll n$.*

This problem is equivalent to compute $z(t_i)$, such that it is minimum of:

$$F : \|x(t_i) - \Phi(t_{i-1}, t_i)x(t_{i-1})\|_U +$$

$$+\|y(t_i) - V(t_i)x(t_i)\|_R$$

with $\|\cdot\|_U$ and $\|\cdot\|_R$ rules induced by the Mahalanobis distance (cfr. paragraph 3) defined as:

$$\|z\|_S = z^T S z$$

We have already seen that the solution is:

$$z(t_i) = \widehat{x(t_i)} + K[y(t_i) - V(t_i)\widehat{x(t_i)}]$$

with

$$\widehat{x(t_i)} = \Phi(t_{i-1}, t_i)x(t_{i-1})$$

or $\widehat{x(t_i)}$ is the solution of the model (for this reason we say that it is the forecast) and $z(t_i)$ is the correction of it obtained using the measures $y(t_i)$.

A feature of KF is that it provides, in addition to the solution, also an estimate of the error on this solution, measured in euclidean norm.

For clarity of speech, recall the concept of covariance matrix and the expected value.
Let ϵ_i, $i = 1, \ldots, n$ a vector of random variables,the matrix of covariances $H = (h_{ij})$ relative to the vector ϵ is defined such that the element h_{ij} is:

$$h_{ij} = COV(\epsilon_i, \epsilon_j)$$

and if X and Y are two random variables, $COV(X, Y)$, is the expected value of the products of the distances X and Y from the mean:

$$COV(X, Y) = E[(X - E(X)(Y - E(Y))]$$

In probability theory the expected value (also called the media, expectation or mathematical expectation) of a random variable is a number that formalizes the heuristic idea of the medium value. In general, the expected value of a discrete random variable (ie assume that only a finite number of values or a countable infinity) is given by the sum of the possible values of that variable, each multiplied by the probability of being engaged (ie to occur), ie the weighted average of the possible results.

Corollary 0.1 *Let $z(t_i)$ be the solution obtained using KF and, let P_i be the covariance matrix of the error $e(t_i) = z(t_i) - x(t_i)$, or P_i is the matrix such that:*

$$P_i = E[e(t_i)e(t_i)^T]$$

with E =expected value ,so we have:

$$P_i = (I - KV)P_{i-1} \quad (P_0 = 0)$$

THE KF NUMERICAL ALGORITHM

All problems that we have been presented can be formulated in the following form:

Definition 0.9 *Given x_0, compute,for all value of $j = 0, 1, 2, ...$, the vector x_{j+1} such that:*

$$x_{j+1} = Mx_j + w_j \quad j = 0, 1, \ldots$$

and such that:

$$y_{j+1} = Cx_{j+1} + v_{j+1}$$

where, for all fixed value of j:

- *$x_j \in \Re^N$ (unknown of the problem),*
- *$y_j \in \Re^P$ (data of the problem),*
- *$w_j \in \Re^N$ has covariance matrix $Q_j \in \Re^{N \times N}$ positive definite (data of the problem),*

- $v_j \in \Re^p$ *has covariance matrix* $R_j \in \Re^{p\times p}$ *(known) positive definite (data of the problem),*
- $M \in \Re^{N\times N}$ *(data of the problem),*
- $C \in \Re^{p\times p}$ *(data of the problem).*

the algorithm based on KF for the calculation of x_j *proceeds in this way*[2] *:*

1. *posed* $P_0 \equiv 0$
2. *for* $j = 0, 1, 2, \ldots$
3. *compute the vector* $x'_{j+1} = Mx_j$ *(prediction provided by the model)*
4. *compute the matrix*

$$\widetilde{P_{j+1}} = MP_jM^T + Q_j$$

(estimation error on x'_{j+1}*, prediction of the model)*

5. *compute the matrix* $\widetilde{R_{j+1}} = R_{j+1} + C\widetilde{P_{j+1}}C^T$ *(auxiliary matrix)*
6. *compute the matrix*

$$K_{j+1} = \widetilde{P_{j+1}}C^T[\widetilde{R_{j+1}}]^{-1}$$

(Kalman matriz)

7. *compute the vector :*

$$x_{j+1} = x'_{j+1} + K_{j+1}[y_{j+1} - Cx'_{j+1}]$$

(solution obtained using Kalman)

8. *compute matrix:*

$$P_{j+1} = (I - K_{j+1}C)\widetilde{P_{j+1}}$$

(calculating the estimate of the solution obtained using KF)

In order to formulate a version of this algorithm that is actually feasible in a finite precision arithmetic system we need to preliminarily analyze its numerical stability and its computational cost.

COMPUTATIONAL COST

Here after we show for each operation (3-8), the computational cost expressed in terms of the number of floating point operations (floats) [11].

3. the vector x'_{j+1} is the result of a matrix-vector product (BLAS1):

$$O(N^2) \quad float$$

4. the matrix $\widetilde{P_{j+1}}$ is the result of a 2 matrix-matrix product (BLAS3)

$$O(N^3) \quad float$$

and 1 sum of matrices (BLAS3)

$$O(N^2) \quad float$$

5. the matrix $\widetilde{R_{j+1}}$ is the result of a 2 matrix-matrix product (BLAS3)

$$O(N^2p + Np^2) \quad float$$

and sum of matrices (BLAS3)

$$O(p^2) \quad float$$

6. the matrix K_{j+1} is the result of a matrix-matrix product (BLAS3):

$$O(N^2p) \quad float$$

and a matrix inversion (BLAS3):

$$O(p^3) \quad float$$

7. the vector x_{j+1} is the result of a matrix-vector product (BLAS2):

$$O(Np) \quad float$$

a subtraction of vectors (BLAS1):

$$O(p) \quad float$$

[2] For simplicity of notation we assume that the matrix M and C does not depend on j.

a matrix vector product (BLAS1)

$$O(Np) \quad float$$

a sum of vectors (BLAS1):

$$O(N) \quad float$$

8. the matrix P_{j+1} is the result of a matrix-matrix product (BLAS3):

$$O(N^2p) \quad float$$

a subtraction of matrix (BLAS3)

$$O(N^2) \quad float$$

a product of matrices (BLAS3):

$$O(N^3) \quad float$$

Assuming that the execution time of one float is the same if we calculate multiplications/divisions or additions/subtractions, it follows that the cost of each step of the algorithm can be summarized as:

3. $O(N^2) \quad float$
4. $O(N^3) + O(N^2) \quad float$
5. $O(N^2p + Np^2) + O(p^2) \quad float$
6. $O(N^2p) + O(p^3) \quad float$
7. $O(Np) + O(p) + O(Np) + O(N) \quad float$
8. $O(N^2p) + O(N^2) + O(N^3) \quad float$

total:

$$O(2N^3+4N^2p+3N^2+Np^2+2Np+N+2p^3+p^2+p)$$

CONDITIONING AND STABILITY

The algorithm implementing the Kalman filter, is known in the literature as the "conventional" or the "classic" implementation of the Kalman filter (CFK). The literature shows that this algorithm is **unstable**. In fact, since 10 years after the introduction of KF (i.e. in 1970), were known (even if only experimentally) the causes and effects of roundoff error propagation on the solution. In this regard, it was used the term *"divergence"* to characterize the roundoff error propagation which leads to covariance matrices not symmetric and not positive definite [2]. Another effect of the roundoff error propagation is seen in the order of magnitude of the elements of the covariance matrix of the solution calculated by the algorithm. These elements become numerically equal to zero and this obviates the role of the matrix itself.

In the following, we review the stability of the algorithm performing the **forward error propagation analysis**, pointing out that the roundoff error propagation depends on the matrix M which describes the forecasting model. In other words, the **algorithm is backward stable**, but if the forecasting problem is ill-conditioned the solution calculated by the algorithm is not accurate.

We perform the stability analysis of the algorithm analyzing the error propagation of roundoff in a single time step and then we extend the results of this analysis for any number of time steps. This assumptions means that the matrices defining the problem to the current step are represented exactly in the finite precision arithmetic system, or are not affected of error propagated in the previous steps.

So we start from the computation of the vector

x'_{j+1} (step 7.) and of the error matrix associated with the vector (step 8.). We observe here after:

- At step 7 and step 8, the Kalman matrix K_{j+1} is the only variable containing the roundoff error (the other quantities are known a priori) so, the roundoff amplification factor in these two steps depends essentially on this matrix, calculated in step 6.
- The roundoff error on the Kalman matrix depends on the matrix $\widetilde{P_{j+1}}$ and on $\widetilde{R_{j+1}}$, which are the only quantities that are actually calculated.
- The matrix $\widetilde{R_{j+1}}$ computed at step 5, and the error propagated on this matrix derives from the error on the matrix $\widetilde{P_{j+1}}$, computed at step 4.
- At step 4, the computation of $\widetilde{P_{j+1}}$ is affected by roundoff error which propagates in the execution of operations of this step, i.e. the product of matrices and matrix addition. The roundoff error propagation of these operations depends on the matrix M that describes the forecasting model.

