Algoritmi di stima con perdita di pacchetti in reti di sensori wireless: modellizzazione a catene di Markov, stima e stima distribuita

Chiara Brighenti, Marco Maddalena, Ivan Tessarolo

28 Marzo 2008

Indice

Introduzione

Inserito nell'ambito delle reti di sensori wireless, questo progetto affrontata due problemi distinti: da un lato, la modellizzazione e l'analisi del processo di arrivi dei pacchetti con ritardi casuali descritti da una catena di Markov, dall'altro la fusione dei dati e la gestione della perdita di pacchetto nei problemi di stima distribuita.

In letteratura esistono numerosi studi che si occupano della stima con perdita dei pacchetti ed eventuale ritardo aleatorio descritti da variabili indipendenti e identicamente distribuite, ma nel caso in cui queste siano descritte da una catena di Markov si conoscono solamente la struttura del filtro a guadagni costanti sub-ottimo nel caso di sola perdita dei pacchetti formulato da Craig Smith.

Nella prima parte di questo lavoro andremo ad estendere al caso di modello a catene di Markov alcuni algoritmi di stima già esistenti, inizialmente considerando solamente il caso di stima con perdita dei pacchetti, andando poi a considerare anche la possibiltà di arrivo con ritardo. Sulla base di questi filtri andremo poi a sviluppare degli ulteriori algoritmi di stima possibilmente più efficienti che andremo poi a valutare attraverso alcune semplici simulazioni.

Nella parte di lavoro dedicata alla stima distribuita, vengono studiati due aspetti fondamentali dei problema di stima in reti di sensori wireless: la fusione dei dati e la gestione della perdita di pacchetto.

Poichè i dispositivi impiegati hanno una fonte di energia limitata e il consumo energetico è per lo più legato alle comunicazioni, si rende necessario minimizzare il traffico in rete, sfruttando le capacità computazionali dei sensori. Nella prima sezione, vengono richiamate alcune techiche note di aggregazione e fusione dei dati, e viene proposto un metodo ottimo di fusione delle stime locali elaborate ai sensori, trascurando perdita di pacchetto e ritardi aleatori.

In effetti, il problema della gestione della perdita di pacchetto nei problemi di stima distribuita rimane in parte irrisolto, in quanto non si è ancora in grado di dire quali funzioni delle misure passate vadano implementate ai nodi in modo tale che la ricezione alla *base station* di un dato di questo tipo consenta il recupero di tutta l'informazione precedentemente persa. Nella seconda sezione, vengono dunque proposte alcune soluzioni, le quali sono delle generalizzazioni di noti algoritmi di stima distribuita al caso con perdita di pacchetto, e ne vengono analizzate le prestazioni tramite simulazione.

Questo lavoro si prefigge quindi di fornire degli strumenti che consentano un efficiente utilizzo delle reti di sensori wireless.

Capitolo 1

Stima con processo di arrivo dei pacchetti a catena di Markov

1.1 Introduzione

Prima di iniziare ad affrontare la trattazione della parte di questo lavor riguardante la stima con le catene di Markov, sarà bene fare prima una veloce panoramica sulla stima in generale con perdita di pacchetti ed eventuali ritardi.

1.1.1 Stima con perdita di pacchetti

Si consideri il seguente modello stocastico lineare tempo invariante:

$$\begin{cases} x_{k+1} & = Ax_k + w_k \\ y_k & = Cx_k + v_k \end{cases} \tag{1.1}$$

dove:

$$\begin{cases} v_k & \sim \mathcal{N}(0, R), \\ w_k & \sim \mathcal{N}(0, Q), \\ x_0 & \sim \mathcal{N}(\bar{x}_0, P_0), \end{cases}$$

con v_k, w_k e x_0 variabili aleatorie gaussiane a media nulla scorrelate tra di loro.

Poniamoci ora nella situazione descritta dallo schema di figura (Figura 1.1) in cui le misure y_k fornite dal sistema vengano trasmesse tramite una rete di comunicazione digitale fino allo stimatore, con la possibiltà che alcuni pacchetti contenenti queste misure vengano persi durante la trasmissione lungo la rete.

Dal punto di vista dello stimatore il processo di misura e trasmissione dei dati può essere così visto:

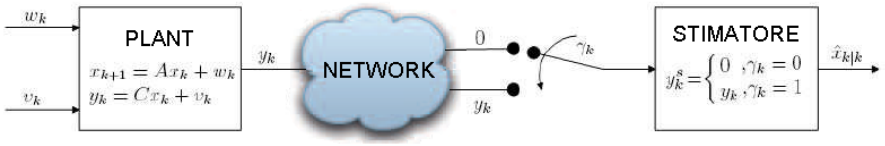

Figure 1.1: Rete di comunicazione dei dati con perdita dei pacchetti.

$$\begin{cases} x_{k+1} = Ax_k + w_k \\ y_k^S = \gamma_k C x_k + \gamma_k v_k \end{cases} \tag{1.2}$$

dove abbiamo introdotto la variabile aleatoria binaria γ_k definita come:

$$\gamma_k \triangleq \begin{cases} \gamma_k = 1 & (se\, la\, misura\, y_k\ arrivata) \\ \gamma_k = 0 & (altrimenti) \end{cases}$$

Come ampiamente affrontato in [9], in questo contesto lo stimatore ottimo dello stato che minimizza la varianza d'errore di stima

$$P_{k|k} \triangleq \mathbb{E}\left[\left(x_k - \hat{x}_{k|k}\right)\left(x_k - \hat{x}_{k|k}\right)^T | y_k^S, ..., y_0^S, \gamma_k, ..., \gamma_0, \bar{x}_0, P_0\right]$$

è dato dalle equazioni:

$$\hat{x}_{k|k} = A\hat{x}_{k-1|k-1} + \gamma_k K_k \left(y_k - CA\hat{x}_{k-1|k-1}\right) \tag{1.3}$$

$$P_{k+1|k} = AP_{k|k-1}A^T + Q - \gamma_k AP_{k|k-1}\left(C^T CP_{k|k-1}C^T + R\right)^{-1} CP_{k|k-1}A^T \tag{1.4}$$

$$= \Phi_{\gamma_k}\left(P_{k|k-1}\right)$$

$$K_k = P_{k|k-1}C^T\left(CP_{k|k-1}C^T + R\right)^{-1} \tag{1.5}$$

a partire dalle condizioni iniziali:

$$\begin{cases} \hat{x}_{-1|-1} = \bar{x}_0 \\ P_{0|-1} = P_0 \end{cases}$$

dove la stimatore ottimo $\hat{x}_{k|k}$ viene calcolato in modo ricorsivo tenendo in memoria solamente la stima e la varianza d'errore calcolate al passo precedente.

Sempre in [9], similmente a quanto visto per il filtro di Kalman a regime [4], è possibile realizzare un filtro a regime, sub-ottimo rispetto allo stimatore descritto dalle (1.3), (1.4) e (1.5), che anzichè ricalcolare ad ogni istante k un nuovo guadagno K_k, utilizzi sempre lo stesso guadagno costante K.

Nelle ipotesi che il processo di arrivo dei pacchetti, modellizzato attraverso la variabile aleatoria γ_k, possa essere descritto con una legge di Bernoulli di parametro λ (che descrive la probabilità di arrivo del pacchetto) indipendente dai rumori di modello e di misura w_k e v_k, lo stimatore sub-ottimo a guadagno K costante lo si ricava dall'equazione:

$$K = P_\infty C^T \left(CP_\infty C^T + R\right)^{-1} \tag{1.6}$$

dove P_∞ è soluzione dell'equazione algebrica di Riccati modificata (MARE):

$$P_\infty = AP_\infty A^T + Q - \lambda AP_\infty C^T \left(CP_\infty C^T + R\right)^{-1} CP_\infty A^T = \Phi_\lambda\left(P_\infty\right) \tag{1.7}$$

Sulle condizioni di esistenza ed unicità della (1.7), ci limitiamo ad affermare che sono garantite se si assume che la coppia (A, C)sia rivelabile e la coppia $\left(A, Q^{1/2}\right)$ sia stabilizzabile. Per quanto riguarda le condizioni di stabilità del filtro in presenza di A instabile, rimandiamo il lettore a [9].

Utilizzando questo stimatore sub-ottimo a guadagno costante K, l'evoluzione nel tempo dell'errore di stima $e_k \triangleq x_k - \hat{x}_k$ è dato da:

$$\begin{aligned} e_{k+1} &= x_{k+1} - \hat{x}_{k+1|k} \\ &= (Ax_k + w_k) - A\left(A\hat{x}_{k-1|k-1} + \gamma_k K\left(y_K^s - CA\hat{x}_{k-1|k-1}\right)\right) \\ &= A\left(I - \gamma_k KC\right)e_k + w_k - \gamma_k AKv_k \end{aligned} \tag{1.8}$$

da cui è possibile ricavare la varianza d'errore di stima:

$$\begin{aligned} \tilde{P}_{k+1|k} &= A\left(I - \gamma_k KC\right)\tilde{P}_{k|k-1}\left(I - \gamma_k KC\right)^T A^T + Q + \gamma_k AKRK^T A^T \\ &= \mathcal{L}_{\gamma_k}\left(K, \tilde{P}_{k|k-1}\right) \end{aligned} \tag{1.9}$$

Essendo lo stimatore a guadagno costante sub-ottimo rispetto allo stimatore dato dalle equazioni (1.3) (1.4) e (1.5), vale perciò la disuguaglianza:

$$P_{k+1|k} \leq \tilde{P}_{k+1|k} \tag{1.10}$$

che rimane valida se passiamo ai valori medi:

$$\bar{P}_{k+1|k} \triangleq \mathbb{E}\left[P_{k+1|k}\right] \leq \mathbb{E}\left[\tilde{P}_{k+1|k}\right] \triangleq \check{P}_{k+1|k} \tag{1.11}$$

1.1.2 Stima con perdita di pacchetti e ritardo

Facendo sempre riferimento al modello (1.1), consideriamo ora uno schema in cui il pacchetto contenente la misura y_k, oltre alla possibilità che venga perso durante la trasmissione lungo la rete, possa giungere allo stimatore con un certo ritardo $\tau_k \in \{0, 1, 2 \ldots N-1\} \cup \{N \sim \infty\}$, dove per $\tau_k = N \sim \infty$ si intende la perdita del pacchetto. Si assume cioè che il pacchetto arriva entro N istanti altrimenti non arriva più allo stimatore. Questo rende necessario l'utilizzo di un buffer di lunghezza pari ad N per poter usufruire di tutti i pacchetti arrivati, come rappresentato in (Figura 1.2). Chiaramente, per poter collocare ogni misura giunta allo stimatore nella posizione giusta all'interno del buffer, il pacchetto, oltre a portare l'informazione y_k, dovrà specificare anche l'istante k a cui la misura è riferita. In generale, se all'istante t arriva il pacchetto, relativo

all'istante $k \leq t$, con un ritardo $\tau_k = t - k$, allora la misura y_k verrà collocata all'interno del buffer nella posizione $t - k + 1$. Ad ogni istante t, le posizioni del buffer che non contengono misure vengono poste a 0 (è un valore fittizio, non si intende misura nulla).

Figure 1.2: Rete di comunicazione dei dati con perdita dei pacchetti e ritardo.

Se si introduce la nuova variabile aleatoria:

$$\gamma_k^t \triangleq \begin{cases} \gamma_k^t = 1 & (se\, la\, misura\, y_k\ presente\, nel\, buffer\, all'istante\, t) \\ \gamma_k^t = 0 & (altrimenti) \end{cases}$$

a cui fa seguito la variabile:

$$\tau_k \triangleq \begin{cases} \infty & (se\, \gamma_k^t = 0 \quad \forall t) \\ \bar{t} - k & (dove\, \bar{t} = \min\{t | \gamma_k^t = 1\}) \end{cases}$$

il processo di misura visto dallo stimatore può allora così essere descritto :

$$\begin{cases} x_{k+1} = Ax_k + w_k \\ y_k^t = \gamma_k^t C x_k + \gamma_k^t v_k \end{cases}$$

In queste condizioni, come descritto in [9], il filtro ottimo viene realizzato in due fasi. Nella prima fase, di inizializzazione, $\forall\, t = 0 \ldots N-1$ il filtro viene descritto dal seguente sistema di equazioni:

$$\begin{cases} \hat{x}_{k|k} = A\hat{x}_{k-1|k-1} + \gamma_k^t K_k^t \left(y_k - CA\hat{x}_{k-1|k-1}\right) \\ P_{k+1|k}^t = AP_{k|k-1}^t A^T + Q - \gamma_k^t AP_{k|k-1}^t \left(C^T C P_{k|k-1}^t C^T + R\right)^{-1} C P_{k|k-1}^t A^T \\ K_k^t = P_{k|k-1}^t C^T \left(C P_{k|k-1}^t C^T + R\right)^{-1} \qquad k = 0, \ldots, t \end{cases} \tag{1.12}$$

a partire dalle condizioni iniziali:

$$\begin{cases} \hat{x}_{-1|-1}^t = \bar{x}_0 \\ P_{0|-1}^t = P_0 \end{cases}$$

Nella seconda fase a regime, $\forall\, t \geq N$, prima si procede all'assegnazione:

$$\begin{cases} \hat{x}^t_{t-N|t-N} = \hat{x}^{t-1}_{t-N|t-N} \\ P^t_{t-N+1|t-N} = P^{t-1}_{t-N+1|t-N} \end{cases}$$

dopo di che si applica sempre la (1.11) per $k = t - N + 1, ..., t$.

Analogamente a quanto visto poco fa, se si suppone che il processo di arrivo dei pacchetti γ^t_k e τ_k sia i.i.d. con probabilità di arrivo $\mathbb{P}[\tau_k \leq h] = \lambda_h \, \forall k$, a cui segue $\mathbb{P}[\tau_k = h] = \lambda_h - \lambda_{h-1} \, \forall k$, si può pensare anche in questo caso di ricorrere ad uno stimatore a guadagni costanti. In [7] viene descritto in modo dettagliato come realizzare lo stimatore sub-ottimo ad N guadagni costanti nota la distribuzione di probabilità di λ_h per $h = 0, ..., N - 1$, vedi (Figura 1.3).

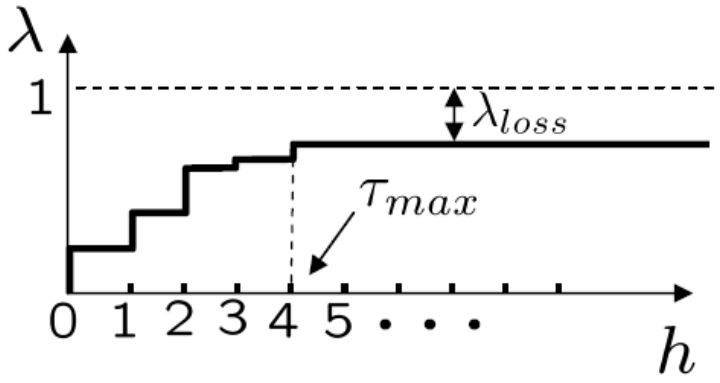

Figure 1.3: Esempio di distribuzione della probabilità di arrivo λ_h con N=5.

Gli N guadagni costanti vengono ricavati dalle equazioni:

$$\begin{cases} K_h = V_h C^T \left(C V_h C^T + R \right)^{-1} & h = 1, ..., N \\ V_N = \Phi_{\lambda_{N-1}} (V_N) & \\ V_h = \Phi_{\lambda_{h-1}} (V_{h+1}) & h = N - 1, ..., 1 \end{cases} \tag{1.13}$$

ed utilizzati nel seguente filtro:

$$\begin{cases} \hat{x}^t_{k|k} = A \hat{x}^t_{k-1|k-1} + \gamma^t_k K_{t-k+1} \left(y_k - C A \hat{x}^t_{k-1|k-1} \right) \, \forall k = t - N + 1, ..., t \\ \hat{x}^t_{t-N|t-N} = \hat{x}^{t-1}_{t-N|t-N} \end{cases} \tag{1.14}$$

dopo una prima inizializzazione per $t = 1, ..., N$.

Anche in questo caso, sulle condizioni di esistenza ed unicità della (1.13), ci limitiamo ad affermare che sono garantite se si assume che la coppia (A, C)sia rivelabile e la coppia $\left(A, Q^{1/2} \right)$ sia stabilizzabile.

1.2 Modello del processo con catena di Markov

Nell'introduzione, ogni volta in cui è stata data una descrizione di tipo probabilistico al processo di perdita dei pacchetti ci si è sempre posti nell'ipotesi che l'arrivo di ogni misura fosse indipendente dalle altre. Questa ipotesi ha permesso di facilitare notevolmente lo sviluppo di stimatori sub-ottimi molto efficienti per questi casi. Nell'applicazione pratica, questa ipotesi risulta essere troppo restrittiva, nella maggior parte dei casi infatti esiste sempre una certa correlazione tra le perdite dei pacchetti. Ad esempio, in un sistema di trasmissione con protocollo di accesso alla rete di tipo casuale, in cui è possibile il verificarsi di collisioni tra due utenti che tentino di collegarsi nello stesso momento, se da un pò di tempo non arrivano più misure allo stimatore, questo può significare che la rete sia intasata ed è quindi molto probabile che anche i prossimi pacchetti non arrivino a destinazione. Processi di perdita di questo tipo, in cui la probabilità di perdita di ogni pacchetto dipende in qualche modo dallo stato in cui si trova la rete di comunicazione possono essere descritti in modo soddisfacente attraverso dei modelli probabilistici a catena di Markov a stati finiti, ed è su questo tipo di modelli che andremo a sviluppare degli stimatori il più possibile efficienti.

Si raccomanda di fare particolare attenzione agli indici delle variabili di questo capitolo, che per maggior chiarezza di esposizione risulteranno differenti rispetto a quelli delle stesse variabili utilizzate nell'introduzione, in quanto d'ora in avanti faranno riferimento agli stati della catena di Markov del modello del processo e non più direttamente agli eventuali perdite o ritardi ad essi associati.

1.2.1 Stima con processo di perdita di pacchetti a catena di Markov

1.2.1.1 Modello a catena di Markov del processo di perdita dei pacchetti

Assumiamo ora che il processo di perdita dei pacchetti γ_k possa essere descritto da un modello probabilistico del tipo a catena di Markov a due stati, dove ad ogni istante k la permanenza del modello in uno stato o nell'altro determina l'arrivo o perdita del k-esimo pacchetto, che per maggior chiarezza di notazione associamo rispettivamente allo stato $s_k = 1$ e allo stato $s_k = 2$ (Figura 1.4).

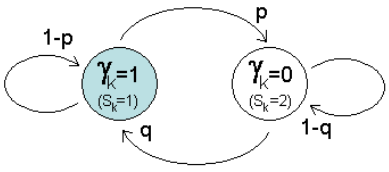

Figure 1.4: Catena di Markov del processo di arrivo γ_k.

Ad ogni istante k la probabilità di rimanere in uno dei due stati o di passare ad un altro sono determinate dai due parametri $0 < p, q < 1$. Su questi due parametri viene costruita la matrice stocastica Π che caratterizza interamente il processo di transizione da uno stato all'altro:

$$\mathbf{\Pi} = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix} \tag{1.15}$$

Definiamo inoltre il vettore riga $\nu = \begin{bmatrix} \nu_1, & \nu_2 \end{bmatrix}$ come la distribuzione di probabilità di trovarsi nello stato $s = 1$ ($\gamma_0 = 1$) o in $s = 2$ ($\gamma_0 = 0$) all'istante iniziale.

Dalla teoria delle catene di Markov, la probabilità di arrivare nello stato s_{k+1} dipende unicamente dallo stato nell'istante precedente s_k (proprietà dei processi Markoviani). Nel caso in cui si conosca solamente la distribuzione di probabilità $\mathbf{P}_k$ di s_k, quella di s_{k+1} è determinabile dalla:

$$\mathbf{P}_{k+1|k} = \left[\mathbb{P}\left(s_{k+1} = 1|\mathbf{P}_k\right), \mathbb{P}\left(s_{k+1} = 2|\mathbf{P}_k\right)\right] = \mathbf{P}_k\mathbf{\Pi}$$

(se si conosce con certezza lo stato della catena $s_k = i$ nell'istante k si ha $\mathbf{P}_k = \mathbf{e}_i^T = [0_1, 0_2, \ldots, 1_i, \ldots, 0_{N+1}]$). Iterando questa operazione n volte si può ricavare la distribuzione di probabilità della catena dopo n passi:

$$\mathbf{P}_{k+n|k} = \mathbf{P}_k\Pi^n \tag{1.16}$$

Come ben noto dal teorema di Markov per le matrici di transizione regolari (> 0) [10], se la matrice stocastia Π è regolare allora esiste un'unica distribuzione stazionaria $\pi = \begin{bmatrix} \pi_1, & \pi_2 \end{bmatrix}$ tale che:

$$\pi = \pi\mathbf{\Pi} \tag{1.17}$$

e per di più si ha che:

$$\left[\lim_{k\to\infty} \mathbf{\Pi}^k\right]_{ij} = \pi_j$$

Ne consegue che a regime:

$$\lim_{k\to\infty} \mathbb{P}(\mathrm{s_k} = \mathrm{j}|\nu) = \lim_{\mathrm{k}\to\infty} \left[\nu\mathbf{\Pi}^{\mathrm{k}}\right]_{\mathrm{j}} = \sum_{i\in S} \nu_i\pi_j = \pi_j \tag{1.18}$$

indipendentemente dalla distribuzione di probabilità ν. Dalla risoluzione dell'equazione (1.17) si possono così ricavare le probabilità a regime:

$$\pi = \begin{bmatrix} \pi_1 = \frac{q}{p+q} \\ \pi_2 = \frac{p}{p+q} \end{bmatrix} \tag{1.19}$$

D'ora in avanti andremo a considerare solamente matrici stocastiche Πregolari.

Chiaramente, anche sotto queste ipotesi di processo di perdita dei pacchetti a cataena di Markov lo stimatore ottimo a minima varianza d'errore è quello

descritto dalle equazioni (1.3) (1.4) e (1.5), in quanto ottimo indipendente dalla natura del processo di arrivo dei pacchetti. Non è più possibile utilizzare invece lo stimatore sub-ottimo a guadagno costante K descritto dalla (1.6) e (1.7). Vediamo perciò di determinarne uno analogo a guadagno costante sulla base di queste nuove ipotesi.

1.2.1.2 Stimatore sub-ottimo a guadagno costante

In modo del tutto simile a come affrontato in [9], sulla base della descrizione del processo di perdita dei pacchetti γ_k sopra descritta, iniziamo con il determinre le equazioni del filtro sub-ottimo a guadagno costante K che minimizzi la varianza d'errore di stima . Anche in queste ipotesi vale la (1.9), dove $\check{P}_{k+1|k}$ è calcolabile, sotto opportune ipotesi, in quanto funzione lineare in γ_k e costituisce un limite superiore per $\bar{P}_{k+1|k}$.

Il calcolo di $\check{P}_{k+1|k}$ può essere così svolto:

$$\begin{aligned}
\check{P}_{k+1|k} &= \mathbb{E}_{\gamma}\left[\mathcal{L}_{\gamma_k}\left(K,\tilde{P}_{k|k-1}\right)\right] \\
&= \mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[\mathbb{E}_{\gamma_k}\left[\mathcal{L}_{\gamma_k}\left(K,\tilde{P}_{k|k-1}\right)\right]\right] \\
&= \mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[\mathcal{L}_{\gamma_k=0}\left(K,\tilde{P}_{k|k-1}\right)\mathbb{P}\left(s_k=2\right)+\mathcal{L}_{\gamma_k=1}\left(K,\tilde{P}_{k|k-1}\right)\mathbb{P}\left(s_k=1\right)\right] \\
&= \mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[\mathcal{L}_{\gamma_k=0}\left(K,\tilde{P}_{k|k-1}\right)\left[(1-q)\,\mathbb{P}\left(s_{k-1}=2\right)+p\mathbb{P}\left(s_{k-1}=1\right)\right]\right]+ \\
&\quad +\mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[\mathcal{L}_{\gamma_k=1}\left(K,\tilde{P}_{k|k-1}\right)\left[q\mathbb{P}\left(s_{k-1}=2\right)+(1-p)\,\mathbb{P}\left(s_{k-1}=1\right)\right]\right]
\end{aligned}$$

per il teorema di Markov, nell'ipotesi che Π sia regolare, per $k-1\rightarrow\infty$ la probabilità $\mathbb{P}\left(s_{k-1}=i\right)=\pi_i$ indipendentemente dagli stati iniziali, possiamo perciò scrivere:

$$\begin{aligned}
\check{P}_{k+1|k} &= \mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[\mathcal{L}_{\gamma_k=0}\left(K,\tilde{P}_{k|k-1}\right)\right]\left\{(1-q)\,\pi_2+p\pi_1\right\}+ \\
&\quad +\mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[\mathcal{L}_{\gamma_k=1}\left(K,\tilde{P}_{k|k-1}\right)\right]\left\{q\pi_2+(1-p)\,\pi_1\right\} \\
&= \mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[A\tilde{P}_{k|k-1}A^T+Q\right]\left\{(1-q)\,\frac{p}{p+q}+p\frac{q}{p+q}\right\}+ \\
&\quad +\mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[A\left(I-KC\right)\tilde{P}_{k|k-1}\left(I-KC\right)^TA^T+Q+AKRK^TA^T\right]\left\{q\frac{p}{p+q}+(1-p)\frac{q}{p+q}\right\} \\
&= \left(A\mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[\tilde{P}_{k|k-1}\right]A^T+Q\right)\pi_2+ \\
&\quad +\left(A\left(I-KC\right)\mathbb{E}_{(\gamma_0\cdots\gamma_{k-1})}\left[\tilde{P}_{k|k-1}\right]\left(I-KC\right)^TA^T+Q+AKRK^TA^T\right)\pi_1 \\
&= \pi_1A\left(I-KC\right)\check{P}_{k|k-1}\left(I-KC\right)^TA^T+\left(1-\pi_1\right)A\check{P}_{k|k-1}A^T+Q+\pi_1AKRK^TA^T
\end{aligned}$$

dove $\pi_2=1-\pi_1$.

