KALMAN FILTERS

Author: Eugenio Rossini

Discrete Taylor Series

It is possible to use the Taylor series, of a given function x(t), evaluated at the point t_0 where the function and its derivatives are known:

$$x(t) = \sum_{k=0}^{\infty} \frac{1}{k!} \left(\frac{d^{(k)} x(\tau)}{d\tau} \right)_{\tau = t_0} (t-t_0)^k$$

to extrapolate the value of the function at the point t where it is unknown. From this expression is easy to obtain the s-order derivative:

$$\frac{d^{(s)} x(t)}{dt} = \sum_{k=s}^{\infty} \frac{1}{(k-s)!} \left(\frac{d^{(k)} x(\tau)}{d\tau} \right)_{\tau = t_0} (t-t_0)^{k-s} = \sum_{k=0}^{\infty} \frac{1}{k!} \left(\frac{d^{(k+s)} x(\tau)}{d\tau} \right)_{\tau = t_0} (t-t_0)^k$$

Considering also the case s=0, the above formula include the previous one. The corresponding discrete version of the Taylor series can be achieved putting t=nT, $t_0=n_0T$ and $\tau=mT$:

$$x(nT) = \sum_{k=0}^{\infty} \frac{1}{k!} \left(\Delta^{(k)} x(mT) \right)_{m = n_0} (nT-n_0T)^k$$

where:

$$\Delta^{(k)} x(mT) = \Delta(\Delta^{(k-1)} x(mT)) \qquad \Delta x(mT) = \frac{x(mT+T/2) - x(mT - T/2)}{T}$$

using:

$$\delta\Delta^{(k)} x(mT) = \delta (\delta^{(k-1)} x(mT)) \; \delta x(mT) = x(mT+T/2) - x(mT - T/2)$$

we can rewrite the previous expression as:

$$x(nT) = \sum_{k=0}^{\infty} \frac{1}{k!} \left(\delta^{(k)} x(mT) \right)_{m = n_0} (n-n_0)^k$$

or, in order to simplify the notation:

$$x_n = \sum_{k=0}^{\infty} \frac{1}{k!} \left(\delta^{(k)} x_m \right)_{m = n_0} (n-n_0)^k$$

as in the continuos case, considering also the case s=0, we obtain the more general formula:

$$\delta^{(s)} x_n = \sum_{k=0}^{\infty} \frac{1}{k!} \left(\delta^{(k+s)} x_m \right)_{m = n_0} (n-n_0)^k$$

A more compact notation is obtained putting: $\delta^{(s)}x_n = x^{(s)}{}_n$:

$$\boxed{x^{(s)}{}_n = \sum_{k=0}^{\infty} \frac{1}{k!} \left(x^{(k+s)}{}_m \right)_{m = n_0} (n-n_0)^k}$$

Considering a finite p-order Taylor expansion, the general formula above can be written as:

$$\begin{pmatrix} x^{(0)}_n \\ x^{(1)}_n \\ x^{(2)}_n \\ \dots \\ x^{(p)}_n \end{pmatrix} = \begin{pmatrix} 1 & n-n_0 & \frac{n-n_0}{2} & \dots & \frac{(n-n_0)^p}{p!} \\ 0 & 1 & n-n_0 & \dots & \frac{(n-n_0)^{p-1}}{(p-1)!} \\ 0 & 0 & 1 & \dots & \frac{(n-n_0)^{p-2}}{(p-2)!} \\ \dots & \dots & \dots & & \dots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} \begin{pmatrix} x^{(0)}_{n0} \\ x^{(1)}_{n0} \\ x^{(2)}_{n0} \\ \dots \\ x^{(p)}_{n0} \end{pmatrix}$$

or, in compact notation:

$$X_n = A\, X_{n0}$$

In particular, if $n-n_0=1$:

$$X_n = A\, X_{n-1} \text{ with: } A = \begin{pmatrix} 1 & 1 & \frac{1}{2} & \dots & \frac{1}{p!} \\ 0 & 1 & 1 & \dots & \frac{1}{(p-1)!} \\ 0 & 0 & 1 & \dots & \frac{1}{(p-2)!} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}$$

Predictor Equations

We will use the above recursive Taylor formula to predict the values X_n assuming that the values X_{n-1} are given. To this end we will use the apex "¯" to indicate the predicted quantity X_n and the apex "+" for the given quantity X_{n-1} (later we will use as "given" data the output of a "corrector" algorithm, this justify the notation):

$$\boxed{X_n^- = A\, X_{n-1}^+} \quad (I)$$

Stop the series expansion at the p^{th} order is equivalent to consider the derivatives of higher order of X_{n-1} equal to zero. We can take into account this fact introducing a vector error V in the estimation of X_n having zero mean and gaussian distribution:

$$X_n = A\, X_{n-1} + V$$

Then the prediction error, if X_n is the exact value, can be expressed as:

$$ERR_n^- = X_n^- - X_n = A\, X_{n-1}^+ - A\, X_{n-1} + V = A\, ERR_{n-1}^+ - V$$

Introducing the covariance matrices and exploiting the fact that ERR_{n-1}^+ and V are not correlated, we obtain:

$$P_n^- = E(ERR_n^- (ERR_n^-)^T) = E((A\, ERR_{n-1}^+ - V)(A\, ERR_{n-1}^+ - V)^T) =$$

$$E(A\, ERR_{n-1}^+ (ERR_{n-1}^+)^T A^T) - E(A\, ERR_{n-1}^+ V^T) - E(V (ERR_{n-1}^+)^T A^T) + E(V V^T) = A\, P_{n-1}^+ A^T + Q \quad \rightarrow$$

$$\boxed{P_n^- = A\, P_{n-1}^+ A^T + Q} \quad (II)$$

The relationship obtained is recursive so it requires an initialisation P_0^+. This can be done estimating the noise added to the signal for enough time before starting the algorithm. When some of the components of X_0 is not observable it is convenient assume high value (theoretically infinity) for the auto-covariance and cross covariance elements of P_0. Q can be used as a matrix parameter to tune the recursion.