This analysis conducted at a macroscopic level, highlights that the roundoff error amplification of the single step of the algorithm is due to the matrix M, and in particular is due to the fact that this algorithm is sensitive to error propagation, not only due to the finite-precision arithmetic system (roundoff errors) but also the errors introduced in the construction of M (approximation, linearization, etc.). Here after we will highlight that the errors amplification factor is the conditioning number of M. The effect of this error propagation is seen in the result of step 8: the matrix $\widetilde{P_{j+1}}$ is not symmetric. This result is demonstrated using the *forward error analysis*, on the algorithm that implements KF, (see [12]). Specifically, in [12] is shown the following result:

Theorem 0.1 *Let $\delta(P_j)$, $\delta(K_j)$, $\delta(x_j)$ roundoff (absolute) error introduced, respectively on P_{j+1}, K_{j+1} and x_{j+1}, during the computation of these quantities in a finite-precision arithmetic system. Let $\Delta(P_j)$, $\Delta(K_j)$, $\Delta(x_j)$ the Euclidean norms of these errors. Through the analysis of roundoff error propagation in the execution of operations of KF in a finite-precision arithmetic system with maximum accuracy u, we have:*

$$\begin{aligned} \Delta(P_j) &= \leq u \cdot \sigma_1^2/\sigma_p^2 \|\widetilde{P_{j+1}}\| \\ \Delta(K_j) &= \leq u \cdot \sigma_1^2/\sigma_p^2 \|K_j\| \\ \Delta(x_j) &= \leq u \cdot (\|F_j\| \cdot \|x_j\| + \|K_j\| \cdot \|y_j\|) + \\ &\quad + \|\Delta(K_j)\| \, (\|C\| \cdot \|x_j\| + \|y_j\|) \end{aligned} \tag{19}$$

with

$$F_j = M - K_j C$$

and σ_i, $i = 1, ..., p$ are the singular values of the Cholesky factor of matrix R_j.

Hereafter we presents an interesting result [12]:

Corollary 0.2 *We get the same limitations if we assume that the matrices M, C, Q_j and R_j are affected by errors of any kind, or if:*

$$\|\delta M\| \leq \epsilon_1 \|M\| \quad \|\delta C_j\| \leq \epsilon_2 \|C_j\|$$

$$\|\delta Q_j\| \leq \epsilon_3 \|Q_j\| \quad \|\delta R_j\| \leq \epsilon_4 \|R_j\|$$

con

$$\epsilon_i \leq u \quad i = 1, 2, 3, 4$$

Proof: these errors are similar to roundoff errors introduced by arithmetic operations, so we get (14).
We observe from (14) that the errors amplification depends on:

- $\sigma_1^2/\sigma_p^2 = \mu(R_j^{1/2})^2 \simeq \mu(R_j)$, with $R_j^{1/2} =$ Cholesky factor of R_j.
- $||F_j|| \leq \rho(F_j) \simeq \rho(M)$

where $\rho(M)$ is the spectral radius of the matrix M and $\mu(R_j)$ is the condition number of R_j. In summary, we deduce that the algorithm implementing KF in the standard version, is sensitive to roundoff error propagation. Moreover, we obtain this result even by performing the macroscopic analysis of the operations carried out by the algorithm.

- $\sigma_1^2/\sigma_p^2 = \mu(R_j^{1/2})^2 \simeq \mu(R_j)$, with $R_j^{1/2} =$ Cholesky factor of R_j.
- $||F_j|| \leq \rho(F_j) \simeq \rho(M)$

In literature there have been proposed many *algorithmic variants* of KF, all characterized by the common idea of

1. introducing a factorization of the covariance matrix calculated in step 8, to work only on one of the decomposition factors, to explicitly force the symmetry of the matrices involved. Some of the factorizations used, are:
   - the Cholesky factorization LL^T (variant known as *"square root"* (SRF);
   - the LDL^T factorization of symmetric matrix;
   - reductions in triangular matrices (QR, SVD, $TSVD$, reductions in triangular/diagonal matrices using Givens matrices, Householder matrices, unitary matrices, etc....)

These transformations can be regarded as a preconditioning of the problem, because they introduce factorizations based on orthogonal matrices that are *perfectly conditioned*. In this way, the algorithm described in steps 1-8 becomes more efficient too, because preconditioning reduces the size of the matrices.