Definiamo così il nuovo operatore:

$$\mathcal{L}_{\pi}\left(K,P\right)=\pi A\left(I-KC\right)P\left(I-KC\right)^TA^T+\left(1-\pi\right)APA^T+Q+\pi AKRK^TA^T \tag{1.20}$$

Da questo punto in poi, seguendo lo stesso procedimento utilizzato in [9] limitatamente al caso in cui A sia stabile, possiamo affermare il seguente teorema:

Teorema 1 *Dato l'operatore $\Phi_\pi(S)$ definito precedentemente, si assuma che A sia stabile e $R > 0$. Allora si ha che la successione:*

$$S_{k+1} = \Phi_\pi(S_k)$$

converge all'unico punto fisso $S_\infty = \Phi_\pi(S_\infty)$ dove $S_\infty > 0$, cioè $\lim_{k\to\infty} S_k = S_\infty$.

Il guadagno costante ottimo ***a regime*** *K_∞ è dato dalla 1.6, con S_∞ soluzione della MARE $S_\infty = \Phi_\pi(S_\infty)$.*

Per quanto riguarda il caso in cui A non sia stabile, sicuramente esiste $\pi_c$tale che $\forall \pi_1 > \pi_c$ lo stimatore è stabile visto che per $\pi_1 = 1$ci riconduciamo al filtro di kalman senza perdita di pacchetti che sappiamo essere stabile mentre per $\pi_1 = 0$ ci troviamo nella condizione di assenza di misure e quindi il filtro risulta sicuramente instabile. Al momento non siamo ancora in grado di fornire il valore esatto o una stima di π_c.

Per ora possiamo fornire solamente una condizione necessaria e sufficiente per l'instabilità dello stimatore a guadagno costante.

Dalla natura del modello del processo di arrivo dei pacchetti a catena di Markov, avremo intervalli di tempo in cui arrivano le misure (chiamati normal cycle), alternati ad intervalli di tempo in cui abbiamo la perdita dei pacchetti (loss cycle). Durante questi ultimi lo stimatore ottimo di equazioni (1.3) (1.4) (1.5) è costretto ad operare la sua azione di stima in catena aperta, con conseguente aumento monotono della varianza d'errore. Al termine di ognuno di questi intervalli di perdita dei pacchetti, lo vedremo meglio più avanti, la varianza d'errore raggiunge un valore di picco $P_{k_{MAX}}$, per poi ridiscendere al successivo arrivo di nuove misure. Diamo allora la seguente definizione:

Definizione 1 *Lo stimatore ottimo è stabile in varianza d'errore di picco (peak covariance stability) se*

$$\sup_{k\geq 1} \mathbb{E}\left[\|P_{k_{MAX}}\|\right] < \infty$$

A questo punto possiamo citare il seguente teorema tratto da [11].

Teorema 2 *Nel caso di processo di arrivo dei pacchetti descritto da una catena di Markov a due stati secondo la (1.15) e si assime inoltre che la coppia secondo(A, C) sia osservabile, allora lo stimatore ottimo è stabile in varianza d'errore di picco se sono soddisfatte le seguenti due condizioni:*

1. $|\lambda_A|^2 (1-q) < 1$
2. $pqd_1^{(1)} \left[1 + \sum_{i=1}^{I_0-1} d_i^{(1)} (1-p)^i\right] \sum_{j=1}^{+\infty} \left\|A^j\right\|^2 (1-q)^{j-1} < 1$

dove:$\|.\|$ indica la norma di matrice; λ_A è l'atovalore di A di modulo maggiore; I_0 è il più piccolo indice di osservabilità tale che la matrice di osservabilità $rank\left[C^T, A^T C^T, \ldots, \left(A^{I_0-1}\right)^T C^T\right] = dim(A)$; $d_i^{(1)}$ e $d_i^{(2)}$, sono costanti positive, con $1 \leq i \leq (I_0 - 1) \vee 1$, che soddisfano la disuguaglianza:

$$\left\|F^i(P)\right\| < d_i^{(1)} \|P\| + d_i^{(2)} \qquad \forall P \in S_0^n$$

con:

$$F(P) = APA^T + Q - APC^T \left(CPC^T + R\right)^{-1} CPA^T$$

$$F^{k+1}(P) = F^k(F(P))$$

e dove $S_0^n = \left\{P : 0 \leq P \leq A\breve{P}A^T + Q, \text{ per qualche } \breve{P} \geq 0\right\}$.

La sequenza $\{P_{k_{MAX}}\}$ ci fornisce uno sviluppo superiore dell'evoluzione della varianza d'errore $P_{k+1|k}$ della (1.4). Ne consegue che lo stimatore ottimo è stabile se e solo se è stabile in varianza d'errore di picco. Riprendendo la (1.10) possiamo trarre il seguente corollario:

Corollario 1 *La condizione di stabilità in varianza d'errore di picco dello stimatore ottimo è condizione necessaria per la stabilità dello stimatore sub-ottimo a guadagno costante del teorema 2.1.*

La verifica della seconda condizione di stabilità del teorema 2.2 è piuttosto laboriosa, ma tale condizione è automaticamente soddisfatta nel caso in cui la matrice C sia invertibile, in quanto si ha che $I_0 = 1$ ed a cui segue, per un corollario in [11] che qui non riportiamo, che $d_1^{(1)} = 0$, soddisfacendo così la condizione. Si ha così che nel caso scalare la stabilità dello stimatore ottimo è garantita se:

$$A^2(1-q) < 1$$

1.2.1.3 Stimatore a due guadagni costanti

Vediamo ora di migliorare attraverso qualche piccolo accorgimento l'efficenza dello stimatore a guadagno costante appena esaminato. Questo lavoro di perfezionamento sarà basato unicamente su di una serie di idee e constatazioni puramente intuitive e non ci addentreremo troppo in dimostrazioni di tipo matematico sull'eventuale ottimalità di questo algoritmo. Ci limiteremo più avanti a valutarne l'efficienza confrontandone le prestazioni con quelle dello stimatore ottimo e quelle dello stimatore sub-ottimo a guadagno costante appena descritto.

L'unico modo per poter migliorare le prestazioni del nostro stimatore è quello di cercare di sfruttare il maggior numero possibile di informazioni a priori che siamo in grado di ricavare dal modello probabilistico del processo in esame. Nello sviluppo dello stimatore a guadagno costante, un'informazione che non abbiamo ancora sfruttato è il tempo di permanenza medio della catena in ognuno dei due stati. Cioè, se arrivo in uno stato, in media per quanto tempo rimarrò in questo stato.

Definiamo perciò $T_{s_k=i}$ il tempo di permanenza della catena di Markov nello stato $s_k = i$, $i \in [1,2]$, in un dato momento. Una volta che sono arrivato in uno dei due stati, la probabilità di rimanerci per t istanti risulta rispettivamente:

$$\mathbb{P}\left(T_{s_k=1}=t\right)=(1-p)^{t-1}\,p$$

$$\mathbb{P}\left(T_{s_k=2}=t\right)=(1-q)^{t-1}\,q$$

Grazie a queste probabilità, possiamo ora calcolare il tempo medio di permanenza nei due stati:

$$\bar{T}_{s_k=1}=\mathbb{E}\left[T_{s_k=1}\right]=\sum_{t=1}^{+\infty}t\mathbb{P}\left(T_{s_k=1}=t\right)=p\sum_{t=1}^{+\infty}t\,(1-p)^{t-1} \tag{1.21}$$

$$\bar{T}_{s_k=2}=\mathbb{E}\left[T_{s_k=2}\right]=\sum_{t=1}^{+\infty}t\mathbb{P}\left(T_{s_k=2}=t\right)=q\sum_{t=1}^{+\infty}t\,(1-q)^{t-1} \tag{1.22}$$

e si può dimostrare che le due serie convergono $\forall p,q<1$. I valori di $\bar{T}_{s_k=1}$ e $\bar{T}_{s_k=2}$ determinati tramite il calcolo delle due serie (1.19) e (1.20) in generale $\in\mathbb{R}$, mentre per i nostri scopi avremo bisogno che $\bar{T}_{s_k=1}$ e $\bar{T}_{s_k=2}\in\mathbb{N}$. Nel seguito di questo paragrafo faremo perciò sempre riferimento all'arrotondamento all'intero più vicino.

Possiamo così immaginare che "mediamente" la catena di Markov del processo rimarrà per $\bar{T}_{s_k=1}$ passi nello stato $s_k=1$ (ricordiamo che equivale a $\gamma_k=1$), dopo di che passerà nello stato $s_k=2$ ($\gamma_k=0$), per rimanerci $\bar{T}_{s_k=0}$ istanti, per poi nuovamente ritornare in $s_k=1$ e così via. Di conseguenza, come già accennato nel paragrafo precedente, l'evoluzione "media" nel tempo della varianza d'errore alternerà degli intervalli di tempo di lunghezza $\bar{T}_{s_k=i}$ in cui la varianza di stima dell'errore converge al valore di regime, ad altri in cui, a causa del mancato arrivo dei pacchetti, lo stimatore procederà in catena aperta con conseguente aumento della varianza d'errore, fino all'arrivo della prossima misura. Avremo così un'andamento del tipo come in figura (Figura 1.5), dove si può osservare a regime un alternarsi di picchi di massimo e di minimo della varianza d'errore.

I due valori di picco massimo e minimo a regime $P_{\infty MAX}$ e $P_{\infty min}$ sono determinabili dalla risoluzione del sistema di equazioni:

$$\begin{cases}P_{\infty MAX}=\Phi_{\gamma=0}\left(P_{\infty min},\bar{T}_{s_k=2}\right)\\ P_{\infty min}=\Phi_{\gamma=1}\left(P_{\infty MAX},\bar{T}_{s_k=1}\right)\end{cases}$$

dove si è utilizzato l'operatore $\Phi_\gamma\left(P,\bar{T}\right)$ definito come l'iterazione dell'operatore (1.7) per $\bar{T}$ volte a partire dal valore iniziale P. La risoluzione di questo sistema non è semplice, risulta quindi più facile determinre i valori di $P_{\infty MAX}$ e $P_{\infty min}$ tramite una iterazione sufficientemente lunga della (1.4), alternando intervalli di tempo in cui arrivano i pacchetti ($\gamma_k=1$) ad altri in cui non arrivano($\gamma_k=0$) di durata rispettivamente $\bar{T}_{s_k=1}$ e $\bar{T}_{s_k=2}$).

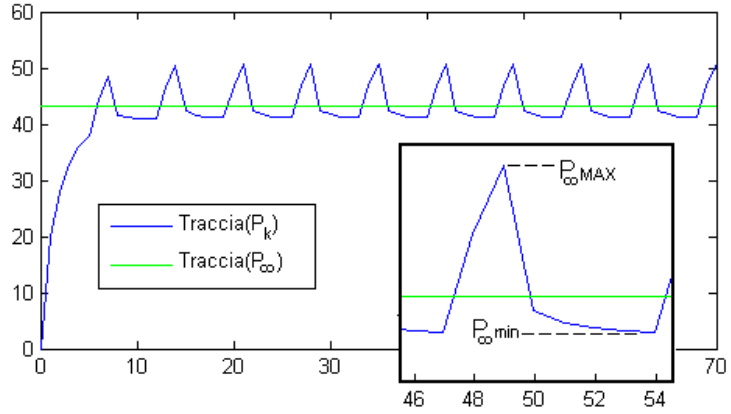

Figure 1.5: Esempio di andamento "medio" della traccia della varianza d'errore dello stimatore ottimo con $p = 0,2$ e $q = 0,6$.

A questo punto risulta chiaro che l'idea per questo nuovo stimatore consiste nell'utilizzare i due valori di picco $P_{\infty MAX}$ e $P_{\infty min}$ nella (1.6) per determinare due guadagni costanti, $K_{\infty MAX}$ e $K_{\infty min}$, anzichè uno solo. Lo stimatore funziona nel seguente modo: negli intervalli di tempo in cui non arrivano pacchetti il filtro deve ovviamente operare in catena aperta, con conseguente aumento della varianza d'errore. All'arrivo della prima misura, dopo un periodo di non arrivo dei pacchetti, la varianza d'errore è aumentata in media al valore di $P_{\infty MAX}$, utilizzeremo quindi il guadagno $K_{\infty min}$ per questa prima stima, riportando così la varianza d'errore a valori prossimi $P_{\infty min}$. A questo punto, per la stima con le restanti misure che arrivano nell'attuale successione (ininterrota) di arrivi di pacchetti viene utilizzato il guadagno $K_{\infty min}$ fino alla prossima perdita di pacchetto.

Questo algoritmo di stima alternativo, a prima vista non molto rigoroso da un punto di vista matematico, come vedremo più avanti, garantisce delle buone prestazioni, sicuramente migliori dello stimatore sub-ottimo a guadagno costante precedentemente trattato.

1.2.2 Stima con processo di perdita dei pacchetti con ritardi a catena di Markov

1.2.2.1 Modello a catena di Markov del prcesso di perdita dei pacchetti con ritardo

Allarghiamo ora il nostro modello a catena di Markov andando ad includere la possibilità che ogni pacchetto di misure, oltre a poter essere perso, possa arrivare con un certo ritardo. Allo stesso modo che abbiamo gia visto nell'introduzione, assumiamo di avere per il nostro processo di arrivo dei pacchetti un insieme finito di N possibili ritardi $\tau_k \in \{0,\, 1,\, 2 \ldots N-1\} \cup \{\infty\}$, dove per $\tau_k = \infty$ si

intende la perdita del pacchetto. Ne risulta una catena di Markov ad $N+1$ stati che per maggior chiarezza ad ogni passo k denominiamo rispettivamente $s_k \in S = \{1, 2, ..., N+1\}$ come in figura (Figura 1.6).

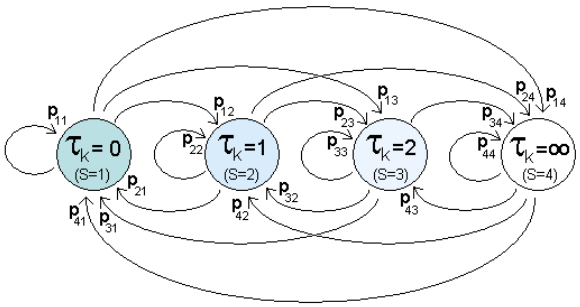

Figure 1.6: Catena di Markov del processo di arrivo dei pacchetti con ritardo per $N+1=4$.

Ad ogni istante k le probabilità di rimanere nello stato i o di passare ad un altro stato $j \neq i$, con $i, j \in S$, sono determinate dai coefficenti $0 < p_{ij} < 1$ tali che:

$$\sum_{j=1}^{N+1} p_{ij} = 1$$

Su questi parametri viene costruita la matrice stocastica $\Pi \in \mathbb{R}^{(N+1)*(N+1)}$che caratterizza interamente il processo di transizione da uno stato all'altro:

$$\mathbf{\Pi} = \begin{bmatrix} p_{1,1} & \cdots & p_{1j} & \cdots & p_{1,N+1} \\ & \ddots & & & \\ \vdots & & p_{ij} & & \vdots \\ & & & \ddots & \\ p_{N+1,1} & & \cdots & & p_{N+1,N+1} \end{bmatrix}$$

definiamo inoltre come già visto la densità di probabilità $\nu = \begin{bmatrix} \nu_1, & \ldots, \nu_{N+1} \end{bmatrix}$ che descrive la probabilità di trovarsi all'istante iniziale nello stato $i = 1, \ldots, N+1$.

Vale ancora il teorema di Markov per le matrici di transizione regolari ed esiste quindi un'unica distribuzione stazionaria $\pi = \begin{bmatrix} \pi_1, & \ldots, \pi_{N+1} \end{bmatrix}$ tale che $\pi = \pi\mathbf{\Pi}$ e per cui si ha che a regime:

$$\lim_{k \to \infty} \mathbb{P}(\tau_k = i) = \pi_i \qquad i \in S$$

indipendentemente dalle probabilità iniziali.

1.2.2.2 Stimatore sub-ottimo ad N guadagni costanti

Per quanto abbiamo visto per il filtro sub-ottimo a guadagno costante del paragrafo 2.1.2 , è logico pensare che lo stimatore sub-ottimo a regime ad N guadagni costanti si possa ottenere inserendo nella (1.12) i valori:

$$\lambda_h = \mathbb{P}(\tau_k \leq h-1) = \mathbb{P}(s_k \leq h) = \sum_{i=1}^{h} \pi_i \qquad h \in \{1, \ldots N\} \tag{1.23}$$

ed effettivamente si può dimostrare che l'intuizione è esatta. Difatti, allo stesso modo del paragrafo 2.1.2, $\check{P}^t_{k+1|k}$ risulta $\forall t \geq k \geq 1$:

$$\begin{aligned}
\check{P}^t_{k+1|k} &= \mathbb{E}_{\gamma^t}\left[\mathcal{L}_{\gamma^t_k}\left(K_{t-k}, \tilde{P}^t_{k|k-1}\right)\right] \\
&= \mathbb{E}_{(\gamma^t_0 \ldots \gamma^t_{k-1})}\left[\mathcal{L}_{\gamma^t_k=0}\left(K_{t-k}, \tilde{P}^t_{k|k-1}\right)\mathbb{P}\left(\gamma^t_k = 0\right)\right] \\
&\quad + \mathcal{L}_{\gamma^t_k=1}\left(K_{t-k}, \tilde{P}^t_{k|k-1}\right)\mathbb{P}\left(\gamma^t_k = 1\right) \\
&= \mathbb{E}_{(\gamma^t_0 \ldots \gamma^t_{k-1})}\left[\mathcal{L}_{\gamma^t_k=0}\left(K_{t-k}, \tilde{P}^t_{k|k-1}\right)\mathbb{P}\left(\tau_k > t-k\right)\right] \\
&\quad + \mathcal{L}_{\gamma^t_k=1}\left(K_{t-k}, \tilde{P}^t_{k|k-1}\right)\mathbb{P}\left(\tau_k \leq t-k\right)
\end{aligned}$$

nella quale a regime la probabilità $\mathbb{P}(\tau_k \leq t-k) = \mathbb{P}(s_k \leq t-k+1) = \lambda_{t-k+1}$ e $\mathbb{P}(\tau_k > t-k) = \mathbb{P}(s_k > t-k+1) = 1-\lambda_{t-k+1}$, indipendentemente dalle condizioni iniziali, possiamo perciò scrivere:

$$\begin{aligned}
\check{P}^t_{k+1|k} &= \lambda_{t-k+1} A\left(I - K_{t-k+1}C\right)\check{P}^t_{k|k-1}\left(I - K_{t-k+1}C\right)^T A^T + \\
&\quad + \left(1-\lambda_{t-k+1}\right) A\check{P}^t_{k|k-1}A^T + Q + \lambda_{t-k+1} A K_{t-k+1} R K^T_{t-k+1} A^T
\end{aligned}$$

Da questo punto in poi, seguendo il ragionamento descritto in [7] per A stabile, si può dimostrare il seguente teorema:

Teorema 3 *Dati gli operatori $\Phi_\pi(S)$ e $\mathcal{L}_\pi(K,P)$ definiti precedentemente. Si assuma inoltre A sia stabile e $R > 0$. Allora gli N guadagni costanti ottimi* ***a regime*** *$\{K_i\}_{i=1}^N$ sono dati dalla 1.12, con V_N soluzione della MARE $V_N = \Phi_{\lambda_N}(V_N)$.*

1.2.2.3 Stimatore ad N^2 guadagni costanti

Come già detto all'inizio del paragrafo 2.1.3, per migliorare le prestazioni del filtro sub-ottimo a N guadagni costanti dobbiamo sfruttare ulteriori informazioni che possiamo ricavare dal modello probabilistico del processo di ritardo di arrivo dei pacchetti. In questo caso il modello ci consente di ricavare l'evoluzione media della catena nei prossimi $N-1$ passi dato lo stato attuale.

Innanzitutto introduciamo un piccolo accorgimento nello stimatore. Consideriamo di abbinare ad ogni misura y_k giunta nel buffer il relativo tempo di

ritardo $\tau_k$con cui vi è arrivata. Questo ritardo, se registrato nel buffer, identifica con certezza lo stato s_k in cui si trovava la catena di Markov all'istante k. Si può allora affermare che ad ogni istante t dalla lettura dell'ultimo slot (in posizione N) del buffer, che dovrebbe contenere la misura y_{t-N+1}, possiamo determinare sempre con certezza lo stato della catena nell'istante $t-N+1$, in quanto che se la misura non è presente significa che la catena si trovava nello stato $s_{t-N+1} = N+1$ ($\tau_k = \infty$, cioè pacchetto perso).

Applicando la (1.16) possiamo perciò determinare ad ogni istante $t \geq k < t-N+1$ le distribuzioni di probabilità:

$$\mathbb{P}\left(s_k | s_{t-N+1} = i\right) = \bar{\mathbf{e}}_i^T \Pi^{k-(t-N+1)} \qquad i \in S$$

probabilità che risultano essere ben più attendibili delle statistiche fornite dal solo teorema di Markov (1.18). Da queste è possibile ricavare:

$$\begin{aligned} \lambda_{h|s_{t-N+1}} &= \mathbb{P}\left(s_{t-h+1} \leq h | s_{t-N+1} = i\right) \\ &= \sum_{j=1}^{h} \left[\bar{\mathbf{e}}_i^T \Pi^{N-h}\right]_j \qquad h \in \{1, \ldots, N\} \end{aligned}$$

che in pratica sono le probabiltà di arrivo in questo istante t di ognuna delle misure relative agli istanti tra $t-N+1$ e t.

L'idea perciò consiste, dopo aver determinato come nella (1.12) la soluzione della MARE:

$$V_N = \Phi_{\lambda_N}\left(V_N\right)$$

dove λ_N è sempre data dalla (1.23), di determinare $\forall s_{t-N+1} \in S$ noto:

$$\begin{cases} K_{i|s_{t-N+1}} = V_{h|s_{t-N+1}} C^T \left(C V_{i|s_{t-N+1}} C^T + R\right)^{-1} & i = 1, ..., N \\ V_{i|s_{t-N+1}} = \Phi_{\lambda_{h|s_{t-N+1}}}\left(V_{i+1|s_{t-N+1}}\right) & i = 1, ..., N-1 \\ V_{N|s_{t-N+1}} = V_N \qquad \forall s_{t-N+1} \in S \end{cases}$$

si ottengono così N insiemi di N guadagni costanti $\left\{K_{i|s_{t-N+1}}\right\}_{i=1}^{N}$(dove a dire il vero $K_{N|s_{t-N+1}} = K_N \, \forall s_{t-N+1} \in S$) da utilizzare come in (1.14) nel filtro:

$$\begin{cases} \hat{x}_{k|k}^t = A\hat{x}_{k-1|k-1}^t + \gamma_k^t K_{t-k+1|s_{t-N+1}} \left(y_k - CA\hat{x}_{k-1|k-1}^t\right) & \forall k = t-N+1, ..., t \\ \hat{x}_{t-N|t-N}^t = \hat{x}_{t-N|t-N}^{t-1} \end{cases} \tag{1.24}$$

dopo una prima inizializzazione per $t = 1, ..., N$.

1.3 Approcio Discrete-Time Markov Jump Linear Systems

Un differente approcio alla stima dello stato, dove il processo di arrivo dei pacchetti è regolato da un modello a catena di Markov come sopra , può essere dedotto dall'applicazione delle teorie dei Discrete-Time Markov Jump Linear Systems (DTMJLS) [12].

1.3.1 Stima con processo di perdita di pacchetti tramite DTMJLS

1.3.1.1 Modello del processo di perdita dei pacchetti descritto con i DTJMLS

Nelle medesime ipotesi sulla modellizzazione del processo di perdita dei pacchetti γ_k finora considerate, il modello 1.1 secondo la teoria dei DTMJLS può essere così riscritto:

$$\begin{cases} x_{k+1} &= A_{s_k} x_k + G_{s_k} n_k \\ y_k &= L_{s_k} x_k + H_{s_k} n_k \end{cases} \tag{1.25}$$

dove $n_k \sim \mathcal{N}\left(0, I_{[dim(R)+dim(Q)]}\right)$, in modo tale da avere che:

$$\begin{bmatrix} w_k \\ v_k \end{bmatrix} = \begin{bmatrix} Q^{1/2} & 0 \\ 0 & R^{1/2} \end{bmatrix} n_k$$

mentre le matrici del sistema dipendono da s_k nel seguente modo:

$$\begin{aligned} A_{s_k} &= (A_{s_k=1} = A,\, A_{s_k=2} = A) \\ G_{s_k} &= \left(G_{s_k=1} = \begin{bmatrix} Q^{1/2} & 0 \end{bmatrix},\, G_{\gamma_k=2} = \begin{bmatrix} Q^{1/2} & 0 \end{bmatrix}\right) \\ L_{s_k} &= (L_{s_k=1} = C,\, L_{s_k=2} = 0) \\ H_{s_k} &= \left(H_{s_k=1} = \begin{bmatrix} 0 & R^{1/2} \end{bmatrix},\, H_{s_k=2} = 0\right) \end{aligned}$$

Diamo ora alcune definizioni che ci serviranno in questo capitolo.

Definizione 2 *Il modello lineare 1.25 è stabile in media quadratica (mean square stable, MSS) per ogni condizione iniziale se esistono μ e $\mathbb{Q}$, indipendenti da x_0 e s_0, tali che:*

1 $\|\mathbb{E}[x_k] - \mu\| \to 0 \qquad per\, k \to \infty$

2 $\left\|\mathbb{E}\left[x_k x_k^T\right] - \mathbb{Q}\right\| \to 0 \qquad per\, k \to \infty$

La definizione di stabilità in media quadratica è l'equivalente della stabilità per i sistemi lineari deterministici. Per determinare se il modello 1.25 è stabile in media quadratica si ricorre al seguente teorema:

Teorema 4 *Il modello 1.25 è MSS e si ha che:*

$$\sum_{k=1}^{+\infty} \mathbb{E}\left[\|x_k\|^2\right] < \infty$$

se e solo se:

$$r_\sigma\left(\mathcal{CN}\right) < 1$$

dove le matrici $\mathcal{C}$ *ed* $\mathcal{N}$ *sono date da:*

$$\mathcal{C} \triangleq \Pi^T \bigotimes I_{n^2} \quad \in \mathbb{R}^{Nn^2}$$

$$\mathcal{N} \triangleq diag\left[A_{s_k} \bigotimes A_{s_k}\right] \quad \in \mathbb{R}^{2n^2}$$

mentre $\bigotimes$ *rappresenta il prodotto di Kronecker,* $n = \dim(A_{s_k})$ *e:*

$$diag\left[A_{s_k}\right] = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & A \end{bmatrix}$$

1.3.1.2 Stimatore sub-ottimo a guadagno costante a minima media quadratica della norma d'errore di stima

Anzichè minimizzare la varianza d'errore di stima, il seguente teorema, trattato in [12] in un ambito più generale per catene di Markov ad $N+1$stati, in questo caso $N+1=2$, ci fornisce uno stimatore che invece di minimizzare la varianza dell'errore di stima va ad ottimizzare la media quadratica della norma dell'errore di stima:

Teorema 5 *Lo stimatore ottimo a guadagno costante per il sistema 1.25, che nell'intervallo temporale finito* $[0,T]$ *minimizza la funzione di costo:*

$$\sum_{k=1}^{T} \mathbb{E}\left(\left\|x_k - \hat{x}_{k|k-1}\right\|^2\right)$$

mentre a regime consente di minimizzare:

$$\lim_{k\to\infty} \mathbb{E}\left[\left\|x_k - \hat{x}_{k|k-1}\right\|^2\right]$$

è realizzato dal filtro:

$$\begin{cases} \hat{x}_{k+1|k} & = \left(A_{s_k} + M_{s_k} L_{s_k}\right) \hat{x}_{k|k-1} + M_{s_k} y_k \\ \hat{x}_0 & = \bar{x}_0 \end{cases} \tag{1.26}$$

in cui i guadagni costanti $\{M_i\}_{i=1}^{N+1}$ *sono dati dalla:*

$$M_i = -A_i Y_i L_i^T \left(H_i H_i \pi_i + L_i Y_i L_i^T\right)^{-1} \qquad i = 1, \ldots, N+1 \tag{1.27}$$

dove $Y = \{Y_i\}_{i=1}^{N+1}$ *è soluzione delle equazioni algebriche di Riccati accoppiate (Filtering CARE):*

$$Y_j = \sum_{i=1}^{N+1} \Pi_{ij} \left[A Y_i A^T + \pi_i G_i G_i^T - A Y_i L_i^T \left(H_i H_i^T \pi_i + L_i Y_i L_i^T\right)^{-1} L_i Y_i A_i^T\right] \tag{1.28}$$

Condizioni sufficienti all'esistenza della soluzione Y *della Filtering CARE sono che il sistema 1.25 sia MSS.*

Le equazioni 1.27 e 1.29 che determinano la realizzazione dello stimatore ottimo sono il frutto della diretta applicazione del più generale teorema 5.5 in [12] nel caso in cui le matrici del sistema siano tempoinvarianti e che il processo a catena di Markov s_k sia ergodico.
Le condizioni di esistenza della soluzione Y sono dettate dal teorema A.15 ed in particolare dal suo corollario A.16 in [12] nel caso particolare in cui il sistema è privo di ingresso tramite cui stabilizzarlo.

La condizione che il sistema 1.25 sotto stima sia MSS è molto restrittiva ma non preclude a questo stimatore un'ampia gamma di applicazioni. Un notevole esempio è la stima dello stato per la stabilizzazione di un sistema tramite retroazione.

Si osservi innanzitutto che nel nostro caso, per $N+1 = 2$, la Filtering CARE (1.28) diviene:

$$\begin{cases} Y_1 = (1-p)\left[A Y_1 A^T + \pi_1 Q - A Y_1 C^T \left(R + C Y_1 C^T\right)^{-1} C Y_1 A^T\right] + q\left[A Y_2 A^T + \pi_2 Q\right] \\ Y_2 = p\left[A Y_1 A^T + \pi_1 Q - A Y_1 C^T \left(R + C Y_1 C^T\right)^{-1} C Y_1 A^T\right] + (1-q)\left[A Y_2 A^T + \pi_2 Q\right] \end{cases} \tag{1.29}$$

In generale l'equazione (1.28) può essere risolta tramite un'iterazione sufficientemente lunga delle seguenti equazioni alle differenze di Riccati accoppiate:

$$Y_j(k+1) = \sum_{i=1}^{N+1} \Pi_{ij} \left[A Y_i(k) A^T + \pi_i G_i G_i^T - A Y_i(k) L_i^T \left(H_i H_i^T \pi_i + L_i Y_i(k) L_i^T\right)^{-1} L_i Y_i(k) A_i^T\right]$$

a partire dalle condizioni iniziali $Y_i(0) = \pi_i \left(P_0 - x_0 x_0^T\right)$. Condizioni sufficienti alla convergenza di $Y_i(k)$ a Y_i.

Per quanto riguarda la (1.27) si può constatare che per $i = 2$ ($\gamma_k = 0$) si ha che:

$$\begin{aligned} M_2 &= -A_2 Y_2 L_2^T \left(H_2 H_2 \pi_2 + L_2 Y_2 L_2^T\right)^{-1} \\ &= 0 \end{aligned}$$

per cui è necessario memorizzare il solo guadagno costante M_1.
Inoltre dalla 1.25 e 1.26 possiamo ricavare che:

$$\begin{aligned} e_{k+1} &= x_{k+1} - \hat{x}_{k+1|k} \\ &= (A_{s_k} x_k + G_{s_k} n_k) - \left((A_{s_k} + M_{s_k} L_{s_k})\hat{x}_{k|k-1} + M_{s_k} y_k\right) \\ &= A_{s_k} x_k + G_{s_k} n_k - (A_{s_k} + M_{s_k} L_{s_k})\,\hat{x}_{k|k-1} - M_{\gamma_k}\,(L_{s_k} x_k + H_{s_k} n_k) \\ &= (A_{s_k} + M_{s_k} L_{s_k})\,e_k + (G_{s_k} + M_{s_k} H_{s_k})\,n_k \\ &= \begin{cases} (A + M_1 C)\,e_k + \left(\begin{bmatrix} Q^{1/2} & 0 \end{bmatrix} + M_1 \begin{bmatrix} 0 & R^{1/2} \end{bmatrix}\right) n_k & se\ s_k = 1 \\ Ae_k + \begin{bmatrix} Q^{1/2} & 0 \end{bmatrix} n_k & se\ s_k = 2 \end{cases} \end{aligned}$$

se confrontata con la 1.7 si nota immediatamente l'equivalenza tra le due equazioni se si pone $M_1 = -AK$. Si ricava così che :

$$\begin{aligned} K &= -A^{\dagger} M_1 \\ &= A^{\dagger} A_1 Y_1 L_1^T (H_1 H_1 \pi_1 + L_1 Y_1 L_1^T)^{-1} \\ &= Y_1 C^T \left(R\pi_1 + C Y_1 C^T\right)^{-1} \end{aligned} \tag{1.30}$$

dove si è utilizzato la matrice pseudoinversa $A^{\dagger}$, non avendo posto vincoli sull'invertibilità di A. Confrontando la 1.30 con la 1.5 verrebbe da pensare all'esistenza di una corrispondenza tra la varianza d'errore media a regime P e la soluzione della filtering CARE 1.29 Y_1. Vediamo ora che questa corispondenza effetivamente c'è. Definiamo:

$$\tilde{P}_{k|s_k=i} = \mathbb{E}\left[e_k e_k^T * \mathbf{1}_{\{s_k=i\}}\right] \qquad j = 1, 2$$

ne segue che:

$$\begin{aligned} \tilde{P}_{k+1|j} &= \mathbb{E}\left[e_{k+1} e_{k+1}^T * \mathbf{1}_{\{s_{k+1}=j\}}\right] \\ &= \mathbb{E}\left[\left[\left(A_{s_k} + M_{s_k} L_{s_k}\right) e_k + \left(G_{s_k} + M_{s_k} H_{s_k}\right) n_k\right] \left[\left(A_{s_k} + M_{s_k} L_{s_k}\right) e_k + \left(G_{s_k} + M_{s_k} H_{s_k}\right) n_k\right]^T * \mathbf{1}_{\{s_k=j\}}\right] \\ &= \sum_{i=0}^{1} \Pi_{ij} \left[(A_i + M_i L_i)\,\mathbb{E}\left[e_k e_k^T * \mathbf{1}_{\{s_k=i\}}\right] (A_i + M_i L_i)^T + \pi_i\,(G_i + M_i H_i)\,\mathbb{E}\left[n_k n_k^T * \mathbf{1}_{\{s_k=i\}}\right] (G_i + M_i H_i)^T\right] \\ &= \begin{cases} (1-p)\left[A\tilde{P}_{k|j=1} A^T + \pi_1 Q - A\tilde{P}_{k|j=1} C^T \left(R + C\tilde{P}_{k|j=1} C^T\right)^{-1} C\tilde{P}_{k|j=1} A^T\right] + q\left[A\tilde{P}_{k|j=0} A^T + \pi_2 Q\right]; & se\ j = 1 \\ p\left[A\tilde{P}_{k|j=1} A^T + \pi_1 Q - A\tilde{P}_{k|j=1} C^T \left(R + C\tilde{P}_{k|j=1} C^T\right)^{-1} C\tilde{P}_{k|j=1} A^T\right] + (1-q)\left[A\tilde{P}_{k|j=0} A^T + \pi_2 Q\right]; & se\ j = 2 \end{cases} \end{aligned}$$

per un teorema in [12] che qui non enunciamo, si ha che $\lim_{k\to\infty} \tilde{P}_{k|j} = \tilde{P}_j$, dove per l'appunto $\tilde{P}_j$ è soluzione della:

$$\begin{cases} \tilde{P}_1 &= (1-p)\left[A\tilde{P}_1 A^T + \pi_1 Q - A\tilde{P}_1 C^T \left(R + C\tilde{P}_1 C^T\right)^{-1} C\tilde{P}_1 A^T\right] + q\left[A\tilde{P}_2 A^T + \pi_2 Q\right]; \\ \tilde{P}_2 &= p\left[A\tilde{P}_1 A^T + \pi_1 Q - A\tilde{P}_1 C^T \left(R + C\tilde{P}_1 C^T\right)^{-1} C\tilde{P}_1 A^T\right] + (1-q)\left[A\tilde{P}_2 A^T + \pi_2 Q\right]; \end{cases} \tag{1.31}$$

si nota subito che la 1.31 è uguale alla 1.29 se si pone $\tilde{P}_i = Y_i$.

E' infine importante aquesto punto notare che andando a minimizzare la funzione di costo a regime:

$$\begin{aligned}\lim_{k\to\infty}\mathbb{E}\left[\left\|x_k-\hat{x}_{k|k-1}\right\|^2\right] &= \lim_{k\to\infty}\mathbb{E}\left[e_k^T e_k\right]\\ &= \lim_{k\to\infty}\sum_{j=1}^{2} tr\left(\tilde{P}_{k|s_k=j}\right)\\ &= \lim_{k\to\infty} tr\left(P_{k|k-1}\right)\end{aligned}$$

si va a minimizzare la traccia della varianza d'errore a regime.

1.3.2 Stima con processo di perdita dei pacchetti con ritardi tramite DTMJLS

1.3.2.1 Modello del processo di perdita dei pacchetti con ritardi descritto con i DTJMLS

Prima di tutto, partendo dalla catena di Markov ad $N+1$ stati s_t che descrive il processo di perditadei pacchetti con ritardi finora considerata (Figura 1.6), cerchiamo di ricavare un nuovo modello a catena di Markov più adatto a descrivere lo stato del buffer. Vogliamo cioè una catena che sia in grado di descrivere ad ogni istante t quali degli N slot del buffer contengono delle misure e quali invece sono ancora vuoti. Se consideriamo che il buffer viene interessato dall'arrivo delle sole misure relative agli istanti da $t-N+1$ a t possiamo allora dire che il miglior modello ricavabile è una catena di Markov ad $(N+1)^2$ stati, dove ogni possibile stato $\tilde{s}_t \in \{(s_{t-N+1},\ldots,s_t),\ s_i 1,\ldots,N+1\}$ rappresenta una stringa di lunghezza N di possibili evoluzioni degli stati s_t della catena di Markov di partenza nell'intervallo di tempo da $t-N+1$ a t. Con questo nuovo modello siamo così in grado di descrivere dettagliatamente l'arrivo di ogni misura relativa all'intervallo di tempo da $t-N+1$ a t nel buffer e quindi dire se questa vi è presente o meno nell'istante t.

Ad esempio, nel caso di avere che $N+1=3$: $\tau_t \in \{0,1\}\cup\{\infty\} \Rightarrow s_t \in \{1,2,3\}$(Figura 1.7), a partire dalla relativa matrice stocastica Π si può ricavare molto facilmente la matrice stocastica $\tilde{\Pi}_{[t-N+1,t]}$del nuovo processo di evoluzione dello stato s_t da $t-N+1$ a t:

$$\Pi=\begin{bmatrix} p_{11} & p_{12} & p_{13}\\ p_{21} & p_{22} & p_{23}\\ p_{31} & p_{32} & p_{33}\end{bmatrix} \Rightarrow \tilde{\Pi}_{[t-N+1,t]}=\begin{bmatrix} p_{11} & p_{12} & p_{13} & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & p_{21} & p_{22} & p_{23} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & p_{31} & p_{32} & p_{33}\\ p_{11} & p_{12} & p_{13} & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & p_{21} & p_{22} & p_{23} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & p_{31} & p_{32} & p_{33}\\ p_{11} & p_{12} & p_{13} & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & p_{21} & p_{22} & p_{23} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & p_{31} & p_{32} & p_{33}\end{bmatrix}$$

1.3.2.2 Optimal linear minimum mean sqare filter (LMMSE)

Per poter applicare un qualche algoritmo di stima dobbiamo però prima estendere il modello (1.25) nel seguente modo:

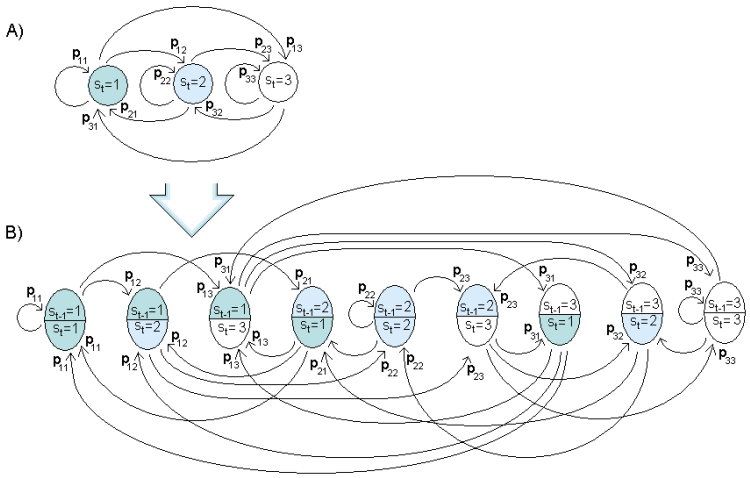

Figure 1.7: Esempio di catena che descrive il processo di evoluzione nell'intervallo $[t-N+1, t]$ di una catena di Markov s_t ad $N+1=3$ stati.

$$\begin{cases} \tilde{x}_{t+1} & = \tilde{A}_{\tilde{s}_t}\tilde{x}_t + \tilde{G}_{\tilde{s}_t}n_t \\ \tilde{y}_t & = \tilde{L}_{\tilde{s}_t}\tilde{x}_t + \tilde{H}_{\tilde{s}_t}n_t \end{cases}$$

nel quale vengono utilizzati i vettori estesi così definiti:

$$\tilde{x}_t \triangleq \begin{bmatrix} x_t \\ \vdots \\ x_{t-N+1} \end{bmatrix} \in \mathbb{R}^{nN}$$

$$\tilde{y}_t \triangleq \begin{bmatrix} y_t \\ \vdots \\ y_{t-N+1} \end{bmatrix} \in \mathbb{R}^{pN}$$

così che $\tilde{x}_t$ contiene l'evoluzione da $t-N+1$ a t dello stato del sistema (1.1) mentre $\tilde{y}_k$ rappresenta praticamente le N misure presenti nel buffer nell'istante t. Di conseguenza le relative matrici estese vengono così definite:

$$
\begin{aligned}
\tilde{A}_{\tilde{s}_t} &\triangleq \begin{bmatrix} A & 0 & \dots & 0 \\ I & 0 & \dots & 0 \\ 0 & \ddots & 0 & \vdots \\ \ddots & 0 & I & 0 \end{bmatrix} \\
\tilde{G}_{\tilde{s}_t} &\triangleq \begin{bmatrix} G_{s_t} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \\
\tilde{L}_{\tilde{s}_t} &\triangleq \begin{bmatrix} L_{s_t} & \dots & L_{s_{t-N+1}} \end{bmatrix} \\
\tilde{H}_{\tilde{s}_t} &\triangleq \begin{bmatrix} H_{s_t} \\ \vdots \\ H_{s_{t-N+1}} \end{bmatrix}
\end{aligned}
$$

Dall'applicazione del teorema 5 si ottiene uno stimatore del vettore di stato esteso $\tilde{x}_t$, le cui prime n componenti sono la stima di x_t.

Va però considerato dal punto di vista dello stimatore non possiamo sapere con certezza ad ogni istante t in quale stato $\tilde{s}_t$ del processo di evoluzione degli stati s_t si trovi la catena. Non possiamo perciò ricorrere al teorema 5, che prevede di conoscere esattamente ad ogni istante t lo stato in cui si trova la catena. Utilizzeremo allora il seguente teorema, sempre tratto da [12], che prevede il caso in cui lo stato della catena di Markov non sia noto e che qui riassumiamo :

Teorema 6 *Sia il sistema (1.25) MSS, con processo di arrivo con ritardo dei pacchetti descritto da una catena di Markov ad $N+1$ stati per la quale ad ogni istante k lo stato della catena s_k non sia noto. Definite le seguenti matrici:*

$$
\begin{aligned}
\mathsf{A} &\triangleq \begin{bmatrix} \Pi_{11}A_1 & \dots & \Pi_{1(N+1)}A_{N+1} \\ \vdots & \dots & \vdots \\ \Pi_{(N+1)1}A_1 & \dots & \Pi_{(N+1)(N+1)}A_{N+1} \end{bmatrix} \in \mathbb{R}^{(N+1)n\times(N+1)n} \\
\mathsf{H} &\triangleq \begin{bmatrix} H_1\pi_1^{1/2} & \dots & H_{N+1}\pi_{N+1}^{1/2} \end{bmatrix} \in \mathbb{R}^{(N+1)r\times p} \\
\mathsf{L} &\triangleq \begin{bmatrix} L_1 & \dots & L_{N+1} \end{bmatrix} \in \mathbb{R}^{(N+1)n\times p} \\
\mathsf{G} &\triangleq diag\left[\begin{bmatrix} \Pi_{1j}\pi_1^{1/2} & \dots & \Pi_{(N+1)j}\pi_{N+1}^{1/2} \end{bmatrix}\right] \in \mathbb{R}^{(N+1)^2r\times(N+1)n}
\end{aligned}
$$

lo stimatore sub-ottimo a guadagni costanti di x_k è dato da:

$$\hat{x}_{k|k} = \sum_{i=1}^{N+1} \hat{z}_i\,(k|k)$$

dove $\hat{z}(k|k)$ *è data dall'equazione ricorsiva:*

$$\begin{cases} \hat{z}(k|k) = \hat{z}(k|k-1) + Z\mathsf{L}^T\left(\mathsf{L}Z\mathsf{L}^T + \mathsf{H}\mathsf{H}^T\right)^{-1}(y_k - \mathsf{L}\hat{z}(k|k-1)) \\ \hat{z}(k|k-1) = \mathsf{A}\hat{z}(k-1|k-1) \\ \hat{z}(0|-1) = \begin{bmatrix} x_0\pi_1 \\ \vdots \\ x_0\pi_{N+1} \end{bmatrix} \end{cases} \tag{1.32}$$

e la matrice Z *è soluzione unica dell'equazione algebrica di Riccati:*

$$Z = \mathsf{A}Z\mathsf{A}^T + \mathsf{G}\mathsf{G}^T - \mathsf{A}Z\mathsf{L}^T\left(\mathsf{L}Z\mathsf{L}^T + \mathsf{H}\mathsf{H}^T\right)^{-1}\mathsf{L}Z\mathsf{A}^T + \mathfrak{B}(Q)$$

nella quale si fa ricorso all'operatore lineare:

$$\mathfrak{B}(Q) \triangleq diag\left[\sum_{i=1}^{N+1}\Pi_{ij}A_iQ_iA_i^T\right] - \mathsf{A}diag\left[Q_i\right]\mathsf{A}^T$$

con $Q = (Q_1, \ldots, Q_{N+1})$ *soluzione unica dell'equazione:*

$$Q_j = \sum_{i=1}^{N+1}\Pi_{ij}\left(A_iZ_iA_i^T + \pi_iG_iG_i^T\right)$$

Dall'applicazione applicazione del teorema appena descritto si ottiene uno stimatore del vettore di stato esteso $\tilde{x}_t$, le cui prime n componenti ci danno la stima di x_t.

Nel complesso questo algoritmo ha il difetto che per stimare il vettore x_t di lunghezza n viene realizzato un filtro (1.31) di dimensioni $nN(N+1)$, quindi in pratica non conviene se confrontato con gli stimatori descritti nel paragrafo 1.2.2.

1.3.3 Filtro ottimo a $(N+1)!N$ guadagni costanti

Il filtro precedente presupponeva di non conoscere lo stato in cui si trovava la catena di Markov delle possibili stringhe di ritardi. In realta' dalle misure contenute nel buffer possiamo ricavare delle informazioni parziali sullo stato della catena. Cerchiamo allora di sfruttare queste ulteriori informazioni per sviluppare un algoritmo alternativo.

1.3.3.1 Identificazione dell'insieme degli stati

Si considerano le stringhe di $N = \tau_{max} + 1$ ritardi

$$\mathbf{i} = [\boldsymbol{\tau}_1^i \ldots \boldsymbol{\tau}_N^i]$$

$$\boldsymbol{\theta}(t) = [\boldsymbol{\tau}_{t-N+1}\ \boldsymbol{\tau}_{t-N+2}\ \ldots \boldsymbol{\tau}_{t-1}\ \boldsymbol{\tau}_t]$$

a partire da $t = N$; si ha ancora una catena di Markov con probabilità di transizione

$$P[\boldsymbol{\theta}(t+1) = \mathbf{j} |\ \boldsymbol{\theta}(t) = \mathbf{i}] = P[\ \boldsymbol{\theta}(t+1) = [\boldsymbol{\tau}_1^j \dots \boldsymbol{\tau}_N^j] \mid \boldsymbol{\theta}(t) = [\boldsymbol{\tau}_1^i \dots \boldsymbol{\tau}_N^i]\ ]$$

Naturalmente tale probabilità sarà diversa da zero solo se $[\boldsymbol{\tau}_2^i \dots \boldsymbol{\tau}_N^i] = [\boldsymbol{\tau}_1^j \dots \boldsymbol{\tau}_{N-1}^j]$

$$P[\ \boldsymbol{\theta}(t+1) = [\boldsymbol{\tau}_2 \boldsymbol{\tau}_3 \dots \boldsymbol{\tau}_N \overline{\boldsymbol{\tau}}] \mid \boldsymbol{\theta}(t) = [\boldsymbol{\tau}_1 \boldsymbol{\tau}_2 \dots \boldsymbol{\tau}_{N-1} \boldsymbol{\tau}_N]\ ] = P[\tau(t+1) = \overline{\boldsymbol{\tau}} \mid \tau(t) = \boldsymbol{\tau}_N]$$

Tuttavia questo stato non è noto al tempo t, infatti se un pacchetto non è ancora arrivato non si conosce ancora quale sia il ritardo con il quale arriverà. Numerando gli slot del buffer con k da 1 a N avremo che il ritardo è noto allo slot k se e solo se $\boldsymbol{\tau}_k \leq N - k$, con l'eccezione per la prima slot dove se il pacchetto non è ancora arrivato si sa che $\boldsymbol{\tau}_1 = \infty$.
Nel seguente schema la X indica che con il ritardo τ corrispondente il pacchetto non è ancora arrivato nello slot di buffer in posizione k e quindi il ritardo è incognito.

$\boldsymbol{\tau}$	τ_1	τ_2	τ_3	$\dots$	τ_{N-2}	τ_{N-1}	τ_N
0	0	0	0	$\dots$	0	0	0
1	1	1	1	$\dots$	1	1	X
2	2	2	2	$\dots$	2	X	X
3	3	3	3	$\dots$	X	X	X
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\tau_{max}-2$	$\tau_{max}-2$	$\tau_{max}-2$	$\tau_{max}-2$	$\dots$	X	X	X
$\tau_{max}-1$	$\tau_{max}-1$	$\tau_{max}-1$	X	$\dots$	X	X	X
τ_{max}	τ_{max}	X	X	$\dots$	X	X	X
∞	∞	X	X	$\dots$	X	X	X

Quindi il nuovo stato i sarà una N-pla $[\tau_1^i \tau_2^i \dots \tau_{N-1}^i \tau_N^i]$ dove τ_1^i può avere valori da 0 a τ_{max} e ∞, τ_2^i da 0 a $\tau_{max}-1$ e X e così via fino τ_N^i che può assumere i valori 0 e X, cosicché il numero totale degli stati noti è $(\tau_{max}+2)!$. L'insieme di tali stati è denotato con $\mathcal{N}$. Dove compaiono delle X non si conosce ancora l'entità del ritardo:

$$\begin{aligned}
\tau_2^i &= X \quad \Rightarrow \quad \boldsymbol{\tau}_2 \in \{\tau_{max}, \infty\} \\
\tau_3^i &= X \quad \Rightarrow \quad \boldsymbol{\tau}_3 \in \{\tau_{max}-1, \tau_{max}, \infty\} \\
&\vdots \\
\tau_{N-1}^i &= X \quad \Rightarrow \quad \boldsymbol{\tau}_{N-1} \in \{2, \dots, \tau_{max}, \infty\} \\
\tau_N^i &= X \quad \Rightarrow \quad \boldsymbol{\tau}_N \in \{1, 2, \dots, \tau_{max}, \infty\}
\end{aligned}$$

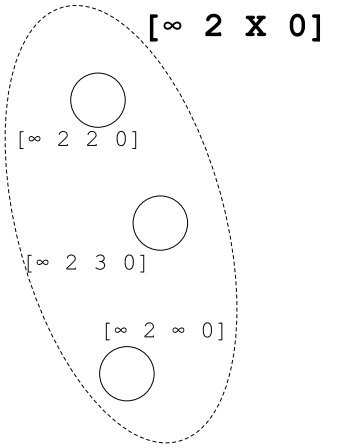

Figura 1.8: Rappresentazione grafica dello stato $i = [\infty\ 2\ X\ 0]$.

Così ad esempio prendendo $\tau_{max} = 3$ lo stato $i = [\infty\ 2\ X\ 0]$ "nasconde" gli stati $[\infty\ 2\ 2\ 0]$, $[\infty\ 2\ 3\ 0]$ e $[\infty\ 2\ \infty\ 0]$ (fig. 1.8).

la probabilità di transizione fra i nuovi stati si tenga conto della seguente considerazione.

Si supponga di avere in una catena di Markov due insiemi di stati disgiunti tra loro a e b come nella figura 1.9.

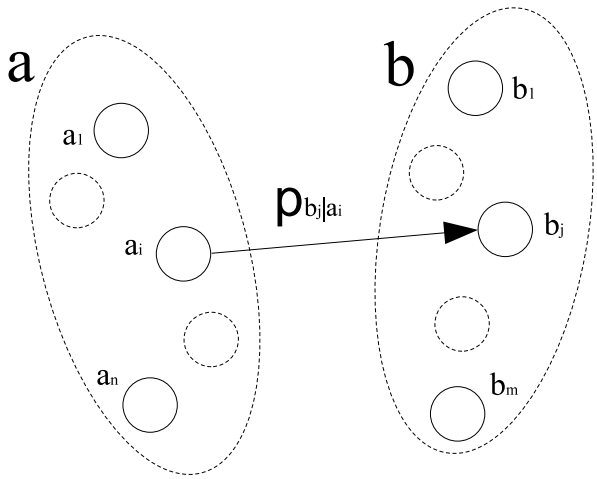

Figura 1.9: Assemblamento di stati in una catena di Markov.

Allora la probabilità di transizione dallo stato a ballo stato b si calcola

$$\begin{aligned} p(b|a) &\overset{(x)}{=} \sum_{j=1}^{m} p(b_j|a) \\ &\overset{(y)}{=} \sum_{j=1}^{m} \frac{\sum_{i=1}^{n} p(b_j|a_i)p(a_i)}{\sum_{i=1}^{n} p(a_i)} \\ &= \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p(b_j|a_i)p(a_i)}{\sum_{i=1}^{n} p(a_i)} \end{aligned}$$

L'uguaglianza (x) si ha perché l'evento $b = \bigcup_{j=1}^{m} b_j$ e gli eventi b_j sono due a due disgiunti. L'uguaglianza (y) si deve al fatto che, avendo gli eventi a_i due a due disgiunti

$$\begin{aligned} P[b_j|a] &= P[b_j|\bigcup_{i=1}^{n} a_i] \\ &= \frac{P[b_j \cap (\bigcup_{i=1}^{n} a_i)]}{P[\bigcup_{i=1}^{n} a_i]} \\ &= \frac{P[\bigcup_{i=1}^{n} (b_j \cap a_i)]}{P[\bigcup_{i=1}^{n} a_i]} \\ &= \frac{\sum_{i=1}^{n} P[b_j|a_i]P[a_i]}{\sum_{i=1}^{n} P[a_i]} \end{aligned}$$

Si osserva che tale probabilità di transizione non è necessariamente tempo-invariante. Questo avviene solo se la catena di Markov di partenza è stazionaria, avendo così $p_{a_i}(t) = p_{a_i}$. Di conseguenza anche la nuova catena di Markov sarà stazionaria, ad esempio per lo stato a visto sopra si avrà

$$p_a = \sum_{i=1}^{n} p_{a_i}$$

Questo discorso vale anche per la probabilità di tansizione $p_{a|a}$ Applicando quanto visto si ottiene la probabilitá di transizione tra gli stati noti dell'insieme $\mathcal{N}$.

1.3.3.2 Implementazione del filtro

La variabile aleatoria di arrivo $\gamma_{t_2}^{t_1}$ è definita

$$\begin{cases} \gamma_{t_2}^{t_1} = 1 & \text{se il pacchetto relativo a } y_{t_2} \text{ e' arrivato al tempo } t_1 \\ \gamma_{t_2}^{t_1} = 0 & \text{altrimenti} \end{cases}$$

Per $k = 1, \ldots, N$ si può scrivere γ_{t-N+k}^{t} in funzione di $\theta(t)$ e k, più precisamente $\gamma_{t-N+k}^{t} = \Gamma_k^{\theta(t)}$ dove la funzione Γ_k^i è definita

$$\begin{cases} \Gamma_k^i = 1 & \text{se } \tau_k^i \leq N - k \\ \Gamma_k^i = 0 & \text{altrimenti} \end{cases}$$

Esempio con $\tau_{max} = 3$:

$$i = [\infty \ 2 \ X \ 0] \Rightarrow \begin{cases} \Gamma_1^i = 0 \\ \Gamma_2^i = 1 \\ \Gamma_3^i = 0 \\ \Gamma_4^i = 1 \end{cases}$$

Si può quindi scrivere il dato arrivato al tempo t relativo a y_{t-N+k}

$$\tilde{y}^t_{t-N+k} = \gamma^t_{t-N+k} y_{t-N+k} = \Gamma^{\theta(t)}_k y_{t-N+k}$$

Selezionando ad ogni istante t lo stato $i = \theta(t)$ la struttura del filtro a buffer finito è la seguente

$$\begin{aligned}
\hat{x}^t_{t-N+1|t-N+1} &= A\hat{x}^t_{t-N|t-N} + K^i_1(\tilde{y}^t_{t-N+1} - \Gamma^i_1 CA\hat{x}^t_{t-N|t-N}) \\
&= A\hat{x}^{t-1}_{(t-1)-N+1|(t-1)-N+1} \\
&\quad + K^i_1(\tilde{y}^t_{t-N+1} - \Gamma^i_1 CA\hat{x}^{t-1}_{(t-1)-N+1|(t-1)-N+1}) \\
\hat{x}^t_{t-N+k|t-N+k} &= A\hat{x}^t_{t-N+k-1|t-N+k-1} \\
&\quad + K^i_k(\tilde{y}^t_{t-N+k} - \Gamma^i_k CA\hat{x}^t_{t-N+k-1|t-N+k-1})
\end{aligned} \tag{1.33}$$

per $k = 2, \ldots, N$. Quindi il numero totale di guadagni da memorizzare e' $(\tau_{max} + 2)!(\tau_{max} + 1)$.Si definisce la matrice di covarianza dell'errore di stima

$$P^{t_1}_{t_2|t_3} \stackrel{\text{def}}{=} E[(x_{t_2} - x^{t_1}_{t_2|t_3})(x_{t_2} - x^{t_1}_{t_2|t_3})^T]$$

Considerando che i pacchetti arrivano con ritardi aleatori

$$E[P^t_{t|t}] = \sum_i E[P^t_{t|t}|\theta(t) = i]v_i(t) = \sum_i E[P^t_{t|t}|\theta(t) = i]v^{ss}_i \tag{1.34}$$

Si cercherà allora il filtro $\{K^i_k, i \in \mathcal{N}\ k = 1, \ldots, N\}$ tale da rendere

$$\min \lim_{t\to\infty} \mathrm{tr}(E[P^t_{t|t}])$$

se effettivamente esiste. Si ricorda che se due matrici X e Y definite positive sono tali che $X \leq Y$ allora a maggior ragione $\mathrm{tr}(X) \leq \mathrm{tr}(Y)$. Si definisce

$$\tilde{P}^{t_1,i}_{t_2|t_3} \stackrel{\text{def}}{=} E[P^{t_1}_{t_2|t_3}|\theta(t_1) = i]$$

Per l'equazione 1.34 si ha che il problema di minimizzazione si ridue a minimizzare $\lim_{t\to\infty} \mathrm{tr}(\tilde{P}^{t,i}_{t|t})$ per ogni $i \in \mathcal{N}$.
Considerando che indipendentemente dai guadagni del filtro scelti si ha

$$\tilde{P}^{t,i}_{t+1|t} = A\tilde{P}^{t,i}_{t|t}A^T + Q$$

e che

$$P_1 \leq P_2 \leq 0 \Rightarrow AP_1A^T + Q \leq AP_2A^T + Q$$

allora minimizzare $\lim_{t\to\infty} \mathrm{tr}(\tilde{P}^{t,i}_{t|t}])$ significa minimizzare $\lim_{t\to\infty} \mathrm{tr}(\tilde{P}^{t,i}_{t+1|t})$.
Si ricava che le matrici di covarianza si aggiornano nel seguente modo:

$$\begin{aligned}\tilde{P}^{t,i}_{t-N+k+1|t-N+k} &= A(I-\Gamma^i_k K_k C)\tilde{P}^{t,i}_{t-N+k|t-N+k-1}(I-\Gamma^i_k K_k C)^T A^T + \\ &\quad + Q + \Gamma^i_k A K_k R K_k^T A^T\end{aligned}$$

per $k = 2, \ldots, N$. Si definisce l'operatore

$$\mathcal{A}(K, P, \Gamma) \stackrel{\text{def}}{=} A(I - \Gamma K C)P(I - \Gamma K C)^T A^T + Q + \Gamma A K R K^T A^T$$

la matrice

$$K_P \stackrel{\text{def}}{=} \Gamma P C^T (C P C^T + R)^{-1}$$

e ancora l'operatore

$$\mathcal{R}(P, \Gamma) \stackrel{\text{def}}{=} \mathcal{A}(K_P, P, \Gamma) = APA^T + Q - \Gamma A P C^T (C P C^T + R)^{-1} C P A^T$$

I seguenti lemmi possono essere dimostrati come in [7].

Lemma 1 $\mathcal{A}(K, P, \Gamma) = \mathcal{R}(P, \Gamma) + \Gamma A(K - K_P)(C P C^T + R)(K - K_P)^T A^T$

Lemma 2 $\mathcal{A}(K, P, \Gamma) \geq \mathcal{R}(P, \Gamma) = \mathcal{A}(K_P, P, \Gamma)$

Lemma 3 $P_1 \geq P_2 \;\Rightarrow\; \mathcal{R}(P_1, \Gamma) \geq \mathcal{R}(P_2, \Gamma)$

Teorema 7 *Si supponga che esitsta* $\{K^j_1, j \in \mathcal{N}\}$ *tale che* $\lim_{t\to\infty} \tilde{P}^{t,i}_{t-N+2|t-N+1}$ *esista e sia minimo tra tutte le possibili scelte. Allora per ogni* $i \in \mathcal{N}$ *il filtro* $\{K^i_2 \ldots K^i_N\}$ *che minimizza* $\lim_{t\to\infty} \tilde{P}^{t,i}_{t+1|t}$ *è dato da*

$$K^i_{k+1} = \Gamma^i_{k+1} V^i_k C^T (C V^i_k C^T + R)^{-1}$$

per $k = 1, \ldots N-1$ *dove si è posto* $V^i_1 = \lim_{t\to\infty} \tilde{P}^{t,i}_{t-N+2|t-N+1}$ *ottenuto con* $\{K^j_1, j \in \mathcal{N}\}$ *e* $V^i_{k+1} = \mathcal{R}(V^i_k, \Gamma^i_k)$.

Dimostrazione

Si consideri un qualsiasi altro set di guadagni $\{A^j_1, j \in \mathcal{N}\}$, tale che $\lim_{t\to\infty} \tilde{P}^{t,i}_{t-N+2|t-N+1}$ esista, e $\{A_2 \ldots A_N\}$.

Si ponga $T^i_1 = \lim_{t\to\infty} \tilde{P}^{t,i}_{t-N+2|t-N+1}$ usando il guadagno $\{A^j_1, j \in \mathcal{N}\}$ e $T^i_{k+1} = \mathcal{A}(A_k, T^i_k, \Gamma^i_k)$ per $k = 1, \ldots, N-1$.

Si ha quindi che $\lim_{t\to\infty} \tilde{P}^{t,i}_{t-N+k+1|t-N+k}$ è uguale a V^i_k oppure a T^i_k rispettivamente nel caso in cui si usino i guadagni K^i_k oppure A_k.

Si tratta quindi di dimostrare che $V^i_N \leq T^i_N$; questo è vero per induzione, infatti per ipotesi si ha $V^i_1 \leq T^i_1$, inoltre se $V^i_k \leq T^i_k$ allora per i lemmi 2 e 3 $V^i_{k+1} = \mathcal{R}(V^i_k, \Gamma^i_k) \leq \mathcal{R}(T^i_k, \Gamma^i_k) \leq \mathcal{A}(A_k, T^i_k, \Gamma^i_k) = T^i_{k+1}$ □

1.3.3.3 Filtro ottimo con guadagni costanti matrice C dipendente dallo stato $\theta(t)$

La prima delle equazioni 1.33 può essere riscritta per ogni $t \geq 1$

$$\begin{aligned}\hat{x}_{t|t}^{t+N-1} &= A\hat{x}_{t-1|t-1}^{(t-1)+N-1} + \\ &\quad + K_1^i(\tilde{y}_t^{t+N-1} - \Gamma_1^i CA\hat{x}_{t-1|t-1}^{(t-1)+N-1})\end{aligned}$$

Scopo di $\{K_1^i, i \in \mathcal{N}\}$ è minimizzare $\lim_{t\to\infty} \tilde{P}_{t-N+2|t-N+1}^{t,i}$ per ogni $i \in \mathcal{N}$ nel caso in cui esista. Con le seguenti definizioni

$$\begin{aligned}\bar{\theta}(t) &\stackrel{def}{=} \theta(t+N-1) \\ \bar{K}^i &\stackrel{def}{=} K_1^i \\ \bar{\Gamma}^i &\stackrel{def}{=} \Gamma_1^i \\ C_i &\stackrel{def}{=} \bar{\Gamma}^i C \\ \bar{y}_t &\stackrel{def}{=} \tilde{y}_t^{t+N-1}\end{aligned}$$

si riduce il problema a stimare lo stato a partire dalla conoscenza di $\bar{y}_t$:

$$\begin{cases} x_{t+1} &= Ax_t + w_t \\ \bar{y}_t &= C_{\bar{\theta}(t)}x_t + v_t \end{cases}$$

minimizzando $\lim_{t\to\infty} \bar{Z}_i(t)$ per ogni $i \in \mathcal{N}$ definita come segue:

$$\begin{aligned}\bar{Z}_i(t) &\stackrel{def}{=} E[P_{t|t}^{t+N-1}|\theta(t+N-1) = i] = \tilde{P}_{t|t}^{t+N-1,i} \\ \bar{M}_i(t) &\stackrel{def}{=} E[P_{t|t-1}^{t+N-1}|\theta(t+N-1) = i] = \tilde{P}_{t|t-1}^{t+N-1,i}\end{aligned}$$

Questo problema viene risolto in [13]. Viene introdotta la probabilità a posteriori

$$p_{ij}^*(t) \stackrel{def}{=} P[\bar{\theta}(t-1) = j|\bar{\theta}(t) = i] = \frac{v_j(t-1)p_{ji}}{v_i(t)} = \frac{v_j^{ss}p_{ji}}{v_i^{ss}}$$

e l'operatore

$$\mathcal{A}_{\bar{K}^i}(M) \stackrel{def}{=} (I - \bar{K}^i C_i)M(I - \bar{K}^i C_i)^T + \bar{K}^i R(\bar{K}^i)^T$$

Si dimostra che le covarianze d'errore definite sopra evolvono come

$$\begin{aligned}\bar{Z}_i(t) &= \mathcal{A}_{\bar{K}^i}(\bar{M}_i(t)) \\ \bar{M}_i(t) &= \sum p_{ij}^*(A\bar{Z}_j(t-1)A^T + Q)\end{aligned}$$

Definizione 3 *Un filtro $\{\bar{K}^i, i \in \mathcal{N}\}$ è detto stabile se e solo se esiste $c \in \mathbb{R}$ tale che $tr(E[Z_i(t)]) \leq c < \infty \ \forall t \ \forall i$.*

Teorema 8 *Sia $(A, Q^{\frac{1}{2}})$ controllabile. Se il filtro è stabile allora esiste il* $\lim_{t\to\infty} tr(E[Z_i(t)])$
.

Teorema 9 *Sia $(A, Q^{\frac{1}{2}})$ controllabile. Per ogni $i \in \mathcal{N}$*

$$\mathcal{R}_i(M) \stackrel{def}{=} M - MC_i{}^T(C_iMC_i{}^T + R)^{-1}C_iM$$

La $\mathcal{R}$-iterazione è allora definita per tutti gli i come

$$\begin{cases} Z_i(t) &= \mathcal{R}_i(M_i(t)) \\ M_i(t) &= \sum p_{ij}^*(AZ_j(t-1)A^T + Q) \end{cases}$$

Esiste un filtro stabile se e solo se la $\mathcal{R}$-iterazione converge per le condizioni iniziali $Z_i(0) = 0\ \forall i$.

Si osservi come la $\mathcal{R}$-iterazione fornisca l'evoluzione della covarianza d'errore con l'utilizzo dei guadagni ottimi del filtro di Kalman tempo-variante. Per trovare i guadagni costanti ottimi si utilizzano i valori a regime come spiegato nel seguente teorema.

Teorema 10 *Sia assuma che la $\mathcal{R}$-iterazione con condizioni iniziali $Z_i = 0\ \forall i$ converga a Z_i^{ss} con corrispondente $M_i^{ss} = \sum p_{ij}^*(AZ_j^{ss}A^T + Q)$. I guadagni*

$$\bar{K}^i = M_i^{ss}C_i{}^T(C_iM_i^{ss}C_i{}^T + R)^{-1}$$

sono stabilizzanti e si ha $\lim_{t\to\infty} \bar{Z}_i(t) = Z_i^{ss}$ *per una qualsiasi scelta delle condizioni iniziali $\bar{Z}_i(0)$; per ogni altra scelta di gudagni stabilizzanti A^i si ha che* $\lim_{t\to\infty} \tilde{Z}_i(t) \geq Z_i^{ss}\ \forall i$

Il numero totale di guadagni e' $(\tau_{max} + 2)!(\tau_{max} + 1) = (N+1)!N$, cioe' per ogni possibile riempimento degli N slot del buffer dobbiamo memorizzare $(N+1)!$gruppi di guadagni costanti $\left\{K_k^i\right\}_{k=1}^N$con $i = 1...(N+1)!$. Questo, come gia' risulta essere sub-ottimo, pero' il numero di guadagni da memorizzare cresce rapidamente al crescere di N. Ad esempio nel caso in cui $N = 5$ risulta necessario memorizzare ben 480 guadagni costanti. Ne consegue che la sua applicazione pratica e' limitata ai soli casi in cui τ_{max} sia piccolo (in pratica al massimo per $N = 3$). Riteniamo percio' che anche questo algoritmo non sia praticabile.

1.4 Simulazioni e valutazioni degli stimatori

In questo capitolo tramite alcune simulazioni daremo delle valutazioni agli algoritmi di stima descritti in questa prima parte.

1.4.1 Algoritmi di stima con perdita dei pacchetti

Per le nostre simulazioni consideriamo di dover stimare il seguente sistema :

$$\begin{aligned} A &= \begin{bmatrix} 0.9 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.7 \end{bmatrix} \\ C &= \begin{bmatrix} 1 & 2 & 4 \end{bmatrix} \\ R &= 30 \\ Q &= \begin{bmatrix} 20 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 20 \end{bmatrix} \end{aligned}$$

con condizioni iniziali:

$$\begin{aligned} x_0 &= \begin{bmatrix} 5 \\ 9 \\ 8 \end{bmatrix} \\ P_0 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{aligned}$$

e con processo di perdita dei pacchetti a catena di Markov determinato dai parametri:

$$\begin{matrix} p = 0.3 \\ q = 0.6 \end{matrix} \quad \Rightarrow \quad \Pi = \begin{bmatrix} 0.7 & 0.3 \\ 0.6 & 0.4 \end{bmatrix}$$

Il sistema risulta perciò essere stabile e nel caso dei DTJMLS si può verificare che il sistema è MSS.

Dalle simulazioni (Figura 1.10 e 1.11) possiamo osservare che, come previsto, la traccia della varianza d'errore dello stimatore ottimo tempo variante (P_{TV}) risulta essere il limite inferiore per le altre tracce. Constatiamo che lo stimatore tempo invariante a 2 guadagni costanti (P_{TI2}) fornisce una varianza d'errore inferiore allo stimatore sub-ottimo tempo invariante ad un solo guadagno costante (P_{TI1}). Per quanto riguarda il filtro sub ottimo basato sui DTJMLS (P_{DTJMLS}) al contrario in questo caso la traccia della varianza d'errore risulta superiore a tutte le altre, anche se comunque molto prossima a queste. Forse questo e' dovuto alla particolare realizzazione del processo di arrivo dei pacchetti di questa simulazione. Queste considerazioni si trasferiscono come ovvio ai tracciati delle effetive norme quadratiche degli errori di stima, da cui pero' il filtro basato sui DTJMLS da' prova della sua ottimalita'.

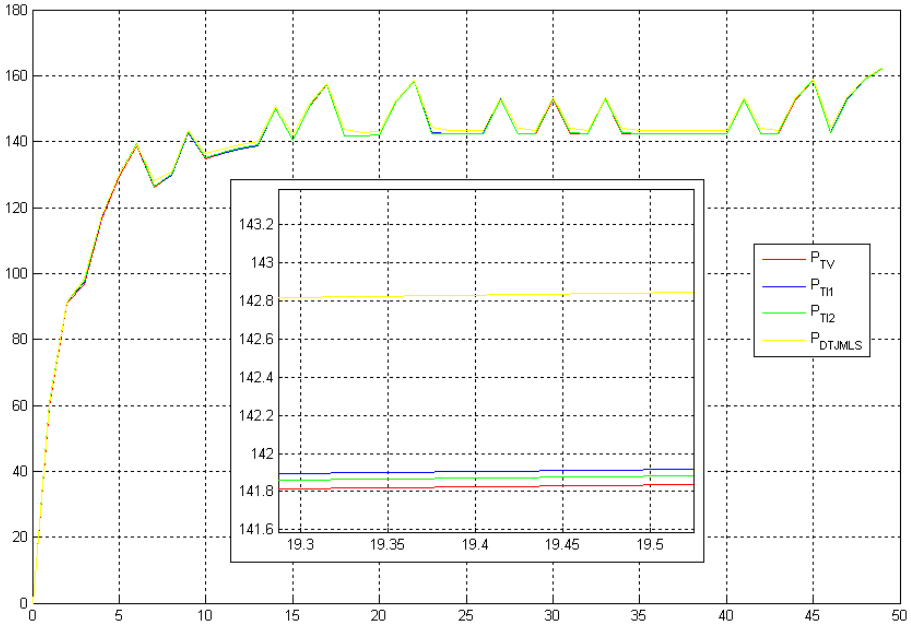

Figure 1.10: Traccia della varianza degli errori di stima con A stabile.

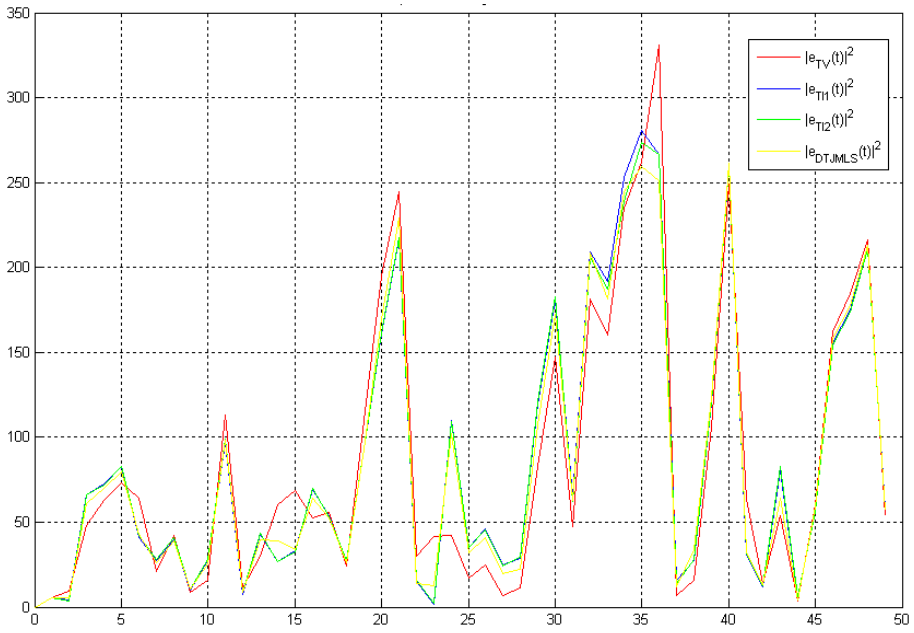

Figure 1.11: Norma quadratica dell'errore di stima con A stabile.

Modifichiamo ora la matrice di stato A:

$$A = \begin{bmatrix} 1.3 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.7 \end{bmatrix}$$

in cui abbiamo introdotto l'autovalore 1.1 tale da rendere il sistema instabile (e non più MSS).

Ripetendo le simulazioni con il nuovo sistema ora instabile (Figure 1.12 e 1.13) possiamo ancora constatare, nonostante non siano piu' soddisfatte le ipotesi del teorema 2.1 sull'ottimalita' dell'algoritmo sub-ottimo a guadagno costante, che le prestazioni a regime sono ancora molto buone se confrontate con le prestazioni del filtro tempovariante ottimo. Inoltre valgono ancora tutte le altre considerazioni fatte per A stabile. Questo ci fa pensare che sia possibile in futuro poter dimostrare l'ottimalita' del filtro a guadagno costante anche nel caso di A instabile sotto opportune ipotesi sul processo di perdita dei pacchetti.

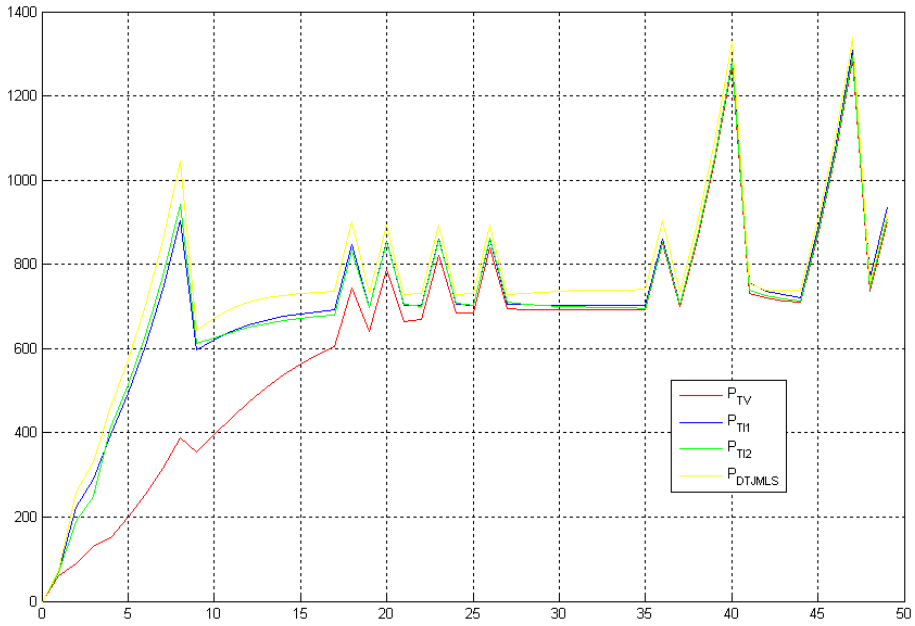

Figure 1.12: Traccia della varianza degli errori di stima con A instabile.

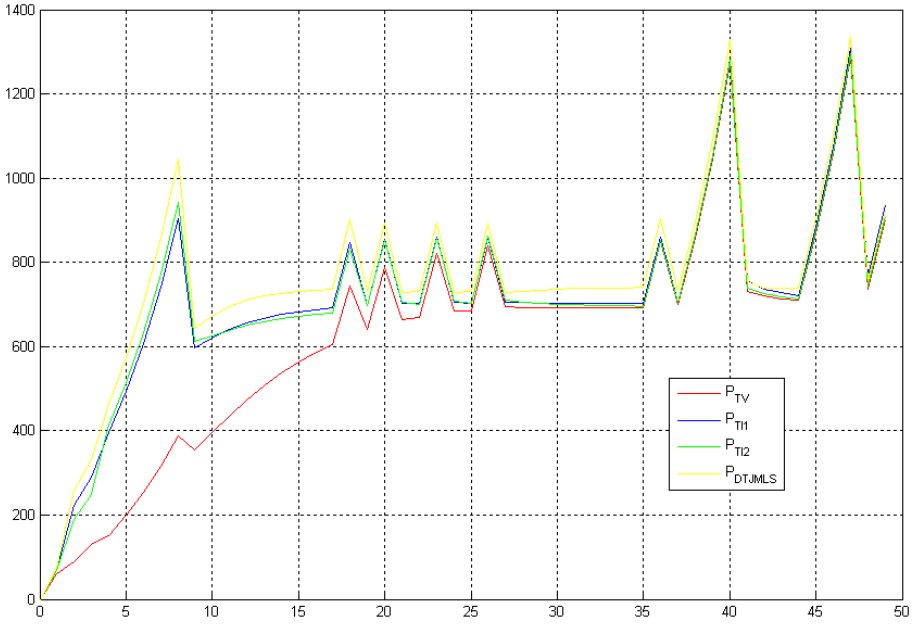

Figure 1.13: Norma quadratica dell'errore di stima con A instabile.

1.4.2 Algoritmi di stima con perdita dei pacchetti e ritardo

Andiamo ora a simulare gli algoritmi di stima con perdita dei pacchetti e ritardo piu' validi descritti precedentemente e valutarne così le prestazioni.

Per le simulazioni di questi stimatori prendiamo ancora in esame il sistema visto precedentemente, mentre il processo di perdita dei pacchetti con ritardo considerato è descritto dalla seguente matrice stocastica regolare:

$$\Pi = \begin{bmatrix} 0.4 & 0.3 & 0.15 & 0.1 & 0.04 & 0.005 & 0.005 \\ 0.25 & 0.3 & 0.25 & 0.1 & 0.05 & 0.03 & 0.02 \\ 0.1 & 0.2 & 0.2 & 0.2 & 0.15 & 0.1 & 0.05 \\ 0.1 & 0.1 & 0.15 & 0.2 & 0.2 & 0.15 & 0.1 \\ 0.05 & 0.1 & 0.15 & 0.15 & 0.2 & 0.2 & 0.15 \\ 0.04 & 0.06 & 0.1 & 0.15 & 0.2 & 0.25 & 0.2 \\ 0.02 & 0.03 & 0.1 & 0.15 & 0.2 & 0.2 & 0.3 \end{bmatrix}$$

Dalle simulazioni eseguite (Figure 1.14 e 1.15), l'andamento effettivo della traccia della varianza d'errore ci confera l'ottimalita' del filtro sub-ottimo tempo invariante ad N guadagni costanti (P_{TIN}) se rapportato alle prestazioni del

filtro ottimo tempo variante (P_{TV}). Si verifica inoltre la bonta' del filtro tempo invariante ad N^2guadagni costanti (P_{TIN^2}), la cui traccia della varianza d'errore rimane costantemente limitata tra quella del filtro ottimo e quella del filtro sub-ottimo.

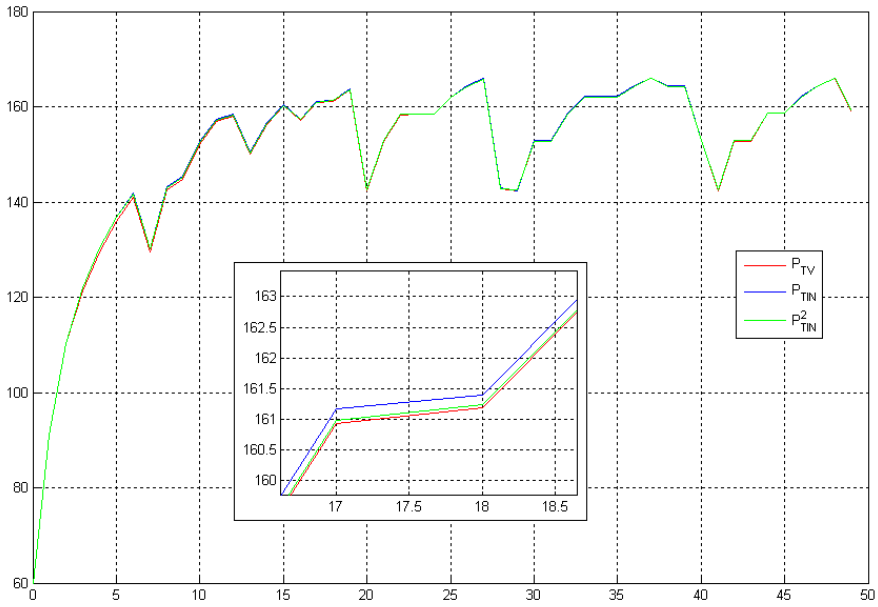

Figure 1.14: Traccia della varianza degli errori di stima in presenza di ritardi con A stabile.

Rendendo nuovamente instabile il sistema in esame come fatto precedentemente, dalle simulazioni (Figure 1.16 e 1.17) possiamo ancora constatare, nonostante non siano piu' soddisfatte le ipotesi del teorema 2.2 sull'ottimalita' dell'algoritmo sub-ottimo ad N guadagni costanti, che le prestazioni a regime sono ancora molto buone se confrontate con le prestazioni del filtro tempovariante ottimo. Inoltre valgono ancora tutte le altre considerazioni fatte per A stabile. Anche in questo caso ipotizziamo che sia possibile in futuro poter dimostrare l'ottimalita' del filtro a guadagno costante anche nel caso di A instabile sotto opportune ipotesi sul processo di perdita dei pacchetti.

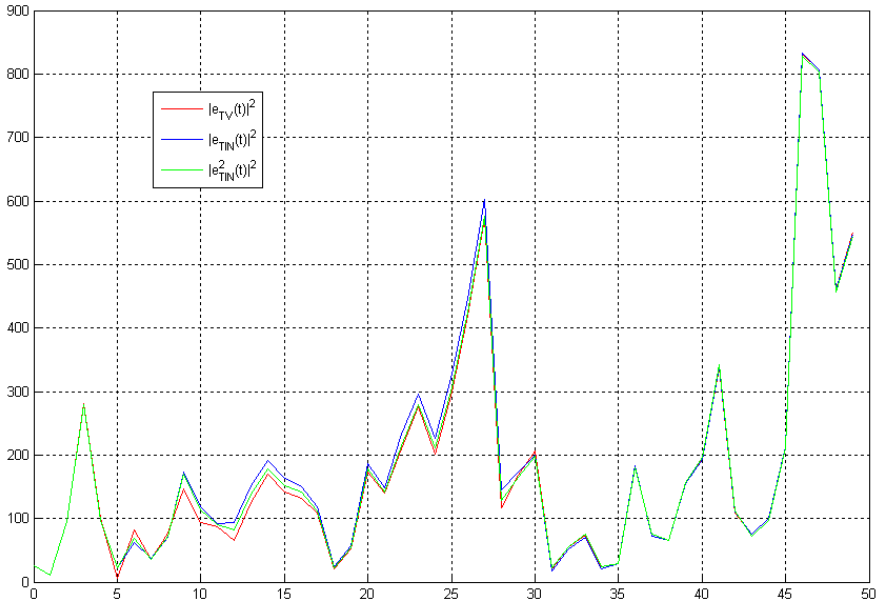

Figure 1.15: Norma quadratica dell'errore di stima in presenza di ritardi con A stabile.

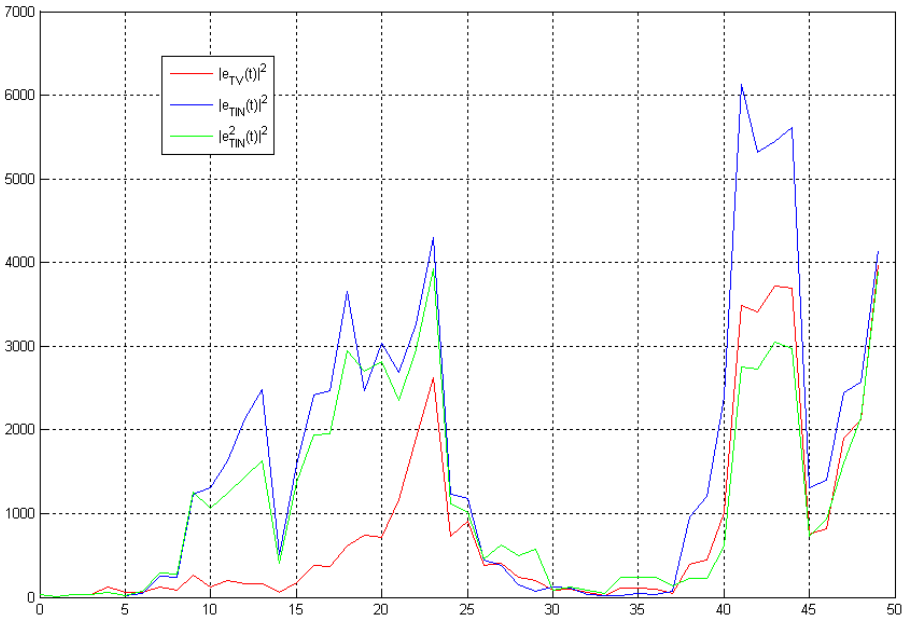

Figure 1.17: Norma quadratica dell'errore di stima in presenza di ritardi con A stabile.

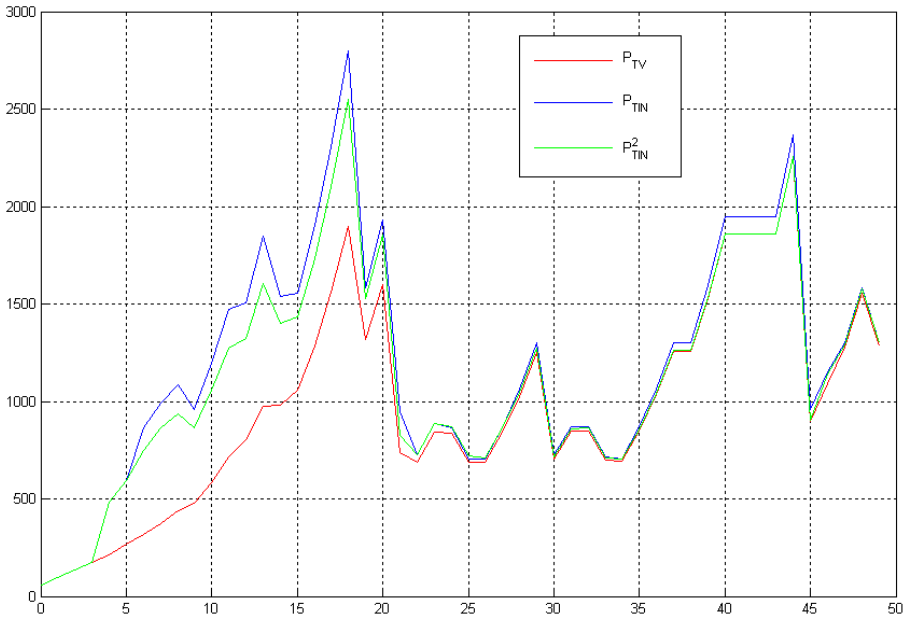

Figure 1.16: Traccia della varianza degli errori di stima in presenza di ritardi con A instabile.

1.4.3 Conclusioni

In generale dalle simulazioni abbiamo potuto verificare in generale la bonta' delle prestazioni dei diversi algoritmi descritti in questo lavoro. Dai risultati delle prestazioni fornite osserviamo che per quanto riguarda gli stimatori a guadagni costanti, sia nel caso di sola perdita di pacchetti e specialmente nel caso di perdita con ritardi, all'aumentare del numero di guadagni costanti utilizzati dallo stimatore non corrisponde un aumento delle prestazioni tale da giustificare il costo legato alla loro memorizzazione.

Capitolo 2

Stima distribuita: fusione di dati e perdita di pacchetti

2.1 Introduzione

I recenti sviluppi nell'ambito della tecnologia wireless e dell'elettronica a bassa potenza hanno consentito un'ampia diffusione di componenti dotati di capacità di misura, calcolo e comunicazione (*smart sensors*), che possono essere efficientemente utilizzati per problemi di stima distribuita. In molte di queste applicazioni, la rete di sensori wireless ha una struttura ad albero, nel senso che tutti i sensori misurano o stimano la grandezza di interesse ed inviano i dati al proprio padre; le informazioni raccolte dalla rete arrivano così ad una unità centrale, detta *base station*, presso la quale si calcola la stima ottima, secondo un qualche criterio.

In condizioni ideali, senza perdita o ritardo di arrivo dei pacchetti, la stima ottima alla base station si ottiene con il filtro di Kalman, utilizzando tutte le misure effettuate dai sensori. Va però considerato che oltre alla perdita di pacchetti, uno dei problemi principali delle reti di sensori wireless è che i dispositivi impiegati hanno una fonte di energia limitata, perciò risulta necessario realizzare un'efficiente procedura di comunicazione tra i nodi. Dato che l'energia richiesta al sensore per la comunicazione è molto maggiore di quella richiesta per la misura ed il calcolo (si veda [2]), generalmente si cerca di minimizzare il numero di trasmissioni, facendo eventualmente elaborare le misure direttamente ai sensori. Per questo motivo, si sta sviluppando un'ampia letteratura sull'argomento, nel tentativo di trovare metodi di stima che minimizzino le trasmissioni tra i nodi e tengano conto anche della perdita di pacchetto.

In realtà, anche nel caso non si effettui alcun tipo di fusione dei dati, il problema della stima distribuita con perdita di pacchetto rimane in parte irrisolto, in quanto non è ancora stato trovato un metodo di stima che consenta il recupero di tutta l'informazione persa, al momento della ricezione di un nuovo dato alla base station.

Perciò, in questo lavoro i problemi di fusione e di perdita di pacchetto verranno analizzati separatamente. Anzitutto, sulla base di opportune riformulazioni del filtro di Kalman, verranno presentate alcune tecniche note di aggregazione e fusione dei dati, e verrà proposto un metodo ottimo di fusione delle stime locali, trascurando perdita di pacchetto e ritardi aleatori. Successivamente, si considererà il problema della perdita di pacchetto nel caso in cui i nodi non effettuino fusione dei dati e quindi alla base station vengano inviate tutte le stime o le misure dei sensori. Verranno proposte alcune generalizzazioni al caso con perdita di pacchetto, degli stessi algoritmi di stima distribuita che saranno la base delle tecniche di fusione di stime locali. Infine, si analizzeranno le prestazioni di questi algoritmi, mediante simulazione.

2.2 Formulazione del problema

Si consideri il seguente modello di stato per il processo osservato dai sensori:

$$\begin{aligned} x_{t+1} &= Ax_t + w_t \\ y_t^i &= C_i x_t + v_t^i\,, \qquad i = 1, ..., M \end{aligned} \tag{2.1}$$

dove M è il numero di sensori, $t \in \mathbb{N}$, $A \in \mathbb{R}^{n\times n}$, x_t, $w_t \in \mathbb{R}^n$, $y_t^i \in \mathbb{R}^{m_i}$, $C_i \in \mathbb{R}^{m_i \times n}$.

Il processo y_t^i descrive le osservazioni effettuate dal sensore i-esimo e v_t^i è il relativo processo di rumore di misura. Siano w_t e v_t^i bianchi, Gaussiani a media zero e scorrelati $\forall i$, cioè tali che $\mathbb{E}\left[w_t w_s'\right] = Q\delta\left(t-s\right)$, $\mathbb{E}\left[v_t^i w_s'\right] = 0$ e $\mathbb{E}\left[v_t^i \left(v_s^j\right)'\right] = R_{ij}\delta\left(t-s\right)$. Sia inoltre lo stato iniziale x_0 Gaussiano bianco, a media $\bar{x}_0$ e varianza P_0.

Sia $C = \left[\begin{array}{ccc} C_1' & \dots & C_M' \end{array}\right]' \in \mathbb{R}^{m\times n}$, dove $m = \sum_i m_i$, la matrice completa di osservazione; supponiamo (A, C) osservabile e $\left(A, Q^{1/2}\right)$ raggiungibile in modo da garantire l'esistenza di uno stimatore stabile.

Il processo di osservazione complessivo è descritto da

$$y_t = Cx_t + v_t\,, \tag{2.2}$$

dove $y_t = \left[\begin{array}{c} y_t^1 \\ \vdots \\ y_t^M \end{array}\right]$ e $v_t = \left[\begin{array}{c} v_t^1 \\ \vdots \\ v_t^M \end{array}\right]$, con matrice varianza R.

Come è noto, la stima ottima (cioè a minima varianza d'errore) all'istante t sulla base di tutte le misure effettuate fino a t è

$$\hat{x}\left(t \mid t\right) = \mathbb{E}\left[x_t \mid y_t, \dots y_0\right]\,, \tag{2.3}$$

la quale è calcolabile ricorsivamente mediante il classico filtro di Kalman.

Per quanto riguarda la rete di sensori, essa è caratterizzata da un grafo di comunicazione che descrive i collegamenti tra i nodi e che supponiamo ne

definisca una struttura ad albero: ogni nodo riceve i dati dai propri figli e li trasmette al padre. Ad un particolare nodo (*radice*) viene affidato il compito di ricevere tutti i dati dai sensori della rete e spedirli alla base station, presso la quale i dati vengono elaborati per ottenere la stima globale ottima della grandezza di interesse.

Supponiamo inoltre che i sensori siano dispositivi con possibilità di calcolo e di memoria e che siano tutti sincronizzati.

2.3 Aggregazione e fusione dei dati in assenza di perdita di pacchetto

In questa sezione analizzeremo il problema della minimizzazione delle trasmissioni effettuate da ogni nodo della rete.

In assenza di perdita di pacchetto o ritardo aleatorio, sono state sviluppate in letteratura varie soluzioni che consentono di ottenere alla base station la stima ottima (2.3). Le soluzioni possono avere fondamentalmente due approcci: uno prevede l'aggregazione dei dati man mano che questi vengono trasmessi verso la radice, l'altro ne realizza invece una fusione, per mezzo di opportune funzioni implementate ai nodi.

L'approccio aggregativo sarà trattato molto brevemente, dato che c'è un'ampia letteratura a riguardo e lo scopo di questa sezione è principalmente l'analisi di metodi di fusione di stime locali.

2.3.1 Metodo aggregativo

L'approccio aggregativo prevede che ogni nodo trasmetta in un unico pacchetto sia la propria misura che quelle a lui arrivate dai figli. In ogni istante t, i nodi della rete ricevono dai figli un pacchetto di misure, aggiungono la propria misura ed inviano il pacchetto così ottenuto al proprio genitore. Perchè una volta giunte alla base station, le misure possano essere correttamente ordinate a formare il vettore y_t, queste vengono allocate nei pacchetti assieme all'indirizzo del sensore che le ha effettuate.[1] Alla base station è sufficiente riordinare le misure ed implementare le equazioni che forniscono la stima globale ottima, quindi il filtro di Kalman basato sul modello di stato (2.1)-(2.2).

L'aggregazione delle misure riduce notevolmente il numero di pacchetti trasmessi dai nodi in ogni istante e migliora l'efficienza della comunicazione, limitando il numero di bit di controllo visto che dati provenienti da sensori diversi vengono inviati con un unico header.

[1] Va osservato che per questa soluzione non è necessario che la rete abbia una struttura di comunicazione ad albero; i dati potrebbero arrivare alla base station anche attraverso percorsi diversi, eventuali doppioni possono essere eliminati.

2.3.2 Metodi di fusione

In generale, le tecniche di fusione si basano su algoritmi di stima la cui struttura consente di calcolare la stima globale alla base station in funzione di quantità che possono essere agevolmente calcolate dai sensori e inviate alla radice. Gli algoritmi qui presentati sono opportune riformulazioni del filtro di Kalman che si adattano a tecniche di fusione, se si ipotizza che i rumori di misura dei sensori siano scorrelati, ovvero $R = diag\,(R_1, \ldots, R_M)$.

2.3.2.1 Algoritmo 1 (Filtro di Kalman in forma d'informazione)

E' noto in letteratura che, supponendo i rumori di misura degli M sensori tutti scorrelati, le equazioni del filtro di Kalman possono essere riscritte nel modo seguente:

$$\hat{x}_{t+1|t} = A\hat{x}_{t|t} \tag{2.4}$$
$$P_{t+1|t} = AP_{t|t}A' + Q\,, \tag{2.5}$$

$$\hat{x}_{t+1|t+1} = P_{t+1|t+1}\left(P^{-1}_{t+1|t}\hat{x}_{t+1|t} + \sum_{i=1}^{M} C_i' R_i^{-1} y_t^i\right) \tag{2.6}$$
$$P_{t+1|t+1} = \left(P^{-1}_{t+1|t} + \sum_{i=1}^{M} C_i' R_i^{-1} C_i\right)\,, \tag{2.7}$$

da cui si può vedere che se ogni sensore calcola e trasmette in ogni istante t la quantità $z_t^i = C_i' R_i^{-1} y_t^i$, la stima alla base station può essere calcolata secondo le equazioni (2.4), (2.5), (2.6), (2.7), dopo aver sommato i dati inviati, in quell'istante, dai sensori. Queste equazioni suggeriscono allora una tecnica di fusione dei dati ai nodi, secondo cui la somma $\sum_{i=1}^{M} z_t^i$ può essere calcolata direttamente dalla rete; ogni nodo calcola

$$\hat{x}_t^i = z_t^i + \sum_{j \in f(i)} \hat{x}_t^j\,, \tag{2.8}$$

dove $f\,(i)$ è l'insieme di figli nel nodo i-esimo. E' facile vedere che, in questo modo, al nodo radice si ottiene proprio $\sum_{i=1}^{M} z_t^i$.

Questa tecnica si implementa efficientemente quando la rete ha una struttura di comunicazione ad albero, per cui non c'è il rischio di sommare più volte uno stesso dato; se la rete non ha struttura ad albero, è necessario che ogni nodo trasmetta assieme al dato $\hat{x}_t^i$ anche gli indirizzi di tutti i sensori (non solo dei figli) che hanno contribuito alla somma $\hat{x}_t^i$, in modo che quando un nodo riceve i dati dei figli, possa scartare eventuali doppioni prima di sommarli.

Si può dunque osservare che questo approccio fonde assieme delle "misure" z_t^i effettuate dai nodi, facendo sì che diminuisca il traffico in rete, il numero di bit di informazione e quindi anche la dimensione dei pacchetti trasmessi dai sensori.

2.3.2.2 Algoritmo 2

Nell'ipotesi fatta nella seconda sezione, che i rumori di modello e di osservazione siano scorrelati, la stima filtrata di Kalman si può aggiornare mediante[2]

$$\hat{x}_{t+1|t+1} = \left[I - \sum_{i=1}^{M} L_i(t+1) C_i\right] A\hat{x}_{t|t} + \sum_{i=1}^{M} L_i(t+1) y_{t+1}^i \quad (2.9)$$

$$P_{t+1|t+1} = \left[I - \sum_{i=1}^{M} L_i(t+1) C_i\right] P_{t+1|t}\,, \quad (2.10)$$

avendo partizionato opportunamente il guadagno di Kalman $L(t+1)$.

Di conseguenza, analogamente a quanto si trova in [3], possiamo calcolare la stima (2.9) come somma di M "stime" locali:

$$z_{t+1|t+1}^i = \left[I - \sum_{i=1}^{M} L_i(t+1) C_i\right] Az_{t|t}^i + L_i(t+1) y_{t+1}^i\,, \quad (2.11)$$

e la stima centralizzata è data da $\hat{x}_{t+1|t+1} = \sum_{i=1}^{M} z_{t+1|t+1}^i$.

Esattamente come nella soluzione precedente, anche in questo caso la struttura dell'algoritmo suggerisce un approccio di fusione delle variabili locali $z_{t|t}^i$, in modo che ogni nodo invii il dato:

$$\hat{x}_{t+1|t+1}^i = z_{t+1|t+1}^i + \sum_{j \in f(i)} \hat{x}_{t+1|t+1}^j\,, \quad (2.12)$$

dove $f(i)$ è l'insieme dei figli del nodo i-esimo.

Alla radice, la (2.12) fornisce proprio la stima globale ottima. Si noti che qui non si rende necessaria l'ipotesi che i rumori di misura dei sensori siano scorrelati (R diagonale), mentre è necessaria anche in questo caso o una struttura ad albero (ogni nodo può avere un solo padre) o la memorizzazione degli indirizzi dei sensori che hanno inviato il proprio dato, in modo da non sommare due volte la stessa variabile.

Mentre nell'algoritmo 1 i nodi per calcolare i dati da trasmettere non devono conoscere altro se non le proprie matrici C_i e R_i, in questo caso ogni nodo deve avere in memoria la matrice di osservazione completa C e la sequenza di matrici $L(t)$ (calcolabili offline).

Sembra quindi che questa sia una soluzione un pò più dispendiosa, ma nell'ottica di estendere le equazioni (2.9), (2.10) al caso con perdita di pacchetto, si può pensare invece che l'invio di una stima possa dare risultati migliori che l'invio di singole misure, dal momento che le stime contengono tutta l'informazione relativa alle osservazioni passate.

[2] Si veda ad esempio [4], p. 302.

2.3.2.3 Algoritmo 3

In letteratura è stato ampiamente trattato il problema della stima distribuita, ed in particolare è stato dimostrato che sotto opportune ipotesi è possibile esprimere la stima globale ottima (2.3) in funzione di tutte le stime di Kalman locali. Sfruttando queste ipotesi e partendo da un'algoritmo iterativo noto che calcola la stima centralizzata in funzione di quelle locali, analizzeremo come questo si adatti ad un approccio di fusione delle stime locali diverso da quelli finora trattati.

Sia $\tilde{x}^i_{t|t}$ la stima di Kalman effettuata dal sensore i-esimo sulla base si tutte le osservazioni $\left\{y^i_t, \ldots, y^i_0\right\}$ e $\tilde{P}^i_{t|t}$ la varianza dell'errore di stima $x_t - \tilde{x}^i_{t|t}$.

Sia inoltre $\hat{x}_{t|t} = \mathbb{E}\left[x_t \mid y_t, \ldots y_0\right]$ la stima ottima centralizzata, calcolabile con le note equazioni del filtro di Kalman, e $P_{t|t}$ la relativa varianza d'errore.

Quando i rumori di misura v^i_t e v^j_t sono scorrelati $\forall i \neq j$, è possibile dimostrare che la stima ottima globale $\hat{x}_{t|t}$ si può calcolare in funzione delle M stime locali, come segue[3]:

$$\hat{x}_{t+1|t+1} = P_{t+1|t+1}\left[P^{-1}_{t+1|t}\hat{x}_{t+1|t} + \sum_{i=1}^{M}\left(\tilde{P}^{i^{-1}}_{t+1|t}\tilde{x}^i_{t+1|t+1} - \tilde{P}^{i^{-1}}_{t+1|t}\tilde{x}^i_{t+1|t}\right)\right] \quad (2.13)$$

$$P^{-1}_{t+1|t+1} = P^{-1}_{t+1|t} + \sum_{i=1}^{M}\left(\tilde{P}^{i^{-1}}_{t+1|t+1} - \tilde{P}^{i^{-1}}_{t+1|t}\right). \quad (2.14)$$

Se tutti i sensori inviassero le proprie stime di Kalman alla radice, la base station potrebbe calcolare la stima centralizzata ottima utilizzando le equazioni (2.13) e (2.14).

Allora, si può pensare che una tecnica efficiente di fusione dei dati sia quella che utilizza queste equazioni ad ogni nodo della rete, in modo tale che il generico sensore i-esimo trasmetta la stima $\hat{x}^i_{t|t} = \mathbb{E}\left[x_t \mid \tilde{x}^i_{t|t}, \tilde{x}^j_{t|t}, \forall j \in d\left(i\right)\right]$, cioè la stima ottima date le stime di Kalman del nodo i stesso e di tutti i nodi j discendenti di i, ossia dei nodi appartenenti al sottoalbero di cui i è la radice.

Quindi, i nodi terminali calcolano e trasmettono la stima di Kalman classica, mentre tutti gli altri nodi dovrebbero inviare al proprio genitore la stima:

$$\begin{aligned}\hat{x}^i_{t+1|t+1} = {} & P^i_{t+1|t+1}\left[P^{i^{-1}}_{t+1|t}\hat{x}^i_{t+1|t} + \left(\tilde{P}^{i^{-1}}_{t+1|t+1}\tilde{x}^i_{t+1|t+1} - \tilde{P}^{i^{-1}}_{t+1|t}\tilde{x}^i_{t+1|t}\right)\right] + \\ & + P^i_{t+1|t+1}\sum_{j\in d(i)}\left(\tilde{P}^{j^{-1}}_{t+1|t+1}\tilde{x}^j_{t+1|t+1} - \tilde{P}^{j^{-1}}_{t+1|t}\tilde{x}^j_{t+1|t}\right) \quad (2.15)\end{aligned}$$

$$P^{i^{-1}}_{t+1|t+1} = P^{i^{-1}}_{t+1|t} + \left(\tilde{P}^{i^{-1}}_{t+1|t+1} - \tilde{P}^{i^{-1}}_{t+1|t}\right) + \sum_{j\in d(i)}\left(\tilde{P}^{j^{-1}}_{t+1|t+1} - \tilde{P}^{j^{-1}}_{t+1|t}\right). \quad (2.16)$$

Affinchè queste equazioni possano suggerire un nuovo metodo di fusione dei dati, occorre riscriverle in una forma appropriata.[4] E' immediato provare che le equazioni appena introdotte, possono essere riscritte come segue:

[3] Si vedano [6] e [8].

[4] Si noti che per come sono scritte ora, queste stime sono calcolate sulla base di dati che ogni nodo deve ricevere da tutti i suoi discendenti e non solo dai figli.

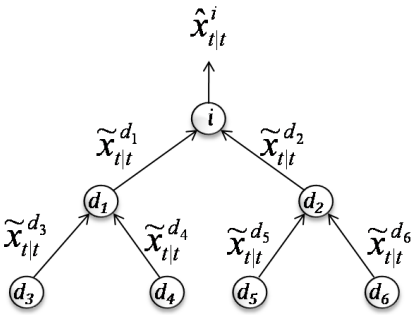

Figura 2.1: Rappresentazione delle eq. (2.15) e (2.16) per un generico nodo i, radice del sottoalbero con nodi $d_1, \ldots, d_6$.

$$\begin{aligned}
\hat{x}^i_{t+1|t+1} &= P^i_{t+1|t+1}\left[P^{i^{-1}}_{t+1|t}\hat{x}^i_{t+1|t} + \left(\tilde{P}^{i^{-1}}_{t+1|t+1}\tilde{x}^i_{t+1|t+1} - \tilde{P}^{i^{-1}}_{t+1|t}\tilde{x}^i_{t+1|t}\right)\right] + \\
&\quad + P^i_{t+1|t+1}\sum_{j\in f(i)}\left(P^{j^{-1}}_{t+1|t+1}\hat{x}^j_{t+1|t+1} - P^{j^{-1}}_{t+1|t}\hat{x}^j_{t+1|t}\right) \qquad (2.17)\\
P^{i^{-1}}_{t+1|t+1} &= P^{i^{-1}}_{t+1|t} + \left(\tilde{P}^{i^{-1}}_{t+1|t+1} - \tilde{P}^{i^{-1}}_{t+1|t}\right) + \sum_{j\in f(i)}\left(P^{j^{-1}}_{t+1|t+1} - P^{j^{-1}}_{t+1|t}\right), \qquad (2.18)
\end{aligned}$$

dove $f(i)$ è l'insieme dei figli del nodo i.

Dimostrazione. E' sufficiente provare che

$$\sum_{j\in d(i)}\left(\tilde{P}^{j^{-1}}_{t+1|t+1}\tilde{x}^j_{t+1|t+1} - \tilde{P}^{j^{-1}}_{t+1|t}\tilde{x}^j_{t+1|t}\right) = \sum_{j\in f(i)}\left(P^{j^{-1}}_{t+1|t+1}\hat{x}^j_{t+1|t+1} - P^{j^{-1}}_{t+1|t}\hat{x}^j_{t+1|t}\right), \forall i.$$

Se $f_1, \ldots, f_n$ sono gli indici dei nodi figli di i, è evidente che

$$\begin{aligned}
&\textstyle\sum_{j\in d(i)}\left(\tilde{P}^{j^{-1}}_{t+1|t+1}\tilde{x}^j_{t+1|t+1} - \tilde{P}^{j^{-1}}_{t+1|t}\tilde{x}^j_{t+1|t}\right) = \\
&\left(\tilde{P}^{f_1^{-1}}_{t+1|t+1}\tilde{x}^{f_1}_{t+1|t+1} - \tilde{P}^{f_1^{-1}}_{t+1|t}\tilde{x}^{f_1}_{t+1|t}\right) + \textstyle\sum_{j\in d(f_1)}\left(\tilde{P}^{j^{-1}}_{t+1|t+1}\tilde{x}^j_{t+1|t+1} - \tilde{P}^{j^{-1}}_{t+1|t}\tilde{x}^j_{t+1|t}\right) + \ldots \\
&\ldots + \left(\tilde{P}^{f_n^{-1}}_{t+1|t+1}\tilde{x}^{f_n}_{t+1|t+1} - \tilde{P}^{f_n^{-1}}_{t+1|t}\tilde{x}^{f_n}_{t+1|t}\right) + \textstyle\sum_{j\in d(f_n)}\left(\tilde{P}^{j^{-1}}_{t+1|t+1}\tilde{x}^j_{t+1|t+1} - \tilde{P}^{j^{-1}}_{t+1|t}\tilde{x}^j_{t+1|t}\right),
\end{aligned}$$

avendo semplicemente separato tra loro i discendenti di i appartenenti a rami diversi (consideriamo il contributo prima del sottoalbero con radice in f_1, poi di quello con radice in f_2, e così via fino a quello con radice in f_n).

Si osserva ora che i primi due termini sommati a secondo membro forniscono $P^{f_1^{-1}}_{t+1|t+1}\hat{x}^{f_1}_{t+1|t+1} - P^{f_1^{-1}}_{t+1|t}\hat{x}^{f_1}_{t+1|t}$, come si vede facilmente dalla (2.15), e così via sino agli ultimi due addendi a secondo membro che forniscono $P^{f_n^{-1}}_{t+1|t+1}\hat{x}^{f_n}_{t+1|t+1} - P^{f_n^{-1}}_{t+1|t}\hat{x}^{f_n}_{t+1|t}$.

Di conseguenza si ottiene che

$$\sum_{j\in d(i)} \left(\tilde{P}^{j^{-1}}_{t+1|t+1} \tilde{x}^{j}_{t+1|t+1} - \tilde{P}^{j^{-1}}_{t+1|t} \tilde{x}^{j}_{t+1|t} \right) =$$
$$= P^{f_1^{-1}}_{t+1|t+1} \hat{x}^{f_1}_{t+1|t+1} - P^{f_1^{-1}}_{t+1|t} \hat{x}^{f_1}_{t+1|t} + \ldots + P^{f_n^{-1}}_{t+1|t+1} \hat{x}^{f_n}_{t+1|t+1} - P^{f_n^{-1}}_{t+1|t} \hat{x}^{f_n}_{t+1|t} ,$$

che è la tesi.

In modo del tutto analogo si prova che la (2.16) si può riscrivere come la (2.18). ▷

Quindi, le equazioni (2.17), (2.18) forniscono un metodo ottimo di fusione delle stime locali ad ogni nodo; si osservi che la stima $\hat{x}^i_{t|t}$ viene calcolata come se di fatto il nodo i fosse la radice di un albero di profondità 1 (i figli di i raccolgono le informazioni di tutti gli altri nodi discendenti di i, come si vede in Figura 2.2). E' evidente che alla radice si ottiene proprio la stima ottima data dalla (2.13).

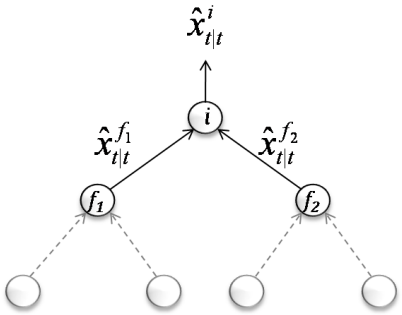

Figura 2.2: Rappresentazione delle eq. (2.17) e (2.18) per un generico nodo i, i cui figli sono f_1e f_2.

Si osservi la sostanziale differenza di questo metodo di fusione rispetto ai due sopra discussi: qui ogni nodo trasmette una vera e propria stima della variabile x_t, mentre le (2.8) e (2.12) (in particolare la (2.8)) sono più propriamente delle funzioni delle misure, che opportunamente rielaborate forniscono la stima globale.

Va osservato, infine, che per questa tecnica di fusione è davvero necessario che la struttura della rete sia ad albero, perchè la stima di ogni sensore deve essere utilizzata una sola volta.

2.4 Analisi e simulazione di algoritmi di stima distribuita con perdita di pacchetto

Nella precedente sezione sono state presentate alcune tecniche di fusione dei dati, che consentono di ottenere alla base station la stima ottima globale.

In questa sezione, invece, supporremo che ogni sensore trasmetta i suoi dati direttamente alla base station senza occuparsi della fusione, trascurando quindi il problema del traffico in rete. Anche in queste condizioni, non è affatto banale trovare un metodo di stima ottimo che tenga conto della perdita di pacchetto.

In letteratura è stato affrontato il problema nel caso semplice di un solo sensore che trasmette i dati alla base station (si veda [7]), ed è risultato che in questo caso la soluzione ottima è che il sensore invii in ogni istante le proprie stime di Kalman. In questo modo, quando la base station non riceve la stima del sensore all'istante corrente, fa evolvere in catena aperta il filtro di Kalman, a partire dall'ultimo dato ricevuto. Questa soluzione ha due caratteristiche fondamentali: anzitutto, se in un certo istante la base station riceve la stima dal sensore è come se avesse ricevuto tutte le misure fino a quell'istante; in secondo luogo, quando un dato non arriva, viene comunque sfruttata tutta l'informazione disponibile (ossia tutte le misure arrivate fino all'ultimo istante di ricezione) e la prestazione è decisamente migliore.

Il problema rimane per il momento irrisolto nel caso più generale di reti con due o più sensori: non si è ancora in grado di dire quali funzioni delle misure passate vadano implementate ai nodi in modo tale che la ricezione alla base station di un dato di questo tipo consenta il recupero di tutta l'informazione precedentemente persa, analogamente a quanto avviene nel caso di un solo sensore che invia le proprie stime di Kalman.

Per approfondire questi aspetti, in questa sezione si generalizzeranno al caso con perdita di pacchetto gli algoritmi di stima proposti nella sezione precendente (come già detto, non si considereranno gli aspetti di aggregazione o fusione dei dati), e verranno effettuate alcune simulazioni per l'analisi ed il confronto delle prestazioni.

2.4.1 Algoritmi per la stima distribuita in presenza di perdita di pacchetto

Consideriamo anzitutto il caso in cui la base station riceva dai sensori delle stime (o variabili) locali. Analogamente a quanto avviene nel caso di un solo sensore, si può pensare che un modo efficiente di utilizzare le stime presenti alla base station in caso di perdita di pacchetti sia quello di aggiornare le stime con il filtro di Kalman in catena aperta. Seguendo questo principio, vengono qui proposte alcune possibli generalizzazioni degli algoritmi trattati in 2.3.2.2 (eq. (2.9)-(2.10)) e 2.3.2.3 (eq. (2.13)-(2.14)), che calcolano la stima globale sulla base di stime locali effettuate dai sensori.[5]

Per quanto riguarda la stima basata sulle misure trasmesse dai sensori, in [1] è dimostrato che la soluzione ottima in caso di perdita di pacchetti è il filtro di Kalman date tutte le osservazioni ricevute alla base station.

Infine, in [5] si può trovare una generalizzazione dell'algoritmo visto in 2.3.2.1 al caso con perdita di pacchetto, che tuttavia non verrà qui implementata, in

[5]Si parla di stime locali anche se abbiamo già evidenziato che l'algoritmo 2 in 2.3.2.2 utilizza, più propriamente, delle generiche variabili locali.

quanto è provato che ha le stesse prestazioni della soluzione che utilizza le misure dei sensori.[6]

Si consideri la variabile $\gamma_t^i = \begin{cases} 0 & \textit{se pacchetto non arrivato} \\ 1 & \textit{se pacchetto arrivato} \end{cases}$ che descrive il processo di arrivi dei pacchetti inviati dal nodo i alla base station e sia $\gamma_t = \left(\gamma_t^1, \ldots \gamma_t^M\right)$. In fase di simulazione si considererà γ_t^i bernoulliana i.i.d. con $\lambda = \mathrm{P}\left[\gamma_t^i = 1\right]$.

2.4.1.1 Algoritmo 1

Una prima soluzione è quella che implementa le equazioni (2.13), (2.14), utilizzando però, in caso di mancata ricezione di una stima locale, la stima ottenuta in catena aperta alla base station; le varianze d'errore dei filtri locali rimangono quelle calcolate a priori (quindi come se non ci fosse perdita di pacchetti).

Le equazioni diventano le seguenti:

$$\begin{aligned} \hat{x}_{t+1|t+1} &= P_{t+1|t+1}\left[P_{t+1|t}^{-1}\hat{x}_{t+1|t} + \sum_{i=1}^{M}\left(\tilde{P}_{t+1|t+1}^{i^{-1}}\bar{x}_{t+1|t+1}^i - \tilde{P}_{t+1|t}^{i^{-1}}\bar{x}_{t+1|t}^i\right)\right] \\ P_{t+1|t+1}^{-1} &= P_{t+1|t}^{-1} + \sum_{i=1}^{M}\left(\tilde{P}_{t+1|t+1}^{i^{-1}} - \tilde{P}_{t+1|t}^{i^{-1}}\right) , \end{aligned}$$

con

$$\bar{x}_{t+1|t}^i = \gamma_t^i\tilde{x}_{t+1|t}^i + \left(1-\gamma_t^i\right)A\bar{x}_{t|t-1}^i \tag{2.19}$$

$$\bar{x}_{t+1|t+1}^i = \gamma_{t+1}^i\tilde{x}_{t+1|t+1}^i + \left(1-\gamma_{t+1}^i\right)\bar{x}_{t+1|t}^i , \tag{2.20}$$

dove $\tilde{x}_{t|t}^i$ è la stima di Kalman del sensore i e $\tilde{P}_{t|t}^i$ la relativa varianza d'errore.

Si osservi che rispetto all'equazione (2.13), vengono qui sostituite le stime locali con $\bar{x}_{t+1|t+1}^i$ e $\bar{x}_{t+1|t}^i$: la base station utilizza le stime locali quando le arrivano, altrimenti calcola la stima non arrivata col filtro in catena aperta a partire dall'ultimo dato ricevuto. Quindi, questo metodo ha il vantaggio di sfruttare l'informazione diponibile alla base station, anche se relativa a tempi passati.

2.4.1.2 Algoritmo 2

Si può pensare che un metodo di stima ancora più efficiente sia quello che considera la perdita di pacchetto anche nell'aggiornamento delle varianze d'errore. Analogamente a quanto fatto nell'algoritmo precedente, quando c'è perdita di pacchetto si fa iterare il filtro di Kalman in catena aperta, sia per l'aggiornamento delle stime che per quello delle varianze locali, secondo:

[6] Si vedano solo cenni teorici in Appendice B.

$$\begin{aligned}\hat{x}_{t+1|t+1} &= P_{t+1|t+1}\left[P^{-1}_{t+1|t}\hat{x}_{t+1|t} + \sum_{i=1}^{M}\left(\bar{P}^{i^\dagger}_{t+1|t+1}\bar{x}^i_{t+1|t+1} - \bar{P}^{i^\dagger}_{t+1|t}\bar{x}^i_{t+1|t}\right)\right]\\ P^{-1}_{t+1|t+1} &= P^{-1}_{t+1|t} + \sum_{i=1}^{M}\left(\bar{P}^{i^\dagger}_{t+1|t+1} - \bar{P}^{i^\dagger}_{t+1|t}\right),\end{aligned}$$

$$\begin{aligned}\bar{P}^i_{t+1|t} &= \gamma^i_t\tilde{P}^i_{t+1|t} + \left(1-\gamma^i_t\right)\left(A\bar{P}^i_{t|t-1}A' + Q\right)\\ \bar{P}^i_{t+1|t+1} &= \gamma^i_{t+1}\tilde{P}^i_{t+1|t+1} + \left(1-\gamma^i_{t+1}\right)\bar{P}^i_{t+1|t},\end{aligned}$$

mentre le $\bar{x}^i_{t+1|t+1}$ e $\bar{x}^i_{t+1|t}$ sono date sempre dalle (2.20), (2.19).[7]

La differenza tra questi due algoritmi è che in questa seconda soluzione le varianze d'errore dei filtri locali vengono aggiornate alla base station a seconda dell'arrivo o meno delle stime, quindi è una completa generalizzazione delle (2.13), (2.14) al caso con perdita di pacchetto.

Si osservi che in entrambe le soluzioni si è considerato che i sensori trasmettano $\tilde{x}^i_{t+1|t}$ e $\tilde{x}^i_{t|t}$,$\forall i$ all'istante t.

2.4.1.3 Algoritmo 3

Consideriamo ora l'algoritmo discusso nel paragrafo 2.3.2.2. Una generalizzazione della (2.9) al caso con perdita di pacchetto consiste, analogamente a quanto fatto nelle precedenti due soluzioni, nel far evolvere in catena aperta il filtro a partire dall'ultimo dato arrivato.

Le equazioni diventano le seguenti:

$$\begin{aligned}\hat{x}_{t|t} &= \sum_{i=1}^{M}\bar{z}^i_{t|t}\\ \bar{z}^i_{t|t} &= \gamma^i_t z^i_{t|t} + \left(1-\gamma^i_t\right)A\bar{z}^i_{t-1|t-1},\end{aligned}$$

dove $z^i_{t|t}$ è data dalla (2.11).

Come l'algoritmo visto in 2.4.1.1, anche questo non prevede l'aggiornamento della varianza d'errore in funzione della sequenza di arrivo dei pacchetti, ma aggiorna solo la $\bar{z}^i_{t|t}$, perciò ci si aspetta una prestazione simile a quella della prima soluzione proposta.

2.4.1.4 Algoritmo 4

Infine, consideriamo anche l'algoritmo di stima che utilizza direttamente le misure effettuate dai sensori.

[7] Si utilizza l'operatore di pseudo-inversa dato che, a seconda del valore della variabile γ^i_t, le matrici $\bar{P}^i_{t|t}$ e $\bar{P}^i_{t+1|t}$ potrebbero essere singolari.

Sia $\tilde{y}_t = \left(\tilde{y}_t^1, \ldots, \tilde{y}_t^M\right) = \left(\gamma_t^1 y_t^1, \ldots, \gamma_t^M y_t^M\right)$ il vettore di dati arrivati nell'istante t alla base station (se un dato non arriva viene considerato nullo). Allora, è noto[8] che il miglior stimatore $\hat{x}_{t|t} = \mathbb{E}\left[x_t \mid \tilde{y}_t, \ldots, \tilde{y}_0, \gamma_t, \ldots, \gamma_0\right]$, è dato dal filtro di Kalman con equazioni (2.4), (2.5) e

$$\begin{aligned} \hat{x}_{t+1|t+1} &= \hat{x}_{t+1|t} + L\left(t+1\right)\left[\tilde{y}_{t+1} - C_{t+1}\hat{x}_{t+1|t}\right] \\ P_{t+1|t+1} &= P_{t+1|t} - P_{t+1|t}C_{t+1}'\Lambda\left(t+1\right)^{\dagger} C_{t+1}P_{t+1|t}\,, \end{aligned}$$

dove

$$\Lambda\left(t\right) = C_{t+1}P_{t|t-1}C_{t+1}' + R_{t+1}\,, \qquad L\left(t\right) = P_{t|t-1}C_{t+1}'\Lambda\left(t\right)^{\dagger}\,,$$

con $C_t' = \left[\begin{array}{ccc} \gamma_t^1 C^{1'} & \cdots & \gamma_t^M C^{M'} \end{array}\right]$ e $\left[R_t\right]_{ij} = \gamma_t^i \gamma_t^j R_{ij},\ i,j = 1, \ldots m.$

Va osservato che mentre per quest'ultimo algoritmo la $P_{t|t}$ rappresenta la varianza d'errore, negli altri casi non è così: quando c'è perdita di pacchetti, le $P_{t|t}$ calcolate negli altri algoritmi *non* sono varianze d'errore (sono state adattate le equazioni viste nella precedente sezione al caso con perdita di pacchetti, ma il significato delle $P_{t|t}$ non è più lo stesso). Per questa ragione, in fase di analisi delle prestazioni si ricorrerà al calcolo delle varianze d'errore campionarie.

2.4.2 Simulazioni: analisi delle prestazioni al variare del rapporto $\frac{Q}{R}$

Uno dei parametri che maggiormente può influire sulle prestazioni degli algoritmi sopra discussi è il rapporto tra la varianza del rumore di modello e quella del rumore di misura. Infatti, è chiaro che le prestazioni degli algoritmi 1, 2 e 3, che utilizzano il filtro di Kalman in catena a perta, peggiorano all'aumentare del rumore di modello, e quelle dell'algoritmo 4, all'aumentare del rumore di misura. Per questo, sono state effettuate alcune simulazioni nel caso di una rete con soli due sensori e di un modello del processo di osservazione di dimensione uno.

Per il confronto delle prestazioni si fa riferimento alle medie a regime delle varianze campionarie d'errore: come noto, l'evoluzione nel tempo delle varianze d'errore in presenza di perdita di pacchetto è di tipo oscillatorio (in quanto dipende dall'arrivo o meno dei pacchetti), ma la media tende, sotto opportune ipotesi, ad un valore di regime. Per questo motivo, le grandezze confrontate nei grafici di seguito in funzione del rapporto q/r, sono le medie a regime delle varianze campionarie calcolate per una data sequenza del processo di arrivi γ_t attraverso 40 simulazioni (sono cioè state generate per ogni algoritmo 40 traiettorie di osservazione e le relative 40 traiettorie di stima alla base station, per una data sequenza di arrivi).

[8] Si veda ad esempio [1]; si evidenzia che stiamo trascurando ritardi aleatori, e considerando solo perdita di pacchetto.

Sia γ_t^i un processo a variabili bernoulliane i.i.d. e sia $\lambda = \mathbb{P}\left[\gamma_t^i = 1\right]$; sia inoltre c una costante moltiplicativa dei valori assoluti delle varianze q e r. Sono stati analizzati cinque casi particolari per valutare le prestazioni degli algoritmi anche in funzione della probabilità λ di arrivo dei pacchetti, della stabilità del processo di misura (cioè della matrice A) e dei valori assoluti di q e r (in questo caso scalari):

1. $A = 0.5$, $\lambda = 0.5$, $c = 1$
2. $A = 1$, $\lambda = 0.5$, $c = 1$
3. $A = 1$, $\lambda = 0.5$, $c = 10$
4. $A = 1$, $\lambda = 0.85$, $c = 1$
5. $A = 2$, $\lambda = 0.85$, $c = 1$

Nel caso 5, si considera λ sufficientemente grande, in modo che lo stimatore di Kalman visto in 2.4.1.4 sia stabile.[9]

Si è data particolare attenzione al caso con $A = 1$ perchè per questo si riescono a confrontare più chiaramente le prestazioni dei vari algoritmi.

Per semplicità sono stati utilizzati gli stessi dati per i due sensori, quindi la stessa matrice di osservazione e lo stesso valore iniziale della traiettoria di stato; essi sono:

$$C_1 = C_2 = 1$$
$$P_0 = 1$$
$$\bar{x}_0 = 0$$
$$x_0^1 = x_0^2 = \mathcal{N}\left(\bar{x}_0, P_0\right) .$$

Dato che le prestazioni degli algoritmi si differenziano al variare *combinato* dei parametri q e r, questi sono stati variati indipendentemente, utilizzando i comandi Matlab

$$r = c \star logspace(2, 0, 50)$$
$$q = c \star logspace(-3, 0.3, 50)$$

che generano vettori in scala logaritmica, in modo da ottenere una variazione uniforme del rapporto q/r nell'intervallo $[0, 2]$ di interesse.

La lunghezza (intesa come numero di campioni) delle traiettorie di osservazione e di stima generate è $T = 100$.

Le varianze d'errore relative agli algoritmi 1, 2, 3 e 4 sono indicate nei grafici rispettivamente con *var_sbs1*, *var_sbs2*, *var_sbs3* e *var_mbs*.

In Figura 2.3 sono rappresentati gli andamenti delle varianze d'errore in funzione del rapporto q/r relativi al caso 1. Si osservi che per $q/r \backsimeq 0$ le varianze tendono ad avere tutte uno stesso andamento ed aumentano al crescere di q/r.

[9] Si veda [7].

Si può dire che complessivamente gli algoritmi 2 e 4 hanno prestazioni migliori degli altri due, tanto più quanto più q/r è grande.

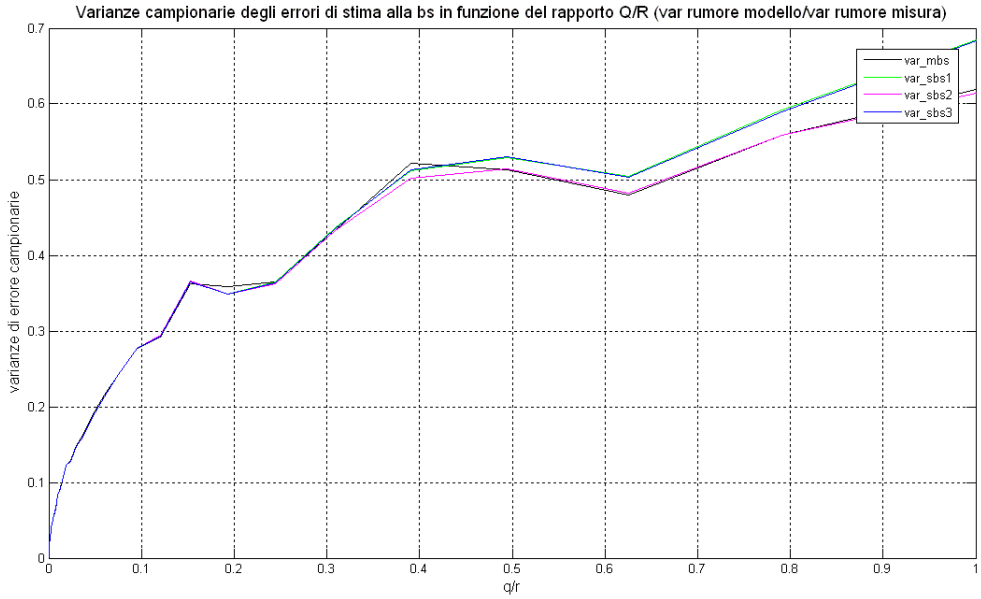

Figura 2.3: Varianze campionarie medie in funzione di $\frac{q}{r}$ nel caso 1.

Nel caso 2 con $A = 1$, si riesce ad apprezzare la differenza di prestazioni degli algoritmi 1, 2, 3 rispetto all'algoritmo 4, che utilizza le misure trasmesse dai sensori. Si può osservare, infatti, che per $q/r \lesssim 0.5$ gli algoritmi che utilizzano le stime locali hanno tutti prestazioni migliori dell'algoritmo 4. Al crescere di q/r, le varianze d'errore relative agli algoritmi 1 e 3 aumentano, mentre quella dell'algoritmo 2 è sempre minore di tutte le altre, risultando perciò quest'ultimo l'algoritmo migliore.

In Figura 2.5 si osservi che per valori di q e r 10 volte maggiori si ha un notevole aumento delle varianze d'errore di tutti gli algoritmi, rispetto ai valori in Figura 2.4; anche in questo caso, per q/r piccolo ($q/r \lesssim 0.2$) la soluzione 4 ha prestazioni peggiori delle altre, mentre al crescere di q/r si avvicina alla prestazione dell'algoritmo 2 che complessivamente risulta il più efficiente, anche in questo caso.

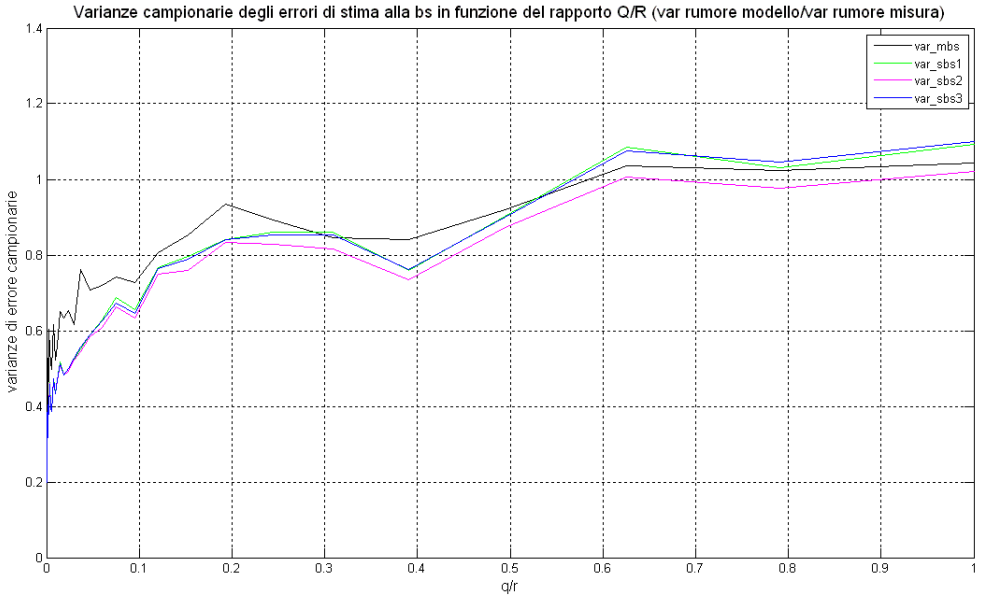

Figura 2.4: Varianze campionarie medie in funzione di $\frac{q}{r}$ nel caso 2.

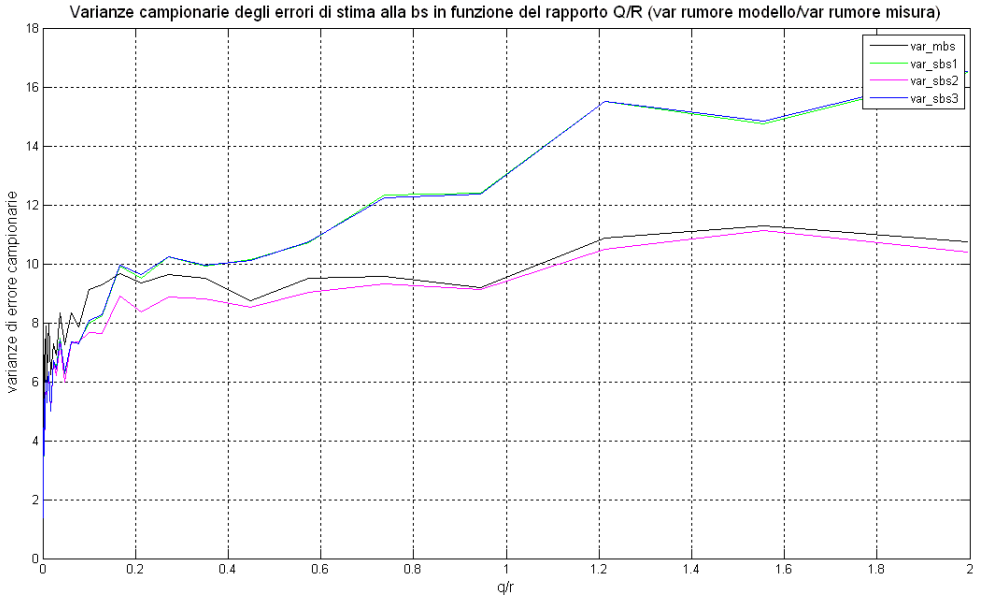

Figura 2.5: Varianze campionarie medie in funzione di $\frac{q}{r}$ nel caso 3.

Si consideri ora il caso 4: come si può vedere dal confronto dei grafici in Figu-

ra 2.4 e 2.6, con l'aumento della probabilità di arrivo dei pacchetti le prestazioni di tutti gli algoritmi migliorano. Inoltre, al crescere di q/r le varianze non tendono ad aumentare, come nei casi precedenti (in particolar modo quelle degli algoritmi 2 e 4). Si osservi che, come negli altri casi analizzati, gli algoritmi 1 e 3 hanno prestazioni molto simili tra loro, e così anche gli algoritmi 2 e 4, per valori di q/r sufficientemente grandi.

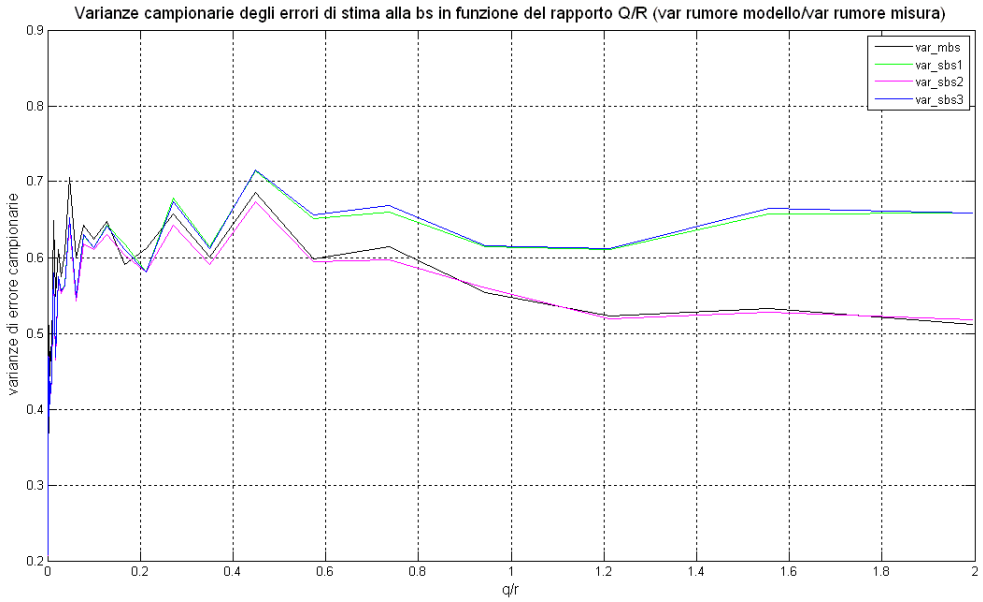

Figura 2.6: Varianze campionarie medie in funzione di $\frac{q}{r}$ nel caso 4.

Infine, si consideri il caso in cui A sia instabile. I grafici in Figura 2.7 mostrano che l'instabilità di A fa sì che le varianze d'errore degli algoritmi 1, 2 e 3 divergano nel tempo, mentre quella dell'algoritmo 4 è sempre finita (come è noto dalla teoria sul filtro di Kalman con perdita di pacchetto; si veda [7]). In questo caso, la soluzione migliore è ovviamente quella che utilizza le misure inviate dai sensori.

In conclusione, si può affermare che in tutti i casi dal primo al quarto, gli algoritmi 1 e 3 hanno uguali prestazioni, indipendentemente dal valore del rapporto q/r, e per q/r piccolo risultano più efficienti dell'algoritmo 4; in effetti, se $q \ll r$ significa che l'aggiornamento della stima con il filtro in catena aperta è basata su un modello praticamente esatto del processo di stato x_t e quindi avrà varianza d'errore minore della varianza relativa al filtro di Kalman che utilizza misure, invece, molto rumorose. Al crescere di q/r, l'algoritmo 4 ha prestazioni sempre migliori che, come già osservato, si avvicinano in molti casi a quelle dell'algoritmo 2. Dalle simulazioni effettuate risulta che la soluzione più efficiente è proprio la seconda, la quale aggiorna sia le stime che le varianze d'errore locali in catena aperta al mancato arrivo dei pacchetti. Le prestazioni

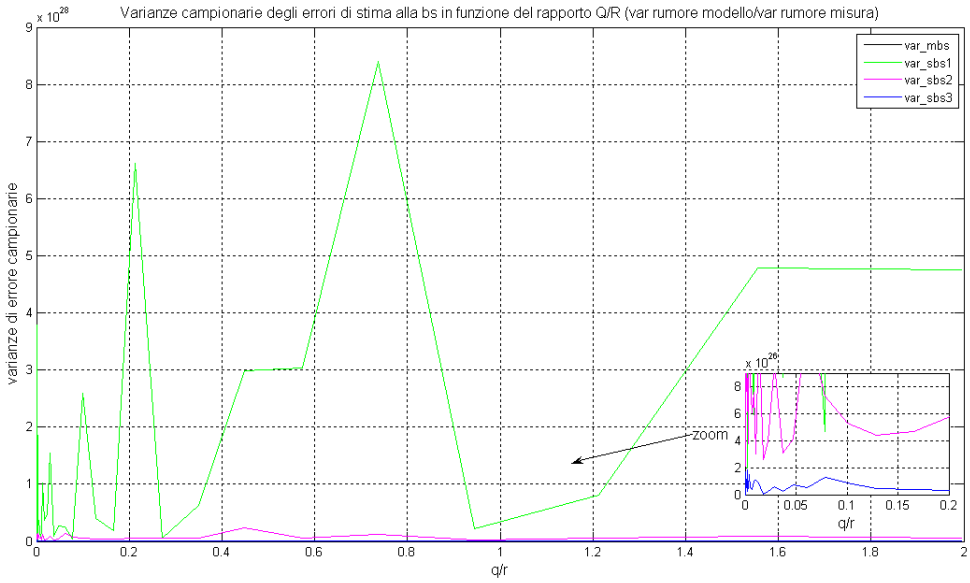

Figura 2.7: Varianze campionarie medie in funzione di $\frac{q}{r}$ nel caso 5.

di quest'algoritmo, infatti, sono uguali a quelle degli algoritmi 1 e 3 per q/r piccolo (tipicamente $q/r \lesssim 0.1$) e uguali (se non migliori) a quelle dell'algoritmo 4 per q/r grandi (tipicamente $q/r \gtrsim 1$); complessivamente, per qualsiasi valore di q/r considerato, questa risulta la soluzione migliore.

Si noti che tutte queste soluzioni si prestano ad essere implementate con un approccio di tipo aggregativo. Va però osservato che in questo modo se il dato trasmesso da un nodo j al padre viene perso, vengono automaticamente persi tutti i dati provenienti da quel ramo della rete, e questo certamente influisce sulle prestazioni degli algoritmi di stima; anche per questo motivo, la scelta dell'algoritmo di stima dipende fortemente dal protocollo di comunicazione utilizzato e dal grafo di connettività della rete.

Infine, è stata effettuata un'ulteriore simulazione nel caso 2, per una rete con 4 sensori. Come si vede dai grafici in Figura 2.8, l'andamento qualitativo delle varianze è lo stesso che si trova in Figura 2.4; si può osservare, però, che con un numero maggiore di sensori il valore critico di q/r per il quale le prestazioni dell'algoritmo 4 diventano migliori di quelle degli algoritmi 1 e 3, è minore.

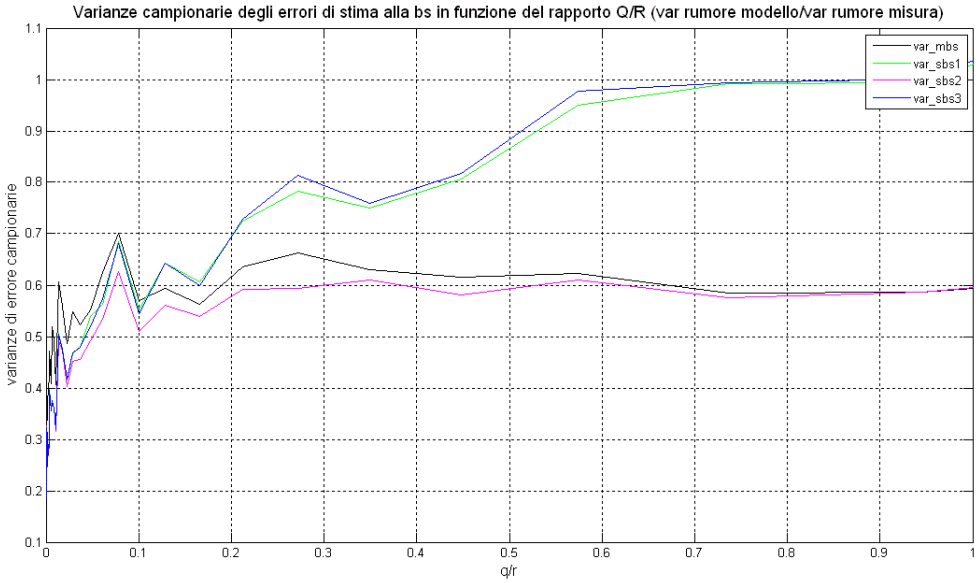

Figura 2.8: Varianze campionarie medie in funzione di $\frac{q}{r}$ nel caso 2, per una rete con 4 sensori.

2.5 Conclusioni

In questo lavoro è stato affrontato il problema della stima distribuita in due aspetti diversi: la fusione dell'informazione, necessaria per la riduzione del traffico in rete, e la perdita di pacchetto negli algoritmi in cui la stima globale è funzione di stime locali elaborate ai sensori.

Nella prima sezione si è ricavato un metodo ottimo per la fusione dei dati, trascurando possibili perdite di pacchetto, ed è stato confrontato con altre tecniche già note in letteratura. Rispetto a queste, il metodo proposto ha il vantaggio di fornire la stima globale come funzione delle stime locali ottime, mentre negli altri casi la stima globale è funzione di variabili locali, le quali di fatto non costituiscono delle stime della grandezza di interesse. Inoltre, questa soluzione consente di disporre ad ogni sensore di una stima "semi-globale", nel senso che il sensore calcola la stima ottima date tutte le misure effettuate dai nodi nel sottoalbero di cui esso è la radice.

Nella seconda sezione, sono state generalizzate al caso con perdita di pacchetto le equazioni (2.13) e (2.9), e sono state effettuate delle simulazioni per l'analisi delle prestazioni di quattro algoritmi in funzione del rapporto tra le varianze dei rumori di modello e di misura. Si è dunque concluso che la seconda soluzione, proposta in 2.4.1.2, risulta essere la più efficiente nei casi in cui la matrice A sia almeno semplicemente stabile, altrimenti, la quarta, proposta in 2.4.1.4, è l'unica a fornire uno stimatore stabile se la probabilità d'arrivo dei pacchetti è sufficientemente alta.

Sulla base dei risultati ottenuti, si ritiene che un ulteriore approfondimento e una fusione di questi problemi possa seguire dall'analisi del terzo metodo di fusione delle stime locali, discusso nella prima sezione (paragrafo 2.3.2.3), in presenza di perdita di pacchetto, secondo l'approccio dell'algoritmo 2 trattato in 2.4.1.2.

Appendix A

Codice Matlab degli algoritmi di stima nel capitolo 1

Riportiamo in questa appendice gli algoritmi di stima descritti nella prima parte di questo lavoro ed utilizzati nella realizzazione della sezione sulle simulazioni e valutazioni degli stimatori. Gli algoritmi sono riportati in codice Matlab sotto forma di funzioni.

Algoritmi di stima con perdita dei pacchetti

- Algoritmo :File KalmanTV.m

```
%----------------------------------
% DESCRIZIONE: calcola il filtro di Kalman ottimo Tempo Variante con
%              perdita di pacchetti
%----------------------------------
% Input:  A,C,Q,R:=matrici di sistema;
%         y:=vettore delle misure
%         gamma:=vettore di arrivo dei pacchetti
%         x0,P0:=stati iniziali del filtro
%
% Output: x,P:=vettore con la stima degli stati e rispettive
%              varianze d'errore di stima
%----------------------------------
function [x,P]=KalmanTV(A,C,Q,R,y,gamma,x0,P0)
T=length(y);
P(:,:,1)=P0;
x(:,1)=x0;
for k=1:T-1
    P(:,:,k+1)=A*P(:,:,k)*A'+ Q - gamma(k)*A*P(:,:,k)*C'*inv(C*P(:,:,k)*C'+R)*C*P(:,:,k)*A';
    K(:,k+1)=P(:,:,k+1)*C'*inv(C*P(:,:,k+1)*C'+R);
    x(:,k+1)=A*x(:,k)+gamma(k+1)*K(k+1)*(y(:,k+1)-C*A*x(:,k));
end
```

- Algoritmo: File KalmanTIMarkov1.m

```
%----------------------------------
% DESCRIZIONE: calcola il filtro di Kalman sub-ottimo Tempo Invariante a
% guadagno costante con perdita di pacchetti a modello di
% Markov.
%----------------------------------
```

```
% Input: A,C,Q,R:=matrici di sistema;
% M:=matrice stocastica della catena di Markov
% y:=vettore delle misure
% gamma:=vettore di arrivo dei pacchetti
% x0,P0:=stati iniziali del filtro
%
% Output: x,P:=vettore con la stima degli stati e rispettive
% varianze d'errore di stima
%----------------------------------
function [x,P]=KalmanTIMarkov1(A,C,Q,R,M,y,gamma,x0,P0)
T=length(y);
I=eye(length(A));
p_greek_1=M(2,1)/[M(1,2)+M(2,1)]; % probabilit a regime di trovarmi nello stato 1
P_MARE=MARE(P0,A,C,Q,R,p_greek_1);
K=P_MARE*C'*inv(C*P_MARE*C'+R); %calcolo del guadagno costante
P(:,:,1)=P0;
x(:,1)=x0;
for k=1:T-1
P(:,:,k+1)=A*[I-gamma(k)*K*C]*P(:,:,k)*[I-gamma(k)*K*C]'*A'+ Q + gamma(k)*A*K*R*K'*A';
x(:,k+1)=A*x(:,k)+gamma(k+1)*K*(y(:,k+1)-C*A*x(:,k));
end
```

- Algoritmo :File KalmanTIMarkov2.m

```
%----------------------------------
% DESCRIZIONE: calcola il filtro di Kalman Tempo Invariante a 2
% guadagni costanti con perdita di pacchetti a modello di
% Markov.
%----------------------------------
% Input: A,C,Q,R:=matrici di sistema;
% M:=matrice stocastica della catena di Markov
% y:=vettore delle misure
% gamma:=vettore di arrivo dei pacchetti
% x0,P0:=stati iniziali del filtro
%
% Output: x,P:=vettore con la stima degli stati e rispettive
% varianze d'errore di stima
%----------------------------------
function [x,P]=KalmanTIMarkov2(A,C,Q,R,M,y,gamma,x0,P0)
T=length(y);
I=eye(length(A));
p_greek(1)=M(2,1)/[M(1,2)+M(2,1)]; % probabilit a regime di trovarmi nello stato 1 o 2
p_greek(2)=M(1,2)/[M(1,2)+M(2,1)];
[t1,t2]=tempiMediPermanenza(M(1,2),M(2,1));
P=ARE(P0,A,C,Q,R);
P_MARE=MARE(P0,A,C,Q,R,p_greek(1));
% determino tramite iterazione sufficientemente lunga l'andamento medio a regime della varianza d'errore P
Pmedio(:,:,1)=P0;
for k=1:10
for u=(k-1)*(round(t1)+round(t2))+1:(k-1)*(round(t1)+round(t2))+round(t1)
Pmedio(:,:,u+1)=A*Pmedio(:,:,u)*A'+ Q - A*Pmedio(:,:,u)*C'*inv(C*Pmedio(:,:,u)*C'+R)*C*Pmedio(:,:,u)*A';
end
for u=(k-1)*(round(t1)+round(t2))+round(t1)+1:k*(round(t1)+round(t2))
Pmedio(:,:,u+1)=A*Pmedio(:,:,u)*A'+ Q;
end
end
Z=length(Pmedio);
for j=1:Z
Pmedio_trace(j)=trace(Pmedio(:,:,j));
Pmare(j)=trace(P_MARE);
end
%ricaviamo le varianze d'errore massime e minime a regime
[Max i_Max] = max(Pmedio_trace);
PmedioMAX=Pmedio(:,:,i_Max);
V=Pmedio_trace(20:Z); %prendiamo solo le traccie a regime (da 20 a M)
[Min i_Min] = min(V);
PmedioMIN=Pmedio(:,:,i_Min+19);
%calcolo del guadagno costante
```

```
% N.B. Il guadagno costante K2 viene utilizzato solamente per la stima con
% il primo pacchetto che arriva dopo un periodo di perdita di pacchetti.
K1 = PmedioMIN*C'*inv(C*PmedioMIN*C'+R);
K2 = PmedioMAX*C'*inv(C*PmedioMAX*C'+R);
P(:,:,1)=P0;
for k=1:T-1
if k>1 & gamma(k-1)==0 & gamma(k)==1
P(:,:,k+1)=A*[I-gamma(k)*K2*C]*P(:,:,k)*[I-gamma(k)*K2*C]'*A'+ Q + gamma(k)*A*K2*R*K2'*A';
else
P(:,:,k+1)=A*[I-gamma(k)*K1*C]*P(:,:,k)*[I-gamma(k)*K1*C]'*A'+ Q + gamma(k)*A*K1*R*K1'*A';
end
end
% Calcolo della stima di x tramite Filtro di Kalman TI modello Markoviano a 2 guadagni K1 eK2
x(:,1)=x0;
for k=1:T-1
if k>1 & gamma(k-1)==0 & gamma(k)==1
x(:,k+1)=A*x(:,k)+gamma(k+1)*K2*(y(:,k+1)-C*A*x(:,k));
else
x(:,k+1)=A*x(:,k)+gamma(k+1)*K1*(y(:,k+1)-C*A*x(:,k));
end
end
```

- Algoritmo :File KalmanDTJMLS.m

```
%-----------------------------------
% DESCRIZIONE: calcola il filtro di Kalman sub-ottimo Tempo Invariante a
%              guadagno costante con perdita di pacchetti a modello di
%              Markov sulla base della teoria dei discrete-time DTJMLS.
%-----------------------------------
% Input:  AA,GG,LL,HH:=matrici stato dipendenti del sistema;
%         Pstocastica:=matrice stocastica della catena di Markov
%         y:=vettore delle misure
%         gamma:=vettore di arrivo dei pacchetti
%         x0,P0:=stati iniziali del filtro
%
% Output: x,P:=vettore con la stima degli stati e rispettive
%              varianze d'errore di stima
%-----------------------------------
function [x,P]=KalmanDTJMLS(AA,GG,LL,HH,Pstocastica,y,gamma,x0,P0)
if(MSS(AA,Pstocastica)==0)
    'Attenzione, il sistema non  MSS.'
end
T=length(y);
I=eye(length(AA(:,:,1)));
N=length(Pstocastica);
A=AA(:,:,1);
C=LL(:,:,1);
Q=GG(:,:,1)*GG(:,:,1)';
R=HH(:,:,1)*HH(:,:,1)';
% probabilit a regime di trovarmi nello stato 1 e 0
p_greek(1)=Pstocastica(2,1)/[Pstocastica(1,2)+Pstocastica(2,1)];
p_greek(2)=Pstocastica(1,2)/[Pstocastica(1,2)+Pstocastica(2,1)];
%Calcolo del guadagno costante K
Y=filtering_CARE(AA,GG,LL,HH,Pstocastica,x0,P0,5);
M=zeros(length(AA(:,:,1)),1);
for i=1:N
    M(:,i)=-AA(:,:,i)*Y(:,:,i)*LL(:,:,i)'*pinv(HH(:,:,i)*HH(:,:,i)'*p_greek(i)+LL(:,:,i)*Y(:,:,i)*LL(:,:,i)');
end
K=-M(:,1);  % N.B. -M(:,1)  l'equivalente di K
% Calcolo della stima di x e della varianza d'errore di stima tramite
% filtro di Kalman TI con approcio JMLS
x(:,1)=x0;
for k=1:T-1
    x(:,k+1)=A*x(:,k)+gamma(k+1)*K*(y(:,k+1)-C*A*x(:,k));
end
P=P0;
for k=1:T-1
    P(:,:,k+1)=A*[I-gamma(k)*K*C]*P(:,:,k)*[I-gamma(k)*K*C]'*A'+ Q + gamma(k)*A*K*R*K'*A';
```

```
end
```

- Algoritmo :File filtering_CARE.m

```
%-----------------------------------
% DESCRIZIONE: calcola la soluzione della filtering-CARE tramite metodo iterativo
%-----------------------------------
% Input:  A, G, L, H:= vedi la definizione 5.7 a pag. 110 del JMLS;
%         P:= matrice stocastica della catena di Markov;
%
% Output: Y:= vettore contenente le N soluzioni della filtering-CARE.
%-----------------------------------
function [YY] = filtering_CARE(A, G, L, H, P, x0, P0,Nfig)
N=length(P);       % = length(A) = length(...) = length(D)
T=100;
p_greek(1)=P(2,1)/[P(1,2)+P(2,1)]; % probabilit a regime di trovarmi nello stato 1 o 2
p_greek(2)=P(1,2)/[P(1,2)+P(2,1)];
%inizializzazione
for j=1:N
    Y(:,:,j,1)=p_greek(j)*(P0-x0*x0');
end
%risoluzione della CARE
for t=1:T
    for j=1:N
        S=zeros(length(A));
        for i=1:N
            S = S + P(i,j)*[A(:,:,i)*Y(:,:,i,t)*A(:,:,i)' +  p_greek(i)*G(:,:,i)*G(:,:,i)'
                - A(:,:,i)*Y(:,:,i,t)*L(:,:,i)'*pinv(p_greek(i)*H(:,:,i)*H(:,:,i)'
                + L(:,:,i)*Y(:,:,i,t)*L(:,:,i)')*L(:,:,i)*Y(:,:,i,t)*A(:,:,i)'];
        end
        Y(:,:,j,t+1)= S;
    end
end
YY=Y(:,:,:,T+1);
%plot della traccia di Y1 e Y2 che convergono nella filtering-CARE
% t = linspace(0,T,T+1);
% figure(Nfig)
% for j=1:T+1
%     Ytrace1(j)=trace(Y(:,:,1,j));
%     Ytrace2(j)=trace(Y(:,:,2,j));
% end
% grid on;
% hold on;
% plot(t, Ytrace1,'g',t,Ytrace2,'b');
% title('Convergenza di Y1 e Y2 nella filtering-CARE');
return
```

Algoritmi di stima con perdita dei pacchetti e ritardo

- Algoritmo :File KalmanTVbufferFinito.m

```
%-----------------------------------
% DESCRIZIONE:calcola il filtro di Kalman Tempo Variante con buffer finito
%-----------------------------------
% Input:  A,C,Q,R:=matrici di sistema;
%         y:=vettore delle misure
%         x0,P0:=stati iniziali del filtro
%         N:=lunghezza del buffer
%         tau:=vettore dei ritardi aleatori dei pacchetti
%
% Output: Xs,Pt:=vettore con la stima degli stati e rispettive
%                varianze d'errore di stima
%-----------------------------------
function [Xs,Pt] = KalmanTVbufferFinito(A,C,Q,R,y,x0,P0,N,tau)
T=length(y);
```

```
%INIZIALIZZAZIONE BUFFER (t=1,...N)
for t=1:N
    for k=1:t
        if k==1
            K1=P0*C'*pinv(C*P0*C'+R);
            X(:,k)=A*x0 + gamma_tk(t,k,tau(k))*K1*(y(:,k)-C*x0);
            P(:,:,k)=A*P0*A'+Q-gamma_tk(t,k,tau(k))*A*K1*C*P0*A';
        else
            K_tk=P(:,:,k-1)*C'*pinv(C*P(:,:,k-1)*C'+R);
            X(:,k)=A*X(:,k-1) + gamma_tk(t,k,tau(k))*K_tk*(y(:,k)-C*X(:,k-1));
            P(:,:,k)=A*P(:,:,k-1)*A'+Q-gamma_tk(t,k,tau(k))*A*K_tk*C*P(:,:,k-1)*A';
        end
    end
    Xs(:,t)=X(:,k);
    Pt(:,:,t)=P(:,:,k);
end
Xt_N=X(:,1);
Pt_N=P(:,:,1);

%BUFFER A REGIME (t=N+1,...T)
for t=N+1:T
    for k=t-N+1:t
         if k==t-N+1
            K_tk=Pt_N*C'*pinv(C*Pt_N*C'+R);
            X(:,k)=A*Xt_N + gamma_tk(t,k,tau(k))*K_tk*(y(:,k)-C*Xt_N);
            P(:,:,k)=A*Pt_N*A'+Q-gamma_tk(t,k,tau(k))*A*K_tk*C*Pt_N*A';
        else
            K_tk=P(:,:,k-1)*C'*pinv(C*P(:,:,k-1)*C'+R);
            X(:,k)=A*X(:,k-1) + gamma_tk(t,k,tau(k))*K_tk*(y(:,k)-C*X(:,k-1));
            P(:,:,k)=A*P(:,:,k-1)*A'+Q-gamma_tk(t,k,tau(k))*A*K_tk*C*P(:,:,k-1)*A';
         end
    end
    Xs(:,t)=X(:,k);
    Pt(:,:,t)=P(:,:,k);
    Xt_N=X(:,t-N+1);
    Pt_N=P(:,:,t-N+1);
end

return
```

- Algoritmo :File KalmanTIbufferFinitoN.m

```
%-----------------------------------
% DESCRIZIONE:calcola il filtro di Kalman Tempo Invariante con buffer finito
%             ad N guadagni costanti
%-----------------------------------
% Input:  A,C,Q,R:=matrici di sistema;
%         y:=vettore delle misure
%         x0,P0:=stati iniziali del filtro
%         lambda:=vettore con le N probabilita' di arrivo
%         tau:=vettore dei ritardi aleatori dei pacchetti
%
% Output: Xs,Pt:=vettore con la stima degli stati e rispettive
%                varianze d'errore di stima
%-----------------------------------
function [Xs,Pt]=KalmanTIbufferFinitoN(A,C,Q,R,y,x0,P0,lambda,tau)
T=length(y);
N=length(lambda);
a=length(A);
I=eye(a);
%Calcolo delle N matrici K costanti##########################################
V(:,:,N)=zeros(a); %determinazione del punto fisso dell'equazione algebrica
for i=2:100        %di Riccati modificata con inizializzazione qualsiasi (P=0)
    V(:,:,N)=A*V(:,:,N)*A'+Q-lambda(N)*A*V(:,:,N)*C'*pinv(C*V(:,:,N)*C'+R)*C*V(:,:,N)*A';
end
K(:,:,N) = V(:,:,N)*C'*pinv(C*V(:,:,N)*C'+R);
%N-1 iterazioni della 1.12
k=N-1;
```

```
while k>=1
    V(:,:,k)=A*V(:,:,k+1)*A'+Q-lambda(k)*A*V(:,:,k+1)*C'*pinv(C*V(:,:,k+1)*C'+R)*C*V(:,:,k+1)*A';
    K(:,:,k)=V(:,:,k)*C'*pinv(C*V(:,:,k)*C'+R);
    k=k-1;
end
%INIZIALIZZAZIONE BUFFER (t=1,...N)#######################################
for t=1:N
    for k=1:t
        if k==1
            X(:,k)=A*x0 + gamma_tk(t,k,tau(k))*K(:,:,k)*(y(:,k)-C*x0);
            P(:,:,k)=A*(I-gamma_tk(t,k,tau(k))*K(:,:,k)*C)*P0*(I-gamma_tk(t,k,tau(k))*K(:,:,k)*C)'*A'
                     + Q + gamma_tk(t,k,tau(k))*A*K(:,:,k)*R*K(:,:,k)'*A';
        else
            X(:,k)=A*X(:,k-1) + gamma_tk(t,k,tau(k))*K(:,:,k)*(y(:,k)-C*X(:,k-1));
            P(:,:,k)=A*(I-gamma_tk(t,k,tau(k))*K(:,:,k)*C)*P(:,:,k-1)*(I-gamma_tk(t,k,tau(k))*K(:,:,k)*C)'*A'
                     + Q + gamma_tk(t,k,tau(k))*A*K(:,:,k)*R*K(:,:,k)'*A';
        end
      end
    Xs(:,t)=X(:,k);
    Pt(:,:,t)=P(:,:,k);
end
Xt_N=X(:,1);
Pt_N=P(:,:,1);
%BUFFER A REGIME (t=N+1,...T)#################################################
for t=N+1:T
    for k=t-N+1:t
         if k==t-N+1
            X(:,k)=A*Xt_N + gamma_tk(t,k,tau(k))*K(:,:,N-t+k)*(y(:,k)-C*Xt_N);
            P(:,:,k)=A*(I-gamma_tk(t,k,tau(k))*K(:,:,N-t+k)*C)*Pt_N*(I-gamma_tk(t,k,tau(k))*K(:,:,N-t+k)*C)'*A'
                     + Q + gamma_tk(t,k,tau(k))*A*K(:,:,N-t+k)*R*K(:,:,N-t+k)'*A';
        else
            X(:,k)=A*X(:,k-1) + gamma_tk(t,k,tau(k))*K(:,:,N-t+k)*(y(:,k)-C*X(:,k-1));
            P(:,:,k)=A*(I-gamma_tk(t,k,tau(k))*K(:,:,N-t+k)*C)*P(:,:,k-1)*(I-gamma_tk(t,k,tau(k))*K(:,:,N-t+k)*C)'*A'
                     + Q + gamma_tk(t,k,tau(k))*A*K(:,:,N-t+k)*R*K(:,:,N-t+k)'*A';
         end
    end
    Xs(:,t)=X(:,k);
    Pt(:,:,t)=P(:,:,k);
    Xt_N=X(:,t-N+1);
    Pt_N=P(:,:,t-N+1);
end

return;
```

- Algoritmo :File KalmanTIbufferFinitoNN.m

```
%-----------------------------------
% DESCRIZIONE:calcola il filtro di Kalman Tempo Invariante con buffer finito
%             di lunghezza N e con N^2 guadagni (cio ad ogni istante t gli
%             N guadagni costanti Ki dipendono dallo stato assunto dalla
%             catena di Markov all'istante t-N+1.
%-----------------------------------
% Input:  A,C,Q,R:=matrici di sistema;
%         y:=vettore delle misure
%         x0,P0:=stati iniziali del filtro
%         M:=matrice stocastica del processo a catena di Markov
%         tau:=vettore dei ritardi aleatori dei pacchetti
%
% Output: Xs,Pt:=vettore con la stima degli stati e rispettive
%                varianze d'errore di stima
%-----------------------------------
function [Xs,Pt]=KalmanTISDfinito(A,C,Q,R,y,x0,P0,M,tau)
T=length(y);
N= length(M);
a=length(A);
I=eye(a);
[p_regime lambda_regime]=regimeP(M);
%##############################################################################
```

```
%Calcolo delle N^2 matrici K costanti ###################################
%#########################################################################
%Calcolo dei vettori lambda|s (raggruppati nella matrice LAMBDA)
LAMBDA=zeros(N,N-1);
% for i=1:N
%     LAMBDA(i,N-1)=lambda_regime(N-1);
% end
for j=1:N-1
    Mj=M^(N-j-1);
    z=0;
    for i=1:N
        for l=1:j
            z=z+Mj(i,l);
        end
        LAMBDA(i,j)=z;
        z=0;
    end
end

% Calcolo degli N^2 guadagni costanti Ki|s (ordinati nella matrice K)
for i=1:N

    V(:,:,N-1)=zeros(a); %determinazione del punto fisso dell'equazione algebrica
    for w=2:100          %di Riccati modificata con inizializzazione qualsiasi (P=0)
        V(:,:,N-1)=A*V(:,:,N-1)*A'+Q-LAMBDA(i,N-1)*A*V(:,:,N-1)*C'*pinv(C*V(:,:,N-1)*C'+R)*C*V(:,:,N-1)*A';
    end
    K(:,:,i,N-1) = V(:,:,N-1)*C'*pinv(C*V(:,:,N-1)*C'+R);
    %N-2 iterazioni
    k=N-2;
    while k>=1
        V(:,:,k)=A*V(:,:,k+1)*A'+Q-LAMBDA(i,k)*A*V(:,:,k+1)*C'*pinv(C*V(:,:,k+1)*C'+R)*C*V(:,:,k+1)*A';
        K(:,:,i,k)=V(:,:,k)*C'*pinv(C*V(:,:,k)*C'+R);
        k=k-1;
    end

end
%#########################################################################
%INIZIALIZZAZIONE BUFFER (t=1,...N-1)#####################################
%#########################################################################
% per l'inizializzazione utilizziamo gli N-1 guadagni costanti K basati sui
% lambda a regime (perci alla fine dobbiamo memorizzare N*(N-1) + (N-1) =
% (N+1)*(N-1) guadagni costanti Ki|s)
%Calcolo delle N-1 matrici K costanti a regime ############################
Vregime(:,:,N-1)=zeros(a); %determinazione del punto fisso dell'equazione algebrica
for i=2:100               %di Riccati modificata con inizializzazione qualsiasi (P=0)
    Vregime(:,:,N-1)=A*Vregime(:,:,N-1)*A'
                     +Q-lambda_regime(N-1)*A*Vregime(:,:,N-1)*C'*pinv(C*Vregime(:,:,N-1)*C'+R)*C*Vregime(:,:,N-1)*A';
end
Kiniz(:,:,N-1) = Vregime(:,:,N-1)*C'*pinv(C*Vregime(:,:,N-1)*C'+R);
k=N-2;
while k>=1
    Vregime(:,:,k)=A*Vregime(:,:,k+1)*A'
                   +Q-lambda_regime(k)*A*Vregime(:,:,k+1)*C'*pinv(C*Vregime(:,:,k+1)*C'+R)*C*Vregime(:,:,k+1)*A';
    Kiniz(:,:,k)=Vregime(:,:,k)*C'*pinv(C*Vregime(:,:,k)*C'+R);
    k=k-1;
end
%inizializzazione del buffer
for t=1:N-1
    for k=1:t
        if k==1
            X(:,k)=A*x0 + gamma_tk(t,k,tau(k))*Kiniz(:,:,k)*(y(:,k)-C*x0);
            P(:,:,k)=A*(I-gamma_tk(t,k,tau(k))*Kiniz(:,:,k)*C)*P0*(I-gamma_tk(t,k,tau(k))*Kiniz(:,:,k)*C)'*A'
                     + Q + gamma_tk(t,k,tau(k))*A*Kiniz(:,:,k)*R*Kiniz(:,:,k)'*A';
        else
            X(:,k)=A*X(:,k-1) + gamma_tk(t,k,tau(k))*Kiniz(:,:,k)*(y(:,k)-C*X(:,k-1));
            P(:,:,k)=A*(I-gamma_tk(t,k,tau(k))*Kiniz(:,:,k)*C)*P(:,:,k-1)*(I-gamma_tk(t,k,tau(k))*Kiniz(:,:,k)*C)'*A'
                     + Q + gamma_tk(t,k,tau(k))*A*Kiniz(:,:,k)*R*Kiniz(:,:,k)'*A';
        end
```

```
    end
    Xs(:,t)=X(:,k);
    Pt(:,:,t)=P(:,:,k);
end
Xt_N=X(:,1);
Pt_N=P(:,:,1);
%#############################################################################
%BUFFER A REGIME (t=N,...T)#####################################################
%#############################################################################
for t=N:T

    if tau(t-N+2)~=inf
        S=tau(t-N+2)+1; %determino lo stato della catena nell'istante t-N+2
    else
        S=1;    %N.B. quando S==N (cio tau=inf)potrei assegnare a S qualsiasi
    end         %     valore tanto gamma_tk(t,k,tau(k))  sempre =0.

    for k=t-N+2:t
         if k==t-N+2
            X(:,k)=A*Xt_N + gamma_tk(t,k,tau(k))*K(:,:,S,t-k+1)*(y(:,k)-C*Xt_N);
            P(:,:,k)=A*(I-gamma_tk(t,k,tau(k))*K(:,:,S,t-k+1)*C)*Pt_N*(I-gamma_tk(t,k,tau(k))*K(:,:,S,t-k+1)*C)'*A'
                    + Q + gamma_tk(t,k,tau(k))*A*K(:,:,S,t-k+1)*R*K(:,:,S,t-k+1)'*A';
        else
            X(:,k)=A*X(:,k-1) + gamma_tk(t,k,tau(k))*K(:,:,S,t-k+1)*(y(:,k)-C*X(:,k-1));
            P(:,:,k)=A*(I-gamma_tk(t,k,tau(k))*K(:,:,S,t-k+1)*C)*P(:,:,k-1)*(I-gamma_tk(t,k,tau(k))*K(:,:,S,t-k+1)*C)'*A'
                    + Q + gamma_tk(t,k,tau(k))*A*K(:,:,S,t-k+1)*R*K(:,:,S,t-k+1)'*A';
        end
    end
    Xs(:,t)=X(:,k);
    Pt(:,:,t)=P(:,:,k);
    Xt_N=X(:,t-N+2);
    Pt_N=P(:,:,t-N+2);
end

return;
```

Appendice B

Filtro di Kalman in forma d'informazione

Il filtro di Kalman in forma d'informazione può essere esteso al caso con perdita di pacchetto utilizzando ancora le variabili γ_t sopra definite. Come si trova in [5], le equazioni diventano:

$$\begin{aligned} \hat{x}_{t+1|t+1} &= P_{t+1|t+1}\left(P^{-1}_{t+1|t}\hat{x}_{t+1|t} + \sum_{i=1}^{M}\gamma_t^i C_i' R_i^{-1} y_t^i\right) \\ P_{t+1|t+1} &= \left(P^{-1}_{t+1|t} + \sum_{i=1}^{M}\gamma_t^i C_i' R_i^{-1} C_i\right) . \end{aligned}$$

Si noti che il dato trasmesso da ogni sensore in questo caso è $\hat{x}_t^i = z_t^i + \sum_{j\in f(i)} \nu_t^j \hat{x}_t^j$, dove ν_t^j è una variabile aleatoria che vale 1 se il pacchetto inviato dal nodo j è arrivato, 0 altrimenti.

Appendice C

Codice Matlab degli algoritmi di stima nel capitolo 2

Di seguito, sono riportate le funzioni Matlab che implementano i quattro algoritmi illustrati nella sezione 2.4.

- Algoritmo 1: File Stima_sbs_1.m

```
%Calcolo della stima e della varianza d'errore alla base-station in
%funzione delle stime locali.
%Quando una stima non arriva il filtro va in catena aperta: utilizza
%l'ultimo dato inviato dal sensore di cui gli
%manca la stima facendo evolvere il filtro locale in catena aperta. Le
%varianze d'errore non vengono modificate, si utilizzano sempre quelle
%calcolate come non ci fosse perdita di pacchetto.
% Av = matrice di stato del sistema
% mu0v = media della condizione iniziale x(0)
% xpsn = predizioni locali (3-dim array)
% xasn = aggiornamenti locali (3-dim array)
% Ppsn = var errori predizione locali
% Pasn = var errori aggiornamento locali
% gam = matrice con sequenze di arrivo per ogni sensore
% xabs = stima ottima globale
function [xabs] = Stima_sbs_1(Av,mu0v,xpsn,xasn,Ppsn,Pasn,Ppbs,Pabs,gam)
xpbs(:,1) = mu0v;
%inizializzazione stima
H = 0;
for ii = 1:size(gam,2) %numero di sensori

    H = H + pinv(gam(1,ii)*Pasn{ii}{1})*(gam(1,ii)*xasn(:,1,ii)+...
    (1-gam(1,ii))*xpsn(:,1,ii)) - pinv(Ppsn{ii}{1})*xpsn(:,1,ii);
    xp_ca(:,ii) = xpsn(:,1,ii);
    xa_ca(:,ii) = xasn(:,1,ii);
```

```
end
xabs(:,1) = Pabs{1}*(pinv(Ppbs{1})*xpbs(:,1) + H);
for k = 1:size(gam,1)-1 %numero di campioni delle stime locali

    H = 0;
    for ii = 1:size(gam,2) %numero di sensori

        %vengono calcolate le stime inviate da ogni sensore;se c'è perdita
        %di pacchetto il filtro va in catena aperta
        %predizione
        xp_ca(:,ii) = gam(k,ii)*xpsn(:,k+1,ii)+(1-gam(k,ii))*Av*xp_ca(:,ii);
        %aggiornamento
        xa_ca(:,ii) = gam(k+1,ii)*xasn(:,k+1,ii)+(1-gam(k+1,ii))*xp_ca(:,ii);
        H = H + pinv(Pasn{ii}{k+1})*xa_ca(:,ii) - pinv(Ppsn{ii}{k+1})*xp_ca(:,ii);
    end
    xpbs(:,k+1) = Av*xabs(:,k);
    xabs(:,k+1) = Pabs{k+1}*(pinv(Ppbs{k+1})*xpbs(:,k+1) + H);

end
```

- Algoritmo 2: File Stima_sbs_2.m

```
%Calcolo della stima e della varianza d'errore alla base-station secondo
%l'algoritmo dall'articolo di Song. Quando una stima non arriva il filtro
%va in catena aperta: utilizza l'ultimo dato inviato dal sensore di cui gli
%manca la stima facendo evolvere il filtro locale in catena aperta. Anche
%le varianze d'errore evolvono in catena aperta.
% Av = matrice di stato del sistema
% Qv = matrice varianza del rumore di modello
% mu0v = media della condizione iniziale x(0)
% P0v = varianza della condizione iniziale x(0)
% xpsn = predizioni locali (3-dim array)
% xasn = aggiornamenti locali (3-dim array)
% Ppsn = var errori predizione locali
% Pasn = var errori aggiornamento locali
% gam = matrice con sequenze di arrivo per ogni sensore
% xabs = stima ottima globale
% Ppbs = è la var d'errore quano non c'è perdita di pacchetto
function [xabs, Ppbs] = Stima_sbs_2(Av,Qv,mu0v,P0v,xpsn,xasn,Ppsn,Pasn,gam)
xpbs(:,1) = mu0v;
Ppbs{1} = P0v;
%inizializzazione stima e varianza d'errore di aggiornamento
Z = 0;
H = 0;
for ii = 1:size(gam,2) %numero di sensori

    Z = Z + pinv(gam(1,ii)*Pasn{ii}{1}+...
    (1-gam(1,ii)*Ppsn{ii}{1}))-pinv(Ppsn{ii}{1});
    H = H + pinv(gam(1,ii)*Pasn{ii}{1})*(gam(1,ii)*xasn(:,1,ii)+...
```

```
    (1-gam(1,ii))*xpsn(:,1,ii))-pinv(Ppsn{ii}{1})*xpsn(:,1,ii);
    xp_ca(:,ii) = xpsn(:,1,ii);
    xa_ca(:,ii) = xasn(:,1,ii);
    Pp_ca{ii} = Ppsn{ii}{1};
    Pa_ca{ii} = Pasn{ii}{1};

end
Pabs{1} = pinv(pinv(Ppbs{1}) + Z);
xabs(:,1) = Pabs{1}*(inv(Ppbs{1})*xpbs(:,1) + H);
for k = 1:size(gam,1)-1 %numero di campioni delle stime locali

    Z = 0;
    H = 0;
    for ii = 1:size(gam,2) %numero di sensori


        %vengono calcolate stime e varianze inviate da ogni sensore;
        se c'è perdita di pacchetto il filtro va in caten aperta
        %predizione
        xp_ca(:,ii) = gam(k,ii)*xpsn(:,k+1,ii)+(1-gam(k,ii))*Av*xp_ca(:,ii);
        %aggiornamento
        xa_ca(:,ii) = gam(k+1,ii)*xasn(:,k+1,ii)+(1-gam(k+1,ii))*xp_ca(:,ii);
        %var errore predizione
        Pp_ca{ii} = gam(k,ii)*Ppsn{ii}{k+1}+(1-gam(k,ii))*(Av*Pp_ca{ii}*Av'+Qv);
        %var errore di stima aggiornata
        Pa_ca{ii} = