# Filtro di Kalman

Antonio Lanciotti, Lorenzo D'Agostino, Arment Pelivani

2019-2020


Rudolf E. Kalman

# Indice

# 1 Introduzione

Il *filtro di Kalman* è un osservatore ottimo dello stato per sistemi lineari in presenza di rumori gaussiani.

La sua versatilità ed utilità lo ha portato ad innumerevoli applicazioni quali il controllo di veicoli di ogni genere (aerospaziali, navali ...), robotica e controllo delle traiettorie, ricostruzione di segnali affetti da disturbi e molto altro.

In questa relazione vengono presentati i procedimenti matematici che permettono il raggiungimento delle equazioni che lo descrivono.

Verrà poi discussa un' applicazione del filtro realizzata in ambiente *MATLAB*.

# 2 Cenni di teoria della probabilità

Il filtro di Kalman è un algoritmo che mira alla ricostruzione dello stato interno di un sistema basandosi unicamente su una serie di misurazioni che, a causa di limiti costruttivi, sono soggette a rumore.

A causa della natura del problema, risulta necessario affrontare alcuni aspetti della teoria della probabilità, in particolare ci soffermeremo sul concetto di variabile aleatoria normale, o Gaussiana, con l'intento di fornire un modello matematico per gli errori di misura che siamo costretti ad affrontare.

## 2.1 Variabili aleatorie o casuali.

Una variabile casuale/aleatoria è una variabile che può assumere valori diversi in dipendenza da qualche fenomeno aleatorio. In particolare diremo che una variabile casuale $X$ si dice continua se esiste una funzione $f(x)$ definita su tutto $\mathbb{R}$ : $P(X \in B) = \int_B f(x)dx$ dove la funzione $f$ si dice *densità di probabilità* della variabile casuale $X$.

L'integrale di tale funzione nel dominio di integrazione $B$ rappresenta pertanto la probabilità che la variabile aleatoria assuma valori appartenenti a $B$.

Le variabili casuali risultano essere un valido strumento matematico per la modellazione dei rumori.

Alle variabili casuali sono associati i concetti di media/valore atteso, di varianza e di covarianza.

### 2.1.1 Valore atteso

Nella teoria della probabilità il valore atteso di una variabile casuale $X$, è un numero indicato con $E[X]$ che formalizza l'idea di valore medio di un fenomeno aleatorio.

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} xf(x)dx \tag{2.1}$$

Si noti che l'operatore valore atteso è lineare:

$$E[aX + bY] = aE[X] + bE[Y] \tag{2.2}$$

### 2.1.2 Varianza

La varianza di una variabile aleatoria è una funzione che fornisce una misura della variabilità dei valori assunti dalla variabile stessa; nello specifico, la misura di quanto essi si discostino dal valore atteso.

La varianza della variabile aleatoria $X$ è definita come il valore atteso del quadrato della variabile aleatoria centrata sulla propria media: $X - E[X]$ :

$$Var(X) = E[(X - E[X])^2] \tag{2.3}$$

### 2.1.3 Covarianza

In statistica e in teoria della probabilità, la covarianza di due variabili aleatorie è un numero che fornisce una misura di quanto le due varino dipendentemente l'una dall'altra.

La covarianza di due variabili aleatorie $X$ e $Y$ è il valore atteso dei prodotti delle loro distanze dalla media:

$$Cov(X, Y) = E[(X - E[X])(Y - E[Y])] \tag{2.4}$$

Due variabili casuali si dicono *incorrelate* se la loro covarianza è nulla.

La covarianza può essere considerata una generalizzazione della varianza

$$Var(X) = Cov(X, X) \tag{2.5}$$

## 2.2 Variabili gaussiane e modellazione dei rumori.

Le variabili gaussiane sono particolari variabili aleatorie caratterizzate da due parametri, $\mu$ e $\sigma^2$, e sono indicate tradizionalmente con:

$$X \sim N(\mu, \sigma^2) \tag{2.6}$$

Sono caratterizzate dalla funzione densità di probabilità:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.7}$$

Si può dimostrare che per le variabili gaussiane vale che:

$$E[X] = \mu \qquad Var[X] = \sigma^2 \tag{2.8}$$

Come anticipato possiamo modellizzare i vettori di disturbo del sistema che consideriamo, attraverso l'utilizzo di variabili aleatorie gaussiane a media nulla e varianza $\sigma^2$, di dimensioni conformi a quelle del sistema considerato.

## 2.3 Vettori casuali

Un vettore casuale è un vettore i cui elementi sono variabili casuali.

Risulta necessario estendere le definizioni date in precedenza per caratterizzare rumori che agiscono su sistemi non scalari.

### 2.3.1 Valore atteso

Si dice valore atteso del vettore casuale $x \in \mathbb{R}^n$ il vettore dei valori attesi delle variabili casuali che lo compongono:

$$E[x] = \begin{pmatrix} E[x_1] & E[x_2] & \dots & E[x_n] \end{pmatrix}^T \tag{2.9}$$

Si definisce *valore quadratico medio* di $x$ come $E[x^T x]$.

### 2.3.2 Matrice di covarianza

Si definisce *matrice di covarianza* del vettore casuale $x \in \mathbb{R}^n$ la matrice $n \times n$:

$$Cov(x, x) = E[(x - E[x])(x - E[x])^T] \tag{2.10}$$

Per come è definita, la matrice di covarianza è una matrice simmetrica semidefinita positiva i cui elementi $\sigma_{ij}^2$ sono le covarianze tra gli elementi $x_i$ e $x_j$ del vettore $x$.