APPENDIX

LEAST SQUARES APPROXIMATION PROBLEM

Hereafter we consider the Problem:

Definition 0.10 **(Gauss)** *We consider the following linear system, in matrix form:*

$$Ax = b \tag{20}$$

with $A = [a_{ij}] \in \Re^{m\times n}$, $b = (b_1, \ldots, b_m)$, $x = (x_1, \ldots, x_n)$, $\quad m >> n$.
We want to compute the vector x.

As the system (20) is overdetermined, in this case the problem is an ill-posed inverse problems.
Regularizing the problem, or considering for example of system (20) the least squares system:

$$\min \|r\|_2 = \min \|Ax - b\|_2$$

with $r =$ the residual vector, $r = Ax - b$, we obtain the solution of best approximation in the sense of least squares solving the system of normal equations [6]:

$$A^T Ax = A^T b \tag{21}$$

This solution is unique in the hypothesis that the matrix A has maximal rank (ie equal to the number of columns n).
So, the solution of the system (20) can be computed solving the system of linear equations (21). This system can be solved using any numerical algorithm that uses the properties of the matrix (symmetric, positive definite ...).

WEIGHTED LEAST SQUARES

Definition 0.11 **(Gauss)** *Assuming the same assumptions as in Problem 3.1.1, we consider the vector* b*, and we assume that it is affected by error (not significant). Suppose we have information on the degree of reliability (uncertainty) of each component of the known term* b*. Can we use this information in the calculation of the solution* x*?*

We define *"reliability"*, or uncertainty, in this context, the error perturbing data. The problem is therefore to use estimates of the error in the calculation of the solution x.

Hereafter we re-formulate the Problem as:

Definition 0.12 *Assuming the same assumptions as in Problem 3.1.1, we denote the vector* b *the "ideal" solution of the system* $Ax = b$ *and* $\widetilde{b} = b + \epsilon$ *the perturbed vector in which we have emphasized the presence of the error* $\epsilon \in \Re^n$*, with* $\epsilon = (\epsilon_1, \epsilon_2, \ldots, \epsilon_n)$*. We suppose that* $\widetilde{b}_2$ *is* more reliable *of* $\widetilde{b}_1$*, or that the error estimate of* $\widetilde{b}_2$ *is less than that of the error* $\widetilde{b}_1$*, ie:*

$$\epsilon_2 < \epsilon_1$$

Can we use this information in the calculation of the solution x*?*

The idea is to introduce the coefficients w_i, $i = 1, \ldots, n$ (called *weight*) that weigh properly the error information on the components of b. In our case, we observe that:

$$\epsilon_2 < \epsilon_1 \Rightarrow w_2 < w_1$$

and, we can also observe that:

$$r_i = \widetilde{b}_i - (Ax)_i = b_i + \epsilon_i - (Ax)_i = \epsilon_i$$

or, weigh the component of the error on the known term is equivalent to weigh the corresponding component of the residue. So:

$$\|r\|_2 = \|\epsilon\|_2$$

consequently, weigh the error component present on the known term is equivalent to weigh the corresponding component of the residue.

The least squares approximation is reflected in the calculation of the vector x that minimizes the euclidean norm *"weighing"* of the residue r, i.e.:

$$(w_1 r_1, w_2 r_2, \ldots, w_n r_n), \quad r_i = \widetilde{b}_i - (Ax)_i$$

$$i = 1, \ldots, n$$

Let

$$W = diag(w_1, w_2, \ldots, w_n)$$

we get:

$$Wr = (w_1 r_1, w_2 r_2, \ldots, w_n r_n)$$

so:

$$\min_x \|Wr\|_2 = (Wr)^T (Wr) \quad , \quad r = \widetilde{b} - Ax$$

In this case, normal equations (21), become:

$$(WA)^T WAx = (WA)^T Wx$$

which solution is expressed as:

$$x = (A^T \underbrace{W^T W}_{C} A)^{-1} A^T \underbrace{W^T W}_{C} \widetilde{b} = \qquad (22)$$

$$(A^T C A)^{-1} A^T C b$$

This system is a particular case ($W = I$) of the system (21). Hereafter we will refer to the known term vector b assuming it affected by errors.

THE COVARIANCE MATRICES AND THE MAHALANOBIS DISTANCE

The use of the euclidean distance, and of the euclidean norm, in the construction of the of least squares approximation is motivated primarily by the assumption that the errors are completely independent of one another, are distributed as a Gaussian, in particular, with zero mean and variance equal to 1 (white noise). In this case, the covariance matrix of the error coincides with the identity matrix.
In the general case in which it is assumed that there is a dependency between the components of the error on data, expressed through a specific covariance matrix, we replace the euclidean distance with the *Mahalanobis distance* calculated by using the covariance matrix .