Corrector Equations

It is possible to tale into account that some of the components of X_n is not observable introducing the matrix H. If we can measure only q<p variables Z_n out of p variable X_n we can put (H will be a q,p matrix):

$$Z_n = \begin{pmatrix} z^{(0)}_n \\ z^{(1)}_n \\ z^{(2)}_n \\ \dots \\ z^{(q)}_n \end{pmatrix} = \begin{pmatrix} 100\dots0 \\ 010\dots0 \\ 001\dots0 \\ \dots\dots\dots\dots \\ 000\dots0 \\ 000\dots0 \end{pmatrix} \begin{pmatrix} x^{(0)}_n \\ x^{(1)}_n \\ x^{(2)}_n \\ \dots \\ x^{(p)}_n \end{pmatrix} := H\, X_n$$

If the observable quantities are not the first ones the matrix H can be obviously rearranged in order to extract toward Z_n only the observable values of X_n. Furthermore we can consider the errors due to the measure of Z_n introducing the zero mean, gaussian distributed, vector W and putting:

$$Z_n = H\, X_n + W \qquad\qquad R = E(W\, W^T)$$

The difference between the measured values Z_n and the corresponding predicted values are called "innovations":

$$N_n = Z_n - H\, X_n^-$$

When the new measure Z_n is available we can combine Z_n and X_n^- in order to improve the estimation X_n^+ of the true value X_n of the process under exam. We try to do this using a linear combination of the two values (L and K depend on n but it is an habit use the notation L, K instead of L_n, K_n):

$$X_n^+ = L\, X_n^- + K\, Z_n$$

Where L is a (p,p) matrix and K is a (p,q) matrix called "Kalman gain". L can be determined immediately observing that:

$$ERR_n^+ = X_n^+ - X_n = L\,(ERR_n^- + X_n) + K\,(H\, X_n + W) - X_n = L\, ERR_n^- + (L + K\, H - I)\, X_n + K\, W$$

But the expectation values of the errors and of W are supposed null, so we have:

$$E(ERR_n^+) = E(ERR_n^-) = E(W) = 0\,; \quad E(X_n) \neq 0 \qquad \rightarrow \qquad L + K\, H - I = 0 \quad \rightarrow \qquad L = I - K\, H$$

$$\boxed{X_n^+ = (I - KH)\, X_n^- + K\, Z_n} \text{ (III)}$$

the matrix K is instead determined imposing that:

$$P_n^+ = E(ERR_n^+\, (ERR_n^+)^T) = \text{minimum}$$

The correction error can be expressed as:

$$ERR_n^+ = X_n^+ - X_n = (I - KH)\, X_n^- + K\, H\, X_n + K\, W - X_n = (I - KH)\, ERR_n^- + K\, W$$

Assuming ERR_n^- and W are not correlated, an explicit expression for P_n^+ is then the following:

$$P_n^+ = E(\,(I - KH)\, ERR_n^-\, (ERR_n^-)^T\, (I - KH)^T\,) + E(\,K\, W\, W^T\, K^T\,) = (I - KH)\, P_n^-\, (I - KH)^T + K\, R\, K^T$$

We need now to minimize P_n^+ with respect to the matrix K. Next paragraph shows that such a matrix K is:

$$\boxed{K = P_n^-\, H^T\, (\, H\, P_n^-\, H^T + R\,)^{-1}} \text{ (IV)}$$

So, substituting in the previous formula we have:

$$P_n^+ = (I\text{-}KH)\, P_n^- (I\text{-}KH)^T + K\, R\, K^T =$$

$$= P_n^- + P_n^- (KH)^T - KH\, P_n^- + KH\, P_n^- (KH)^T + K\, R\, K^T =$$

$$= P_n^- + P_n^- H^T K^T - KH\, P_n^- + K\,(\, H\, P_n^- H^T + R\,)\, K^T = \text{(using here the optimum K)} =$$

$$= P_n^- + P_n^- H^T K^T - KH\, P_n^- + P_n^- H^T K^T = P_n^- - KH\, P_n^- \rightarrow$$

$$\boxed{P_n^+ = (I - KH)\, P_n^-} \quad (V)$$

Covariance Minimization

For minimising P_n^+ with respect to the matrix K we mean the minimization of the variance of each error $(ERR_n^+)_k$ or equivalently (the variances are positive) the minimization of $Tr(P_n^+)$ with respect to the qp variables $(K)_{sr}$ s=1..p, r=1..q. Before prove the minimum condition for P_n^+ let us consider two partial results:

$$\frac{\partial\, Tr(KA)}{\partial K} = \left(\frac{\partial\, Tr(KA)}{\partial k_{rs}} \right) = \left(\frac{\partial}{\partial k_{rs}} \sum u \sum v\, k_{uv}\, A_{vu} \right) = (\, A_{sr}\,) = (\, A_{rs}^T\,) = A^T$$

$$\frac{\partial\, Tr(KAK^T)}{\partial K} = \left(\frac{\partial\, Tr(KAK^T)}{\partial k_{rs}} \right) = \left(\frac{\partial}{\partial k_{rs}} \sum u \sum v \sum h\, k_{uv}\, A_{vh}\, k_{hu}^T \right) =$$

$$= \left(\sum u \sum v \sum h\, (\, \delta_{uv}^{rs}\, A_{vh}\, k_{hu}^T + k_{uv}\, A_{vh}\, \delta_{uh}^{rs}\,) \right) = \left(\sum h\, (\, A_{sh}\, k_{hr}^T\,) + \sum v\, (\, k_{rv}\, A_{vs}\,) \right) =$$

$$= \left(\sum h\, (\, A_{sh}\, k_{hr}^T + k_{rh}\, A_{hs}\,) \right) = \left(\sum h\, (\, A_{hs}^T\, k_{rh} + k_{rh}\, A_{hs}\,) \right) = \left(\sum h\, k_{rh}\, (\, A_{hs}^T + A_{hs}\,) \right) = K\,(\, A^T + A)$$

Using these results we obtain:

$$\frac{\partial\, Tr(P_n^+)}{\partial K} = \frac{\partial\, Tr(\,(I\text{–}KH)\, P_n^- (I\text{–}KH)^T + K\, R\, K^T\,)}{\partial\, K} = (I - KH)\,(\, P_n^- + (P_n^-)^T\,)\,(-H^T) + K\,(\, R + R^T\,)$$

so putting the derivative equal to zero and considering that the matrices P_n^- and R are symmetric we have:

$$-2\,(I - KH)\, P_n^- H^T + 2\, K\, R = 0 \quad \rightarrow \quad - P_n^- H^T + K\, H\, P_n^- H^T + K\, R = 0 \quad \rightarrow$$

$$K = P_n^- H^T (H\, P_n^- H^T + R)^{-1}$$

Equations Summary and Multistep Prediction

The Kalman filter is realised implementing the five equations:

$$\boxed{X_n^- = A\, X_{n-1}^+} \quad \text{(I)} \qquad \text{Predictor}$$

$$\boxed{P_n^- = A\, P_{n-1}^+ A^T + Q} \quad \text{(II)} \qquad \text{Predictor}$$

$$\boxed{X_n^+ = (I - KH)\, X_n^- + K\, Z_n} \quad \text{(III)} \qquad \text{Corrector}$$

$$\boxed{K = P_n^- H^T (H\, P_n^- H^T + R)^{-1}} \quad \text{(IV)} \qquad \text{Gain}$$

$$\boxed{P_n^+ = (I - KH)\, P_n^-} \quad \text{(V)} \qquad \text{Corrector}$$

starting from X_0^+ and P_0^+ and applying first of all the two predictor equation in order to obtain the values X_1^- and P_1^-. Computing then the gain K and finally applying the corrector equation to have the corrected values X_1^+ and P_1^+. The algorithm then proceeds applying recursively the five equations. It can be used both to predict the new value X_n^- of the process and to obtain a better estimation X_n^+ of the signal X_n in presence of noise (usually the Kalman filter gives an X_n^+ whose variance is better than that observed value Z_n of X_n).

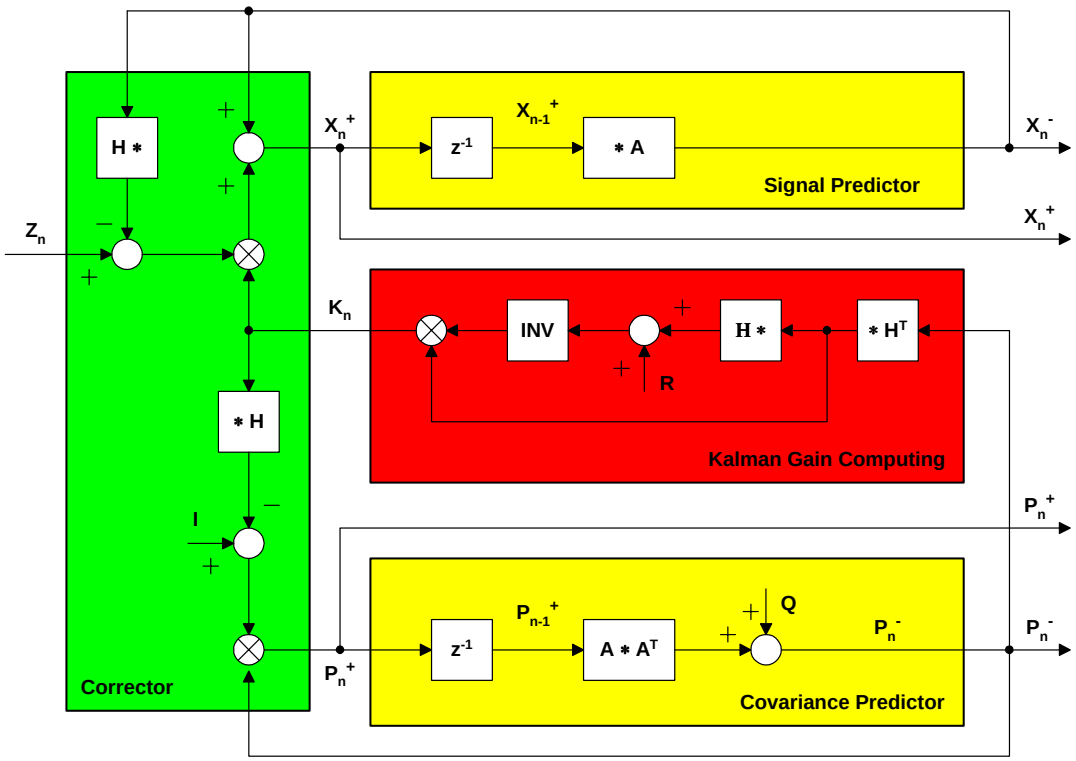
The Kalman filter can be also used as multistep predictor just adding a further equation to the previous ones. The basic five equations work as usual as a single step predictor-corrector and they allow the progress of the algorithm whilst at each step a multistep prediction can be obtained using the following equation:

$$\boxed{X_{n+n0-1}^- = \underline{A}\, X_{n-1}^+} \quad \text{(VI)} \qquad \text{Multistep Predictor}$$

where $\underline{A}$ is the matrix obtained from the Taylor expansion leaving the $n-n_0$ in it (that is without putting $n=n_0+1$). Note that contrary to the single step prediction X_n^-, the multistep prediction X_{n-n0+1}^- is not used in the next step of the algorithm but it only gives the output of the filter.

Block Diagram

Next picture shows a block diagram of the Kalman filter:



Transfer Function

The Kalman filter can be described by a transfer function when it reaches its steady state. In this state in fact the matrices P_n^- and P_n^+ can be considered as constants: $P_n^- = P_n^+ = P$ and so also the gain $K_n=K$. The equations (II), (IV) and (V) are no longer used and the filter is obtained from the equations (I) and (III):

$$\boxed{X_n^- = A\, X_{n-1}^+} \quad \text{(I)} \qquad \text{Predictor}$$

$$\boxed{X_n^+ = (I - KH)\, X_n^- + K\, Z_n} \quad \text{(III)} \qquad \text{Corrector}$$

Substituting (III) in (I) we have the transfer function of the predictor:

$$X_n^- = A\,(I - KH)\, X_{n-1}^- + A\,K\,Z_{n-1} \quad \rightarrow \quad (I - A\,(I - KH)\, z^{-1})\, X^-(z) = A\,K\,Z(z)\, z^{-1} \quad \rightarrow$$

$$\boxed{G(z)^- = X^-(z)\, Z(z)^{-1} = (I - A\,(I - KH)\, z^{-1})^{-1}\, A\,K\,z^{-1}} \qquad \text{Predictor Transfer Function}$$

Substituting (I) in (III) we have the transfer function of the corrector:

$$X_n^+ = (I - KH)\, A\, X_{n-1}^+ + K\, Z_n \quad \rightarrow \quad (I - (I - KH)\, A\, z^{-1})\, X(z)^+ = K\, Z(z) \quad \rightarrow$$

$$\boxed{G(z)^+ = X^+(z)\, Z(z)^{-1} = (I - (I - KH)\, A\, z^{-1})^{-1}\, K} \qquad \text{Corrector Transfer Function}$$

Second Order Kalman Filter with H=(1,0) – Equations (MAPLE)

To implement a 2nd order Kalman filter with observable variables defined by H=(1,0) the following scalar equation (see variable with "_update" appended) and definitions can be used (Maple exported as RTF).

Basic Matrices Definitions

```
> with(LinearAlgebra):
> A:= Matrix(2,2,[[1,1],[0,1]]);
> H:= Matrix(1,2,[1,0]);
> Pm:= Matrix(2,2,[[Pm11,Pm12],[Pm21,Pm22]]);
> Pp:= Matrix(2,2,[[Pp11,Pp12],[Pp21,Pp22]]);
> ID:= Matrix(2,2,[[1,0],[0,1]]);
> R:= Matrix(1,1,[r]);
> K:= Matrix(2,1,[[K1],[K2]]);
> Q:= Matrix(2,2,[[Q11,Q12],[Q21,Q22]]);
> Xm:= Matrix(2,1,[[Am],[Bm]]);
> Xp:= Matrix(2,1,[[Ap],[Bp]]);
```

$$A := \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \qquad H := [1 \quad 0] \qquad Pm := \begin{bmatrix} Pm11 & Pm12 \\ Pm21 & Pm22 \end{bmatrix} \qquad Pp := \begin{bmatrix} Pp11 & Pp12 \\ Pp21 & Pp22 \end{bmatrix} \qquad ID := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$R := [\, r \,] \qquad K := \begin{bmatrix} K1 \\ K2 \end{bmatrix} \qquad Q := \begin{bmatrix} Q11 & Q12 \\ Q21 & Q22 \end{bmatrix} \qquad Xm := \begin{bmatrix} Am \\ Bm \end{bmatrix} \qquad Xp := \begin{bmatrix} Ap \\ Bp \end{bmatrix}$$

Gain Computing

```
> K_1:= MatrixMatrixMultiply(Pm,Transpose(H)):
> K_2:= MatrixMatrixMultiply(H,K_1):
> K_update:=  MatrixMatrixMultiply(K_1,MatrixInverse(K_2+R));
```

$$K\_update := \begin{bmatrix} \frac{Pm11}{Pm11 + r} \\ \frac{Pm21}{Pm11 + r} \end{bmatrix}$$

Predictor Equations

```
> Xm_update:= MatrixMatrixMultiply(A,Xp);
> Pm_1:= MatrixMatrixMultiply(A,Pp):
> Pm_update:= MatrixMatrixMultiply(Pm_1,Transpose(A))+Q;
```

$$Xm\_update := \begin{bmatrix} Ap + Bp \\ Bp \end{bmatrix}$$

$$Pm\_update := \begin{bmatrix} Pp11 + Pp21 + Pp12 + Pp22 + Q11 & Pp12 + Pp22 + Q12 \\ Pp21 + Pp22 + Q21 & Pp22 + Q22 \end{bmatrix}$$

Corrector Equations

```
> Xp_1:= z - MatrixMatrixMultiply(H,Xm)[1,1]:
> Xp_update:= Xm + MatrixScalarMultiply(K,Xp_1);
> Pp_1:= ID - MatrixMatrixMultiply(K,H):
> Pp_update:= MatrixMatrixMultiply(Pp_1,Pm);
```

$$Xp\_update := \begin{bmatrix} Am + (z - Am)\, K1 \\ Bm + (z - Am)\, K2 \end{bmatrix} \qquad Pp\_update := \begin{bmatrix} (1 - K1)\, Pm11 & (1 - K1)\, Pm12 \\ -K2\, Pm11 + Pm21 & -K2\, Pm12 + Pm22 \end{bmatrix}$$

in these equations appendix “m” is for “minus” (predictor equations) and the appendix “p” is for “plus” (corrector equations). The algorithm starts computing first of all the predictor equations supposing its input values known at the 0-iteration.

Only SUMS are necessary to compute the prediction.

Then, using Xm and Pm, the algorithm continues computing the gain K.

SUMS and DIVISIONS are necessary to compute the gain.

Finally the corrector equations are calculated.

Only SUMS and MULTIPLICATIONS are necessary to compute the prediction.

The number of overall operations necessary to carry out one step of the kalman filter are reported in the next table (the number of operations are here considered optimised, that is the same operation is carried out just once):

	Sums	Multiplications	Divisions
Predictor	**9**	**-**	**-**
Gain	**1**	**-**	**2**
Corrector	**6**	**6**	**-**

Using Maple the following transfer function can be calculated for the predictor:

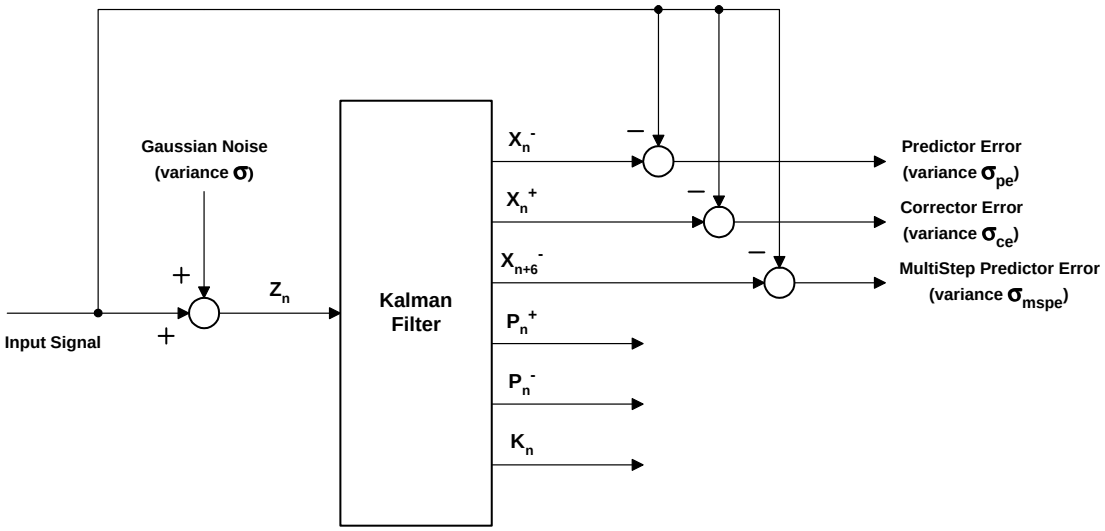
Transfer Function

```
> F_1:= MatrixMatrixMultiply(A,K):
> F_2:= MatrixMatrixMultiply(F_1,H):
> F_num:=  MatrixScalarMultiply(F_1,z^(-1)):
> F_den:=  ID - MatrixScalarMultiply(A-F_2,z^(-1)):
> F:=  MatrixMatrixMultiply(MatrixInverse(F_den),F_num):
> F11:= simplify(F[1,1]):
> F21:= simplify(F[2,1]):
> G:=Matrix(2,1,[[F11],[F21]]);
```

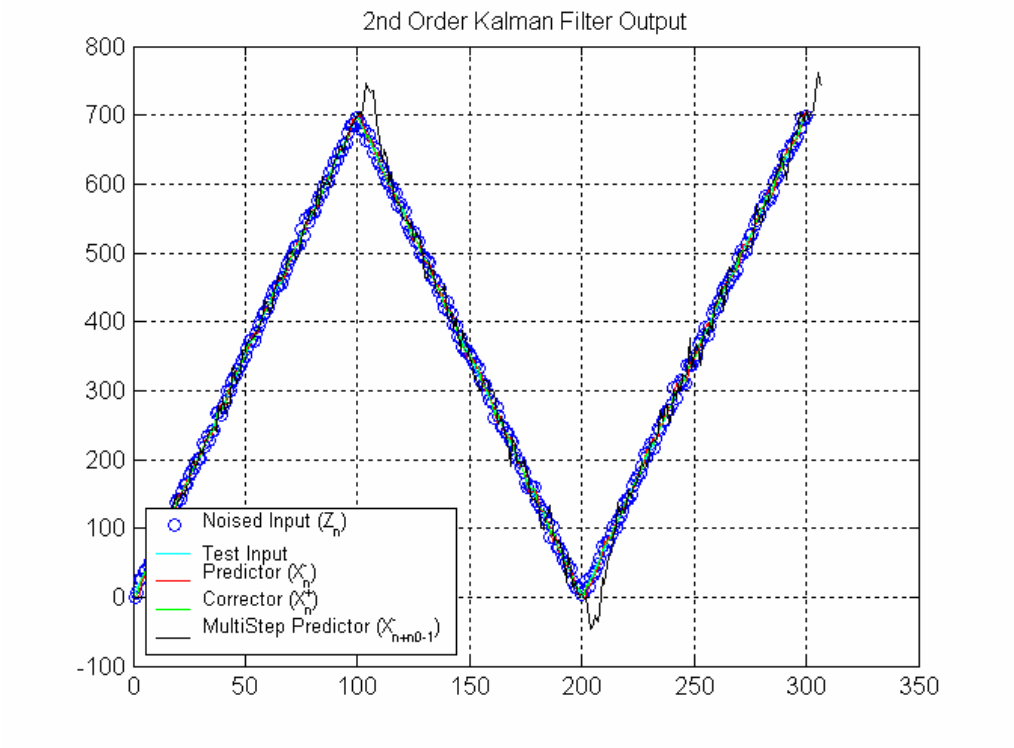
$$G := \begin{bmatrix} \dfrac{K1\,z + K2\,z - K1}{z^2 - 2\,z + 1 + K1\,z - K1 + K2\,z} \\ \dfrac{K2\,(z-1)}{z^2 - 2\,z + 1 + K1\,z - K1 + K2\,z} \end{bmatrix}$$

Second Order Kalman Filter with H=(1,0) – Simulations (MATLAB)

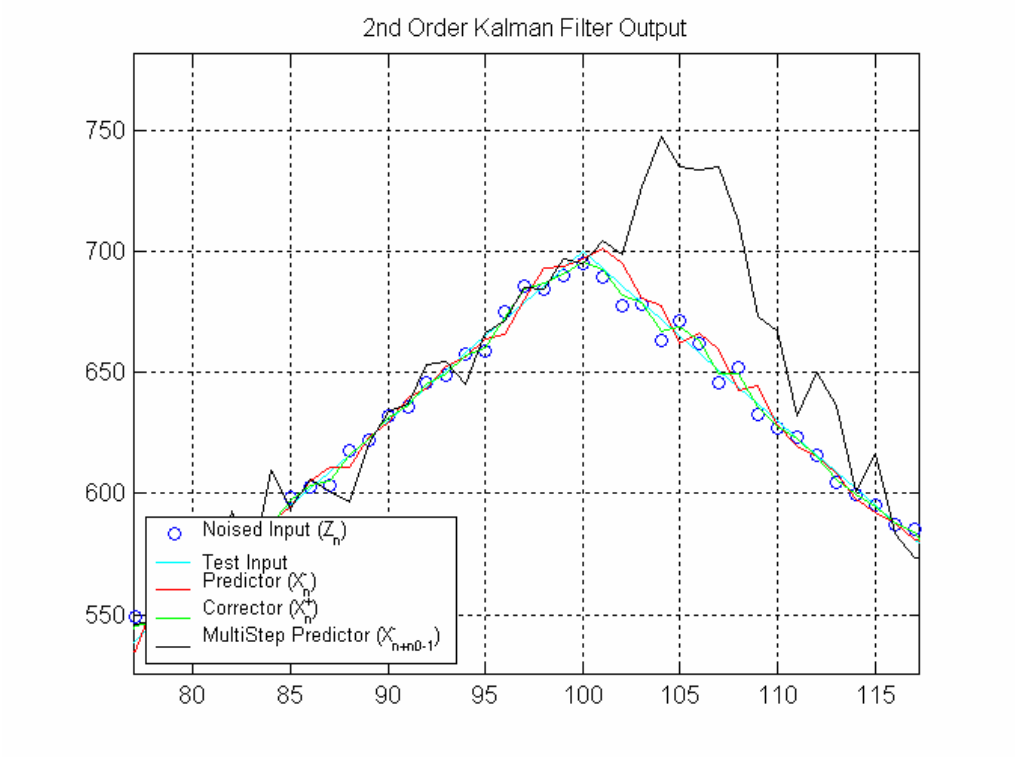
A MATLAB simulation of this filter highlighted the behaviour of each parameter of the filter. The following test block diagram has been simulated:



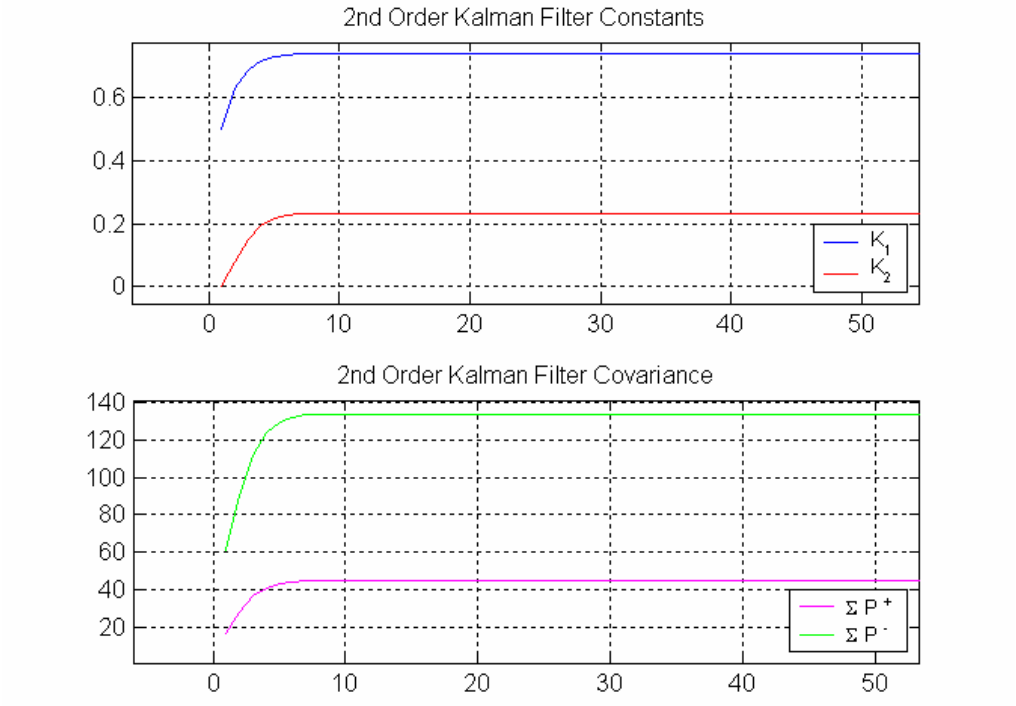
The input signal used during the simulation was a ramp as reported in the next figure:



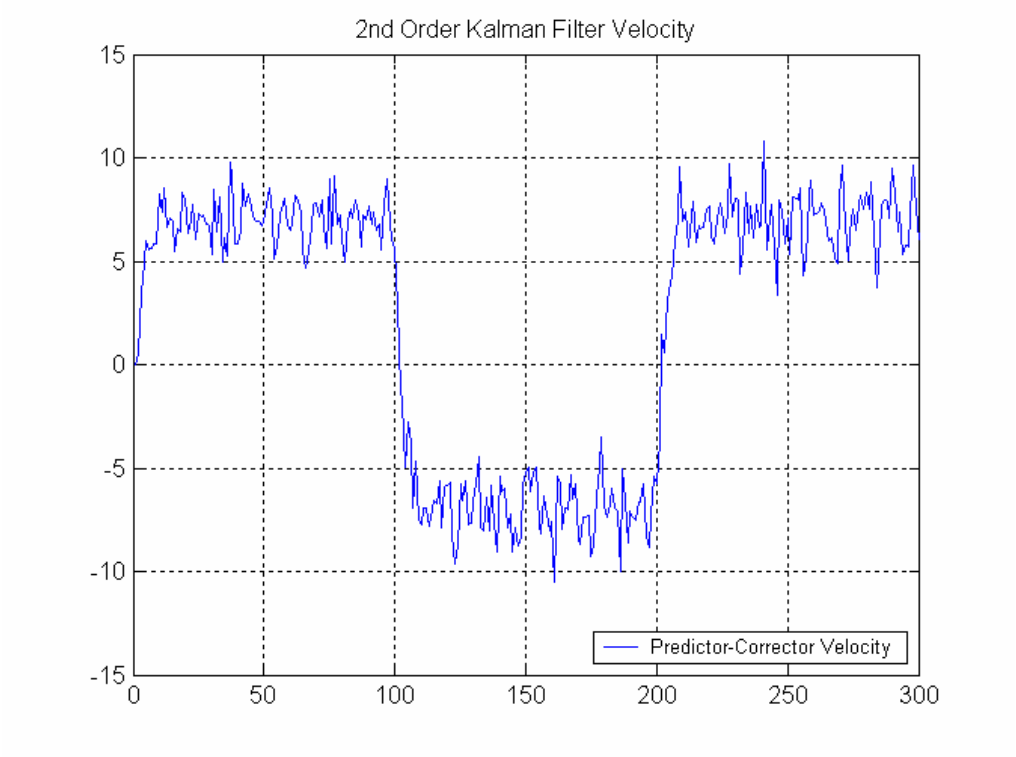
The multistep prediction carried out was of 7 steps (X_{n+6}^-) and it is possible to note that at the end of the ramp this causes a 7-steps prediction error. The single step predictor cause just a single step error in the same point of the ramp as we can see looking at the following zoomed figure:



The filter converges to an almost constant “kalman gain vector” and “covariance matrices”. Next picture report their evolution:



After just a few iterations both the gain and the covariance are stable. The covariance parameter plotted is the sum of all the (positive) elements of the matrix. The guessed velocity is shown in the next figure:



The errors at the output of the previous block diagram are reported in the next table:

Input Noise Variance (σ)	Single Step Predictor Error Variance (σ_{pe})	Single Step Corrector Error Variance (σ_{ce})	Multi Step Predictor Error Variance (σ_{mspe})
24.14	27.50	14.71	409.14

As we can note, the corrector output has an error variance less than the input noise variance, that is the filter has improved the estimation of the input noised signal. The variance of the single step predictor is not so good as the corrector because obviously the filter in this case tried to predict the value that the signal will have in the future. When the number of steps in the future increases, also the error variance increases due to the uncertainty of the forecast.

α-β Filters (2nd Order Kalman Filters with H=(1,0), K=constant)

If in a second order system we consider K1= α= constant and K2= β= constant (that is tuned at the beginning and not adaptive) and put:

$$X_n^- = (x_n^-, v_n^-)^T \qquad X_n^+ = (x_n^+, v_n^+)^T$$

Using the equations obtained with Maple for the predictor (Xm_update) and for the corrector (Xp_update) in the previous paragraph, the Kalman equations become (being z_n the filter input as usual):

$$x_n^+ = x_n^- + \alpha\,(z_n - x_n^-) \qquad \text{(a)} \qquad \text{Corrector}$$

$$v_n^+ = v_n^- + \beta\,(z_n - x_n^-) \qquad \text{(b)} \qquad \text{Corrector}$$

$$x_n^- = x_{n-1}^+ + v_{n-1}^+ \qquad \text{(c)} \qquad \text{Predictor}$$

$$v_n^- = v_{n-1}^+ \qquad \text{(d)} \qquad \text{Predictor}$$

Substituting the last equation in (b) we obtain the classical equations of an α-β filter:

$$x_n^+ = x_n^- + \alpha\,(z_n - x_n^-) \qquad \text{(a)} \qquad \text{Position Estimation}$$

$$v_n^+ = v_{n-1}^+ + \beta\,(z_n - x_n^-) \qquad \text{(b2)} \qquad \text{Velocity Estimation}$$

$$x_n^- = x_{n-1}^+ + v_{n-1}^+ \qquad \text{(c)} \qquad \text{Position Prediction}$$

If the filter is realised to estimate and predict moving objects could be necessary take into account the sampling interval T. This can be done simply making homogeneous the previous equations:

$$x_n^+ = x_n^- + \alpha\,(z_n - x_n^-) \qquad \text{(a)} \qquad \text{Position Estimation}$$

$$v_n^+ = v_{n-1}^+ + \beta\,(z_n - x_n^-)\,/\,T \qquad \text{(b3)} \qquad \text{Velocity Estimation}$$

$$x_n^- = x_{n-1}^+ + T\,v_{n-1}^+ \qquad \text{(c2)} \qquad \text{Position Prediction}$$

Predictor-Corrector Filters

The basic idea of a Kalman filter is in considering an approach predictor-corrector where the predictor is based on the Taylor series and the corrector on a linear combination of the predicted value X_n^- with the measured value Z_n when it becomes available. This basic idea is then summarised by the following two equations:

$$\boxed{X_n^- = A\, X_{n-1}^+} \quad \text{Predictor}$$

$$\boxed{X_n^+ = L\, X_n^- + K\, Z_n} \quad \text{Corrector}$$

If we simplify the Kalman approach making L and K constants, that is they are predefined coefficients instead of adaptive, we obtain a class of predictor-corrector filters. The classical filters used as smoothers in control loops can be for example obtained putting L=I. If we consider in fact the case of 2nd order (p=2) filters with a single input (q=1), and put:

$$X_n^- = (x_n^-, v_n^-)^T \qquad\qquad X_n^+ = (x_n^+, v_n^+)^T$$

the previous equations give:

$$\begin{pmatrix} x_n^+ \\ v_n^+ \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_n^- \\ v_n^- \end{pmatrix} + \begin{pmatrix} k_1 \\ k_2 \end{pmatrix} z_n \quad \text{Corrector}$$

$$\begin{pmatrix} x_n^- \\ v_n^- \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_{n-1}^+ \\ v_{n-1}^+ \end{pmatrix} \quad \text{Predictor}$$

passing to the scalar equations we have:

$$x_n^+ = x_n^- + k_1\, z_n \qquad \text{(A)} \qquad \text{Corrector}$$

$$v_n^+ = v_n^- + k_2\, z_n \qquad \text{(B)} \qquad \text{Corrector}$$

$$x_n^- = x_{n-1}^+ + v_{n-1}^+ \qquad \text{(C)} \qquad \text{Predictor}$$

$$v_n^- = v_{n-1}^+ \qquad \text{(D)} \qquad \text{Predictor}$$

substituting (A) in (C) and (B) in (D) and putting $ZT(x_n^-) = P$, $ZT(v_n^-) = V$, $ZT(z_n) = I$, we obtain:

$$x_n^- = x_{n-1}^- + k_1\, z_{n-1} + v_{n-1}^+ \quad \rightarrow \quad P = z^{-1} P + k_1 z^{-1} I + z^{-1} V \qquad \text{(C2)} \qquad \text{Predictor}$$

$$v_n^- = v_{n-1}^- + k_2\, z_{n-1} \quad \rightarrow \quad V = z^{-1} V + k_2 z^{-1} I \qquad \text{(D2)} \qquad \text{Predictor}$$

The transfer function of the predictor filter is then given by:

$$\boxed{G(z) = \frac{P(z)}{I(z)} = k_1 \frac{z^{-1}}{1 - z^{-1}} + k_2 \left(\frac{z^{-1}}{1 - z^{-1}} \right)^2} \quad \text{Predictor Transfer Function}$$

This is the classical filter used in the 2nd order control loops.