A sua volta si definisce la matrice di *cross-covarianza* tra due vettori casuali $x$ e $y$, la matrice

$$Cov(x, y) = E[(x - E[x])(y - E[y])^T] \tag{2.11}$$

Due vettori $x$ e $y$ si dicono *incorrelati* se $Cov(x, y) = 0$.

# 3 Automatica

## 3.1 Sistemi dinamici a tempo discreto

Un sistema dinamico a tempo discreto è il modello matematico di un oggetto che interagisce con l'ambiente circostante attraverso canali di ingresso e di uscita rappresentati attraverso vettori $u$ e $y$ di variabili dipendenti dal tempo. Si differenziano dalla classe dei sistemi a tempo continuo dal fatto che in questo caso il tempo è rappresentato come una variabile intera $k \in \mathbb{Z}$.

Si avrà pertanto che in ogni istante di tempo $k$ il sistema modificherà le proprie uscite sulla base dei segnali in ingresso.

Il vettore $u \in \mathbb{R}^m$ rappresenta i segnali che l'oggetto riceve in ingresso dall'esterno mentre il vettore $y \in \mathbb{R}^p$ rappresenta i segnali che l'oggetto fornisce in uscita.

In generale il comportamento del sistema non dipende esclusivamente da questi due vettori, ovvero non vi è un legame diretto tra ingresso e uscita: infatti il sistema può avere uno stato interno che evolve in funzione degli ingressi e degli stati precedenti. Lo stato di un sistema è rappresentato da un vettore $x \in \mathbb{R}^n$.

Il modello del sistema è pertanto costituito da equazioni che descrivono l'evoluzione dello stato del sistema in funzione dell'ingresso, dello stato e del tempo ed esprimono la relazione d'uscita:

$$x_{k+1} = f(x_k, u_k, k) \tag{3.1a}$$

$$y_k = g(x_k, u_k, k) \tag{3.1b}$$

dove $f$ e $g$ sono opportune funzioni vettoriali.

Consideriamo una particolare classe di sistemi, quelli lineari strettamente propri, in cui $f$ e $g$ sono funzioni lineari e l'uscita non dipende direttamente dall'ingresso ma solo dallo stato. In questo caso le equazioni generali del sistema sono:

$$x_{k+1} = A_k x_k + B_k u_k \tag{3.2a}$$

$$y_k = C_k x_k \tag{3.2b}$$

dove $A, B, C$ sono matrici di coefficienti in generale variabili nel tempo. Se tali matrici sono costanti al variare di $k$ il sistema si dice *tempo invariante*.

## 3.2 Sistemi stocastici

Il modello matematico di un sistema è un'astrazione che necessariamente deve trascurare alcuni fenomeni che sarebbero troppo complessi da descrivere. Nel caso in cui gli effetti di tali fenomeni non siano trascurabili, è possibile considerarli nel modello rappresentandoli come fenomeni stocastici, ovvero come variabili aleatorie.

Tali variabili possono anche modellizzare le incertezze nella misura delle uscite del processo, ad esempio nel caso in cui esse siano affette da rumore.

Il modello del sistema tenendo conto di tali fenomeni può essere riscritto come:

$$x_{k+1} = A_k x_k + B_k u_k + W_k w_k \tag{3.3a}$$
$$y_k = C_k x_k + v_k \tag{3.3b}$$

Le ipotesi che facciamo per caratterizzare i termini $w$ e $v$ sono:

- $w$ e $v$ sono vettori casuali gaussiani a media nulla:

$$w_k \sim \mathcal{N}(0,\, Q_k), \qquad Q_k = Q_k^T > 0 \tag{3.4a}$$
$$v_k \sim \mathcal{N}(0,\, R_k), \qquad R_k = R_k^T > 0 \tag{3.4b}$$

- $w$ e $v$ sono incorrelati:

$$E[w_{k_1} v_{k_2}^T] = 0 \qquad \forall\, k_1, k_2 \geq 0 \tag{3.5}$$

- $w$ e $v$ sono bianchi:

$$E[w_{k_1} w_{k_2}^T] = 0 \qquad \forall\, k_1 \neq k_2 \tag{3.6a}$$
$$E[v_{k_1} v_{k_2}^T] = 0 \qquad \forall\, k_1 \neq k_2 \tag{3.6b}$$

- $x_0$ è un vettore casuale gaussiano con media e covarianza note:

$$x_0 \sim \mathcal{N}(\bar{x}_0,\, P_0), \qquad P_0 = P_0^T > 0 \tag{3.7}$$

- $w$ e $v$ sono incorrelati con $x_0$:

$$E[x_0 w_k^T] = 0 \qquad \forall\, k \geq 0 \tag{3.8a}$$
$$E[x_0 v_k^T] = 0 \qquad \forall\, k \geq 0 \tag{3.8b}$$

Con le ipotesi fatte è possibile passare al problema della progettazione di un osservatore che restituisca una stima dello stato interno del sistema filtrando i rumori e le incertezze sull'evoluzione dello stato e sulla misura dell'uscita.

# 4 Osservatore ottimo

Nella teoria del controllo, l'osservatore è un sistema dinamico che ha lo scopo di stimare lo stato di un altro sistema. L'osservatore è utile in quanto la conoscenza dell'evoluzione dello stato di un processo permette di risolvere problemi come la stabilizzazione e il controllo.

Nel caso di sistema lineare, l'osservatore può essere progettato come una copia del processo (del quale deve essere pertanto noto il modello) con l'aggiunta di un termine correttivo proporzionale alla differenza tra le uscite del processo e dell'osservatore.

Tale osservatore prende il nome di *Osservatore di Luenberger* ed ha la seguente espressione:

$$\hat{x}_{k+1} = A_k\hat{x}_k + B_k u_k + K_k(y_k - C\hat{x}_k) \tag{4.1}$$

Si considera il sistema lineare con disturbi di processo e misura (3.3), con stato iniziale $x(k_0) = x_0$ e con ingresso $u_k$ misurabile per ogni $k \geq k_0$.

Per valutare la precisione dell'osservatore si definisce l'errore di stima, $e_k \triangleq x_k - \hat{x}_k$, il quale è regolato da un'equazione dinamica che si può ottenere valutando tale espressione per $k = k + 1$ e sostituendo sfruttando le equazioni (3.3a) e (4.1):

$$\begin{aligned} e_{k+1} &= A_k x_k + B_k u_k + W_k w_k - [A_k\hat{x}_k + B_k u_k + K_k(y_k - C\hat{x}_k)] = \\ &= (A_k - K_k C_k)e_k + W_k w_k - K_k v_k \end{aligned} \tag{4.2}$$

Osserviamo che l'errore è a sua volta un sistema stocastico, dato che la sua espressione dipende dai termini $v_k$ e $w_k$, pertanto ne definiamo la matrice di covarianza all'istante $k + 1$:

$$P_{k+1} = E[e_{k+1}e_{k+1}^T] \tag{4.3}$$

Tale matrice rappresenta l'*errore quadratico medio* di stima all'istante $k + 1$.

Per procedere con l'analisi ricordiamo le ipotesi (3.4)-(3.8) fatte sui termini stocastici presenti nelle equazioni del sistema.

## 4.1 Ottimizzazione

L'obiettivo che ci si prefigge è quello di determinare la matrice $K_k$ tale che la stima fornita dall'osservatore sia il più attendibile possibile, ovvero ad ogni istante $k = 0, 1, ...$ si vuole determinare ricorsivamente il guadagno $K_k$ dell' osservatore in modo da minimizzare l' errore quadratico medio di stima dello stato.

Sostituendo nella (4.3) la (4.2) e sfruttando le ipotesi (3.4)-(3.8) si ottiene:

$$\begin{aligned} P_{k+1} &= E\left\{[(A_k - K_k C_k)e_k + W_k w_k - K_k v_k][(A_k - K_k C_k)e_k + W_k w_k - K_k v_k]^T\right\} = \\ &= (A_k - K_k C_k)E[e_k e_k^T](A_k - K_k C_k)^T + W_k E[w_k w_k^T]W_k^T + K_k E[v_k v_k^T]K_k^T = \\ &= (A_k - K_k C_k)P_k(A_k - K_k C_k)^T + W_k Q_k W_k^T + K_k R_k W_k^T \end{aligned} \tag{4.4}$$

Il problema si riduce quindi alla seguente ottimizzazione quadratica:

$$\begin{aligned} K_k &= \underset{K}{\operatorname{argmin}} \left\{ \underbrace{(A_k - KC_k)P_k(A_k - KC_k)^T + W_kQ_kW_k^T + KR_kW_k^T}_{P_{k+1}} \right\} \\ &= \underset{K}{\operatorname{argmin}} \left\{ K \underbrace{(R_k + C_kP_kC_k^T)}_{S_k} K^T - K \underbrace{C_kP_kA_k^T}_{V_k^T} - \underbrace{A_kP_kC_k^T}_{V_k} K^T + W_kQ_kW_k^T + A_kP_kA_k^T \right\} \\ &= \underset{K}{\operatorname{argmin}} \left\{ KS_kK^T - KV_k^T - V_kK^T + A_kP_kA_k^T + W_kQ_kW_k^T \right\} \end{aligned} \tag{4.5}$$

Essendo $S_k \triangleq R_k + C_kP_kC_k^T > R_k > 0$ la matrice $S_k$ risulta invertibile, pertanto si può scrivere:

$$\begin{aligned} K_k &= \underset{K}{\operatorname{argmin}} \left\{ (K - V_kS_k^{-1})S_k(K - V_kS_k^{-1})^T - V_kS_k^{-1}V_k^T + A_kP_kA_k^T + W_kQ_kW_k^T \right\} \\ &= V_kS_k^{-1} \\ &= A_kP_kC_k^T(R_k + C_kP_kC_k)^{-1} \end{aligned} \tag{4.6}$$

L'ultima espressione fornisce quindi il guadagno ottimo $K_k$, all' istante $k$, a cui corrisponde il minimo errore quadratico medio all'istante $k+1$ dato da:

$$\begin{aligned} P_{k+1} &= A_kP_kA_k^T - V_kS_k^{-1}V_k^T + W_kQ_kW_k^T \\ &= A_kP_kA_k^T - A_kP_kC_k^T(R_k + C_kP_kC_k^T)^{-1}C_kP_kA_k^T + W_kQ_kW_k^T \end{aligned} \tag{4.7}$$

In definitiva si ha il seguente risultato:
dato il sistema LTV (3.3) che soddisfa le ipotesi (3.4)-(3.8), l'osservatore di Luenberger che minimizza ad ogni istante $k \geq 0$ l'errore quadratico medio di stima dello stato $P_k \triangleq E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)T]$ è fornito dal seguente algoritmo ricorsivo (che procede per $k = 0, 1, ...$):

$$K_k = A_kP_kC_k^T(R_k + C_kP_kC_k)^{-1} \tag{4.8}$$

$$\hat{x}_{k+1} = A_k\hat{x}_k + B_ku_k + K_k(y_k - C\hat{x}_k) \tag{4.9}$$

$$P_{k+1} = A_kP_kA_k^T - A_kP_kC_k^T(R_k + C_kP_kC_k^T)^{-1}C_kP_kA_k^T + W_kQ_kW_k^T \tag{4.10}$$

# 5 Equazioni del filtro

Il filtro di Kalman è un'implementazione ricorsiva degli algoritmi di stima che risolve il problema della ricostruzione dello stato di un sistema lineare.

Sia $\hat{x}_k$ la stima $k$-esima dello stato $x_k$ , e sia questa stima gaussiana a error medio nullo ($E[e_k] = 0$) con covarianza $P_k = E[e_k e_k^T]$.

Desiderando costruire una stima $\hat{x}_{k+1}$, dovremo tenere conto di due sorgenti di informazione, la conoscenza del modello (mediante l'utilizzo della legge di propagazione dello stato) e la conoscenza delle misure. Distinguiamo quindi due diverse stime di $x_k$, una prima $\hat{x}_k^-$ costruita conoscendo le misure sino $y_{k-1}$, ed una seconda $\hat{x}_k$, che utilizza anche la misura $y_k$.

Basandoci sui risultati della sezione precedente, costruiamo quindi le stime:

$$\hat{x}_k^- = A_k \hat{x}_{k-1} + B_k u_k \tag{5.1}$$

$$P_k^- = A_k P_{k-1} A_k^T + B_k^w Q_k (B_k^w)^T \tag{5.2}$$

$$L_k = P_k^- C_k^T (C_k P_k^- C_k^T + R_k)^{-1} \tag{5.3}$$

$$\hat{x}_k = \hat{x}_k^- + L_k (y_k - C_k \hat{x}_k^-) \tag{5.4}$$

$$P_k = (I - L_k C_k) P_k^- \tag{5.5}$$

L'algoritmo così descritto prende il nome di *filtro di Kalman* discreto.
Il guadagno della innovazione nel filtro, $L_k$ è fondamentalmente un rapporto tra la incertezza nella stima dello stato P e la incertezza nella misura R: conseguentemente, se le misure sono molto accurate (R piccola), la nuova stima $\hat{x}_k$ sarà poco legata alla precedente. Se viceversa sono disponibili misure poco affidabili ma vecchie stime relativamente buone, si propagheranno queste nel futuro appoggiandosi sostanzialmente al modello del sistema.

**ROBA NUOVA**

Come osservato nel precedente paragrafo, l'algoritmo (4.8)-(4.10) propaga una stima $\hat{x}_k$ e la relativa covarianza $P_k$ dello stato $x_k$ basata sulle osservazioni passate $y_{0:k-1}$, ovvero effettua la predizione ad un passo dello stato. Per questo motivo, nel seguito le quantità $\hat{x}_k$ e $P_k$ in (4.8)-(4.10) verranno indicate in modo più appropriato come $\hat{x}_{k|k-1} = \hat{x}_k$, e rispettivamente, $P_{k|k-1} = P_k$.

In molte applicazioni pratiche, l'obiettivo è quello di ottenere, ad ogni istante k, una stima dello stato $x_k$, e la relativa covarianza, basata su tutte le osservazioni $y_{0:k} = y_{0:k-1} \cup y_k$, inclusa quella presente, acquisite fino all'istante k.

Il problema in oggetto, noto come filtraggio, è finalizzato alla determinazione della cosiddetta stima filtrata $\hat{x}_{k|k}$ dello stato $x_k$ basata sulle osservazioni $y_{0:k}$ e della relativa covarianza $P_{k|k} = E[\tilde{x}_{k|k}\tilde{x}_{k|k}^T]$ ,con $\tilde{x}_{k|k} \triangleq \tilde{x}_k - \hat{x}_{k|k}$.

A tale scopo, si possono considerare $\hat{x}_{k|k}$ e $P_{k|k}$ come stima e covarianza a-posteriori della variabile aleatoria $x_k$, a partire da stima e covarianza a priori $\hat{x}_{k|k-1}$ e $P_{k|k-1}$ ,sulla base dell'osservazione $y_k$ in (3.3b), utilizzando il metodo di stima BLUE[1].

In altri termini, date le statistiche a priori $(\hat{x}_{k|k-1}, P_{k|k-1})$ e l'osservazione lineare $y_k = C_k x_k + v_k$ di $x_k$ si vogliono determinare le statistiche a posteriori $(\hat{x}_{k|k}, P_{k|k})$ di $x_k$ mediante stima BLUE. Si ricorda che stima e covarianza a posteriori BLUE sono fornite da:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \underbrace{E[\tilde{x}_{k|k-1}\tilde{y}_{k|k-1}^T]E[\tilde{y}_{k|k-1}\tilde{y}_{k|k-1}^T]^{-1}}_{L_k}(y_k - \hat{y}_{k|k-1}) \tag{5.6}$$

$$P_{k|k} = P_{k|k-1} - E[\tilde{x}_{k|k-1}\tilde{y}_{k|k-1}^T]E[\tilde{y}_{k|k-1}\tilde{y}_{k|k-1}^T]^{-1}E[\tilde{x}_{k|k-1}\tilde{y}_{k|k-1}^T]^T \tag{5.7}$$

dove

$$\begin{aligned} \hat{y}_{k|k-1} &= E[y_k] = C_k\hat{x}_{k|k-1} \\ \tilde{y}_{k|k-1} &= y_k - \hat{y}_{k|k-1} = C_k\tilde{x}_{k|k-1} + v_k \\ E[\tilde{x}_{k|k-1}\tilde{y}_{k|k-1}^T] &= P_{k|k-1}C_k^T \\ E[\tilde{y}_{k|k-1}\tilde{y}_{k|k-1}^T] &= R_k + C_kP_{k|k-1}C_k^T \end{aligned} \tag{5.8}$$

Sostituendo le (5.8) in (5.6)-(5.7) si ottengono le seguenti equazioni di aggiornamento da $\hat{x}_{k|k-1}, P_{k|k-1}$ a $\hat{x}_{k|k}, P_{k|k}$:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + L_k(y_k - C_k\hat{x}_{k|k-1}) \tag{5.9}$$

$$P_{k|k} = P_{k|k-1} - P_{k|k-1}C_k^T(R_k + C_kP_{k|k-1}C_k^T)^{-1}C_kP_{k|k-1} \tag{5.10}$$

con

$$L_k = P_{k|k-1}C_k^T(R_k + C_kP_{k|k-1}C_k^T)^{-1} \tag{5.11}$$

Le equazioni (5.9)-(5.10) consentono di correggere la stima predittiva $\hat{x}_{k|k-1}$e la relativa covarianza $P_{k|k-1}$ con l'ultima osservazione $y_k$, per ottenere la stima filtrata $\hat{x}_{k|k}$ e relativa covarianza $P_{k|k}$.

[1] *Best Linear Unbiased Estimator*, per approfondimenti [2]

Per questo motivo (5.9)-(5.10) vengono dette *equazioni di correzione* della stima e della covarianza, e la matrice $L_k$ in (5.11) prende il nome di *guadagno di correzione*.

Si noti che, posto $K_k = A_k L_k$, l'equazione di aggiornamento della stima (4.9) può essere espressa come segue:

$$\hat{x}_{k+1|k} = A_k \underbrace{[\hat{x}_{k|k-1} + L_k(y_k - C_k\hat{x}_{k|k-1})]}_{\hat{x}_{k|k}} + B_k u_k \tag{5.12}$$

La precedente relazione esprime la stima predittiva ad un passo $\hat{x}_{k+1|k}$ come il risultato dell'applicazione del modello di stato (3.3a), per $w_k = 0$, alla stima filtrata $\hat{x}_{k|k}$. Analogamente, l'equazione di aggiornamento della covarianza (4.10) può essere riscritta come:

$$P_{k+1|k} = A_k \underbrace{[P_{k|k-1} - P_{k|k-1}C_k^T(R_k + C_k P_{k|k-1} C_k^T)^{-1} C_k P_{k|k-1}]}_{P_{k|k}} A_k^T + W_k Q_k W_k^T \tag{5.13}$$

Pertanto, si ottengono le seguenti equazioni di predizione da $(\hat{x}_{k|k}, P_{k|k})$ a $(\hat{x}_{k+1|k}, P_{k+1|k})$:

$$\hat{x}_{k+1|k} = A_k \hat{x}_{k|k} + B_k u_k \tag{5.14}$$

$$P_{k+1|k} = A_{k|k} P_{k|k} A_k^T + W_k Q_k W_k^T \tag{5.15}$$

Riassumendo i precedenti sviluppi, la ricorsione (4.9)-(4.10) del filtro di Kalman può essere suddivisa in due fasi diverse: la correzione (5.9)-(5.10) seguita dalla predizione (5.14)-(5.15). La forma correzione-predizione del filtro di Kalman viene riportata nella pagina seguente.

## 5.1 Filtro di Kalman nella forma correzione-predizione

**Dati:**

- matrici : $A_k, B_k, C_k, W_k, Q_k, R_k$;
- stima iniziale : $\hat{x}_{1|0}$;
- covarianza iniziale : $P_{1|0}$;
- $k = 1, 2...$

**Correzione:**

$$
\begin{aligned}
S_k &= R_k + C_k P_{k|k-1} C_k^T && \textit{covarianza dell'innovazione} \\
L_k &= P_{k|k-1} C_k^T S_k^{-1} && \textit{guadagno di correzione} \\
e_k &= y_k - C_k \hat{x}_{k|k-1} && \textit{innovazione} \\
\hat{x}_{k|k} &= \hat{x}_{k|k-1} + L_k e_k && \textit{correzione della stima} \\
P_{k|k} &= P_{k|k-1} - L_k S_k L_k^T && \textit{correzione della covarianza} \\
&= (I - L_k C_k) P_{k|k-1} (I - L_k C_k)^T + L_k R_k L_k^T
\end{aligned}
$$

**Predizione:**

$$
\begin{aligned}
\hat{x}_{k+1|k} &= A_k \hat{x}_{k|k} + b_k && \textit{predizione della stima} \\
P_{k+1|k} &= A_k P_{k|k} A_k^T + W_k Q_k W_k^T && \textit{predizione della covarianza}
\end{aligned}
$$

# 6 Documentazione software MATLAB

In questo paragrafo viene descritta l'implementazione del *filtro di Kalman* come sistema dinamico ed il task implementato dal nostro gruppo in ambiente di programmazione *MATLAB*, usando l'approccio della programmazione orientata agli oggetti.

## 6.1 `sistema.m`

*MATLAB* presenta già un'implementazione dei modelli di sistemi dinamici lineari ma si è preferito realizzarne una nuova implementazione che considerasse anche gli errori di processo e di misura in modo da rispettare le ipotesi del problema.

In particolare nel file `sistema.m` viene implementata la *classe* dei sistemi dinamici stocastici che utilizzeremo.

Per semplicità abbiamo implementato soltanto sistemi tempo invarianti, per cui tutte le matrici che definiscono il sistema sono costanti.

### 6.1.1 Proprietà

Le proprietà di cui dispongono gli oggetti di questa classe sono :

```
classdef sistema < handle
%SISTEMA
%Classe che descrive un sistema dinamico lineare tempo
    invariante

  properties (Access = protected)
    A,B,C,D,W,Q,R,x;    %A,B,C,D matrici del sistema
    % W matrice di guadagno del rumore di processo
    % Q matrice di covarianza del rumore di processo
    % R matrice di covarianza del rumore di misura
    n,m,p,q;      %n dim stato, m dim ingresso, p dim uscita, q
        dim rumore di processo
    u;       %ultimo ingresso ricevuto
  end
```

I metodi implementati sono il costruttore dell'oggetto che va ad inizializzarlo e le due equazioni di evoluzione dello stato interno e di uscita.

### 6.1.2 Costruttore

La creazione dell'oggetto `sistema` avviene tramite l'inizializzazione dei suoi parametri:

```
function obj = sistema(A,B,C,D,W,Q,R,x0)
```

Al costruttore vanno passate tutte le matrici relative al caso preso in analisi (comprese le covarianze) ed il suo stato iniziale.
Al suo interno vengono effettuati tutti i controlli necessari a verificare che i parametri rispettino le seguenti proprietà:

- $A$ deve essere quadrata (dimensione $n \times n$);
- $B$ deve avere $n$ righe (dimensione $n \times m$);
- $C$ deve avere $n$ colonne (dimensione $p \times n$);
- $D$ deve essere di dimensione $p \times m$;
- $W$ deve avere $n$ righe (dimensione $n \times q$);
- $Q$ deve essere quadrata (dimensione $q \times q$) e definita positiva;
- $R$ deve essere quadrata (dimensione $p \times p$) e definita positiva;
- $x0$ vettore di lunghezza $n$.

### 6.1.3 Evoluzione dello stato

In *MATLAB* nei metodi delle classi che utilizzano le proprietà delle stesse, risulta necessario passare come argomento l'oggetto corrente. Questo è possibile attraverso la parola chiave `obj`.

```
function update(obj, u)
  % aggiorna lo stato del sistema
  if (nargin<2)
    u = zeros(obj.m,1);  % se u viene omesso si considera nullo
  end
  obj.u = u;  % salva l'ultimo ingresso ricevuto
  xn = obj.A*obj.x+obj.B*obj.u+obj.W*mvnrnd(zeros(obj.q,1),obj.Q
      )';
  % calcola il nuovo stato x(k+1) = Ax(k) + Bu(k) + Ww(k) : w =
      rumore di processo
  obj.x = xn;                % aggiorna lo stato con quello nuovo
end
```

La funzione accetta come parametro esterno l'ingresso dato al sistema; esso può essere omesso, in tal caso viene considerato nullo.
Implementa l'equazione di stato $x_{k+1} = Ax_k + Bu_k + Ww_k$ andando ad aggiornare la variabile di stato $x$ dell'oggetto.

### 6.1.4 Lettura dell'uscita

Il metodo *leggiUscita* implementa l'equazione $y(t) = Cx(t) + Du(t) + w$ restituendo in output il valore di $y$.
Il metodo non necessita di ulteriori argomenti in ingresso:

```
function y = leggiUscita(obj) % restituisce in output l'uscita
    del sistema
```

In più è stata implementato il metodo per la lettura dello stato interno in quanto l'accesso diretto alle proprietà del sistema è, per ragioni di integrità, consentito unicamente all'oggetto stesso. La lettura dello stato del sistema non sarebbe possibile nella realtà, infatti tale metodo viene utilizzato solo per monitorare il comportamento del sistema, tali dati non verranno utilizzati direttamente.

```
function x = leggiStato(obj) % get dello stato per plot.
```

## 6.2 `filtrokalman.m`

La classe `filtrokalman` è stata implementata come un estensione della precedente classe `sistema`, infatti il *filtro di Kalman* essendo un osservatore dello stato è a sua volta un sistema dinamico.
Tale estensione si realizza attraverso il concetto di ereditarietà delle classi, infatti `kalmanfilter` eredita proprietà e metodi di `sistema` e ciò si indica attraverso il simbolo `<` :

```
classdef kalmanfilter < sistema
```

### 6.2.1 Proprietà

Oltre alle proprietà della classe `sistema` da essa ereditate, vengono introdotte la matrice di guadagno, la matrice di covarianza dello stato corretto e la predizione del prossimo stato e della relativa covarianza:

```
properties %(Access = protected)
  L;          % matrice guadagno di Kalman
  P;          % matrice di covarianza dello stato corretto
  xPr, PPr;   % predizione dello stato e relativa covarianza
end
```

### 6.2.2 Costruttore

Come in `sistema.m` la classe `filtrokalman` accetta come argomenti in ingresso le matrici relative al modello del sistema da osservare e la stima iniziale dello stato (`x0`) con la relativa covarianza (`P0`); se quest'ultima è omessa viene considerata come valore di default la matrice identità di ordine $n$:

```
function obj = kalmanfilter(A, B, C, D, Q, R, x0, P0)
```

All'interno del costruttore viene richiamato il costruttore della superclasse `sistema` al fine di inizializzare le variabili relative al modello nell'oggetto `filtrokalman`:

```
  obj@sistema(A, B, C, D, Q, R, x0);
```

Inoltre, viene verificato che stima iniziale e covarianza siano valide.

### 6.2.3 Evoluzione

Come per la superclasse corrispondente, la classe *filtrokalman* ha un metodo per il calcolo dell'evoluzione del sistema. Il metodo ereditato dalla classe `sistema` viene sovrascritto (*override*) in modo da implementare l'algoritmo ricorsivo di stima descritto nel capitolo precedente:

```
function update(obj, u, y) % stima lo stato
  obj.u=u;

  %calcolo guadagno di Kalman
  obj.L = obj.PPr*obj.C'/(obj.C*obj.PPr*obj.C'+obj.R);

  %correzione
  obj.x = obj.xPr+obj.L*(y-obj.C*obj.xPr);
  I_LC = (eye(obj.n)-obj.L*obj.C);
  obj.P = I_LC*obj.PPr*I_LC'+obj.L*obj.R*obj.L';

  %predizione
  obj.xPr = obj.A*obj.x + obj.B*u;
  obj.PPr = obj.A*obj.P*obj.A'+obj.W*obj.Q*obj.W';
end
```

I parametri d'ingresso, oltre al riferimento all'oggetto, sono rispettivamente l'ingresso e l'uscita del sistema da osservare al tempo $k$.

### 6.2.4 Lettura stima

Una volta effettuato l'aggiornamento del filtro, per ottenere il valore della stima dello stato calcolata si utilizza il metodo `leggiStima` che restituisce il vettore dello stato stimato.

```
function x = leggiStima(obj)
    x = obj.x;
end
```

Se si è interessati al filtraggio dell'uscita del sistema, si può utilizzare il metodo `leggiUscitaStimata` che restituisce l'uscita del sistema calcolata sulla base dello stato stimato.
Entrambi i metodi non necessitano di alcun parametro esterno.

```
function y = leggiUscitaStimata(obj)
  y = obj.C*obj.x + obj.D*obj.u;
end
```

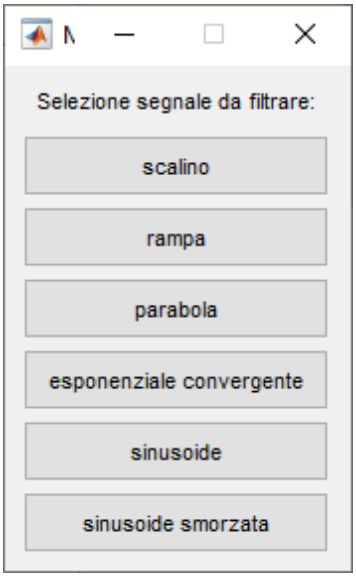
Sono stati implementati anche i metodi `leggiL` e `leggiP` che permettono di ottenere le matrici L e P del filtro all'istante corrente. Ciò sarà utile per osservarne l'evoluzione attraverso le funzioni grafiche di *MATLAB*.

```
function L = leggiL(obj)
  L = obj.L;
end
function P = leggiP(obj)
  P = obj.P;
end
```

## 6.3 Main task : `filtraggio.m`

Il task che ci siamo prefissati di raggiungere è quello di ricostruire un segnale disturbato da rumore bianco gaussiano. Questa applicazione risulta molto utile in ambito ingegneristico in quanto anche i migliori trasduttori, per limiti costruttivi, presentano delle variazioni nelle misure seppur piccole.
Oltre a questo i trasduttori migliori sono reperibili soltanto ad un costo elevato, per cui si può pensare in certe condizioni di risparmiare sulla sensoristica applicando alle misure più rumorose di un eventuale trasduttore economico il *filtro di Kalman* così da ottenere dei valori affidabili a prezzi più accessibili.

All'inizio dello script vengono definiti il tempo di campionamento e la durata della simulazione in secondi.
Viene poi visualizzato un menu che permette di scegliere la natura del segnale da filtrare; i segnali possibili sono tutti e soli quelli ottenibili come uscite da sistemi lineari.
I modelli dei generatori di segnale utilizzati vengono riportati nella sezione successiva. Cliccando uno dei segnali il programma provvederà alla creazione del modello del generatore ed alla sua successiva discretizzazione attraverso le funzioni *built-in* di *MATLAB*.



Menu di scelta dei segnali

```
sys = ss(A,B,C,D);
sysd = c2d(sys,dt);
[Ad,Bd,Cd,Dd] = ssdata(sysd);
```

Successivamente vengono definite le matrici di covarianza dei rumori di processo e misura. I valori di tali matrici possono essere variati per aumentare o diminuire la rumorosità del segnale da filtrare.

```
Q=1e-3;
R=1e-1*eye(p);
```

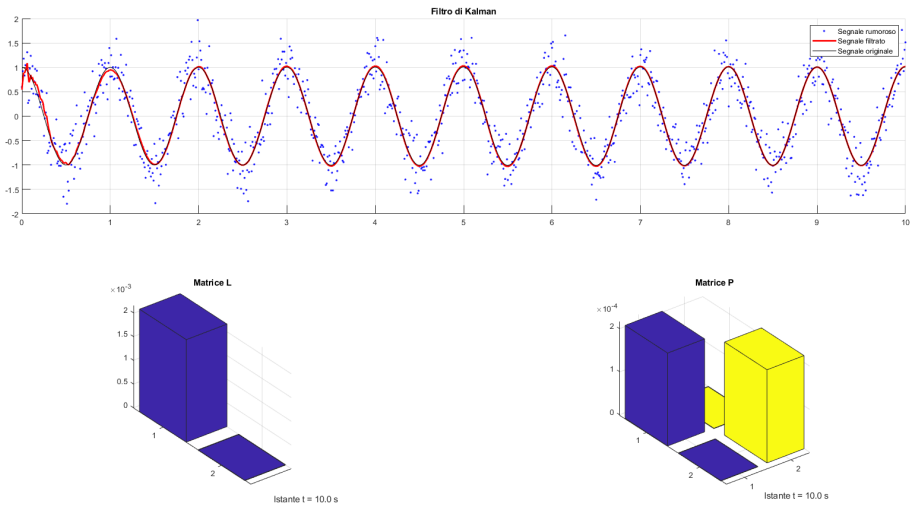
Vengono a questo punto inizializzati gli oggetti relativi al generatore di segnale e al filtro di Kalman e si definisce il ciclo che esegue la simulazione.

```
% inizializzazione sistema generatore di segnale rumoroso
sys=sistema(Ad,Bd,Cd,Dd,W,Q,R,x0);

% inizializzazione filtro di kalman
P0=eye(n);
k=filtrokalman(Ad,Bd,Cd,Dd,W,Q,R,zeros(n,1),P0);
```

```
%% simulazione
for i=1:length(t)
  x(:,i)=sys.leggiStato();      % lettura stato sistema (segnale
      da ricostruire, per plot)
  y(:,i)=sys.leggiUscita();     % lettura uscita sistema (segnale
       rumoroso)
  sys.update();                 % calcolo del nuovo stato del
      sistema
  k.update(0,y(:,i));           % aggiornamento filtro -->
      parametri u=0 e y(segnale rumoroso)
  xs(:,i)=k.leggiStima();       % lettura stima kalman
  Lplot(:,:,i)=k.leggiL();      % lettura matrice L per
      animazione
  Pplot(:,:,i)=k.leggiP();      % lettura matrice P per
      animazione
end
```

Una volta completata la simulazione, viene visualizzato il plot con il confronto tra segnale da ricostruire, campioni rumorosi e segnale ricostruito dal filtro. Nella stessa finestra viene visualizzata l'animazione delle matrici di covarianza dello stato e del guadagno.
Di seguito viene riportato un esempio di filtraggio di un segnale sinusoidale:



## 6.4 Modelli dei generatori di segnale

I generatori di segnale sono sistemi lineari autonomi ($B = 0$) ad uscita scalare ($p = 1$). La loro evoluzione a partire dallo stato iniziale $x(0)$ genera segnali diversi in base alla natura della matrice dinamica $A$.
In particolare il segnale viene prodotto nell'ultima componente dello stato $x(t)$, quindi la matrice di uscita sarà del tipo: $C = \begin{pmatrix} 0 & ... & 0 & 1 \end{pmatrix}$.
Di seguito sono riportati i modelli utilizzati nell'applicazione, che per semplicità sono scritti a tempo continuo e poi discretizzati da MATLAB.

- Segnali polinomiali di grado $n$:

$$x(t) = a_n t^n + ... + a_1 t + a_0$$

$$A = \begin{pmatrix} 0 & 1 & 0 & ... & 0 \\ 0 & 0 & 1 & ... & 0 \\ \vdots & & & \ddots & \\ 0 & 0 & ... & 0 & 1 \\ 0 & 0 & ... & 0 & 0 \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}, \qquad x(0) = \begin{pmatrix} a_0 \\ a_1 \\ ... \\ a_n \end{pmatrix}$$

  - Scalino ($n = 0$): $A = 0$
  - Rampa ($n = 1$): $A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$
  - Parabola ($n = 2$): $A = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$

- Segnale esponenziale:

$$x(t) = x_0 e^{\alpha t}$$

$$A = \alpha \in \mathbb{R}, \qquad x(0) = x_0$$

- Segnale sinusoidale ad ampiezza costante:

$$x(t) = a\cos(\omega t)$$

$$A = \begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix} \in \mathbb{R}^{2\times 2}, \qquad x(0) = \begin{pmatrix} a \\ 0 \end{pmatrix}$$

- Segnale sinusoidale smorzato:

$$x(t) = ae^{\alpha t}\cos(\omega t)$$

$$A = \begin{pmatrix} \alpha & -\omega \\ \omega & \alpha \end{pmatrix} \in \mathbb{R}^{2\times 2}, \qquad x(0) = \begin{pmatrix} a \\ 0 \end{pmatrix}$$

# Riferimenti bibliografici

[1] Luigi Chisci. Stima dello stato di un sistema dinamico. 2019. https://e-l.unifi.it/pluginfile.php/980259/mod_resource/content/1/Cap7-Stima-Stato.pdf.

[2] Luigi Chisci. Stima parametrica. 2019. https://e-l.unifi.it/pluginfile.php/980250/mod_resource/content/1/Cap5-Stima-Parametrica.pdf.