The Mahalanobis distance is a distance measure introduced by *P. C. Mahalanobis* in 1936. Differs from the euclidean distance because it adds information on the correlations of the data set. This distance is defined as follows:

Definition 0.13 *Ley M a symmetric matrix with positive coefficients. The Mahalanobis distance is defined as:*

$$d_M(x, y) = (x - y)^T M^{-1} (x - y)$$

If M is a diagonal matrix, this distance is also called the **weighted euclidean distance**.

Rem 0.1 *The interpretation of the Mahalanobis distance when the weight matrix is diagonal is quite evident, in fact, in this case, it is a linear scaling of the euclidean distance. But in the more general case, this is not immediate. Let's look at an interesting geometric interpretation.*
Suppose we want to estimate the probability that a vector x of the plane belongs to a set Ω, which has

a known center O_Ω. Intuitively, we can say that, as x is closer to the O_Ω, the more likely that x belongs to that set. If, to measure the distance of x from the center of Ω we use the Euclidean distance

$$d = \|x - O_\Omega\|_2$$

this means that we are assuming that the set Ω is a sphere and all points contribute equally (ie have the same probability) to belong to this set. Assuming that O_Ω is placed in the origin of the Cartesian reference system O, the equation

$$d = \|x - O\|_2^2 = x^T x$$

defines the center of the axes and radius d. So, the hypothesis that all the points have equal probability of belonging to Ω is reflected in the fact that the distance of these points from the center of the whole represents a sphere.
We assign to each component x_i of x, weight p_i, and climb to the corresponding weight p_i. The euclidean norm of the vector x is:

$$d = \|x\|_D = x^T D^{-1} x \quad D = diag(p_1, p_2)$$

and in this case defines the ellipse centered at the origin coincident with the principal axes (or parallel) with the coordinate axes. In this case, the two weights represent the length of the axes of the ellipse. This means that the position in which you find the point x influences the probability that x is in $Omega$. In fact, in correspondence of the greater weight the ellipse is more flattened and then the point x must be closer to the center to be in the set, while at the major axis the point x can also be located at a greater distance from the center of $Omega$ also belongs to it. Finally, if you want to consider the correlations between the components of x, expressed by the covariance matrix S, the equation:

$$d = \|x\|_S = x^T S^{-1} x$$

defines an ellipsoid with the principal axes rotated with respect to the Cartesian axes. To verify that a point x is in this set we need to calculate its distance (euclidean) from the center of Ω and involving all components of x in the computation of this quantity. In other words, the Mahalabinos distance of a vector from the center of the reference system, in the general case, geometrically describes an ellipsoid with the coordinate axes not orthogonal.

Bibliography

[1] N. K. Nichols, *Matematical Concepts of Data Assimilation*, W. Lahoz et al eds. Data Assimilation, Springer-Verlag (Berlin), 2010.

[2] R.J. Fitzgerald - *Divergence of the Kalman filter*, IEEE Trans. on Automatic Control, Vol. 16 (6), 1971.

[3] C.F. Gauss - *Theoria Motus Corporum Coelestium*, 1809.

[4] R. E. Kalman - *A New Approach to Linear Filtering and Prediction Problem*, su Transaction of AMSE - Journal of Basic Engineering, 1960.

[5] N. Wiener - *The Extrapolation, Interpolation and Smoothing of Stationary Time Series,* John Wiley & Sons, Inc., New York, N.Y.,1949

[6] A. Murli - *Matematica Numerica: metodi, algoritmi e software*, liguori editore, Parte prima, 2009.

[7] A. Murli - *Matematica Numerica: metodi, algoritmi e software*, liguori editore, Parte seconda, 2012.

[8] A. Murli, L. D'Amore, V. De Simone - *The Wiener Filter and Regularization Methods for Image Restoration Problems*, International Conference on Image Analysis and Processing (ICIAP 1999), IEEE, 1999.

[9] H. W. Sorenson - *Least square estimation: from Gauss to Kalman*, IEEE Spectrum, Vol. 7, pp. 63-68, 1970.

[10] A. N. Tikhonov, V. Y. Arsenin - *Solution of ill posed problems*, Wiley, 1977.

[11] J. Mendel - *Computational requirements for a Discrete Kalman filter,* IEEE Transactions on Automatic Control, Vol. AC-16, N. 6, 1971.

[12] M. Verhaegen, P. Van Dooren cfr. Numerical aspects of different kalman filter implementations, 1986, IEEE Transactions on Automatic Control, Vol. AC-31, N.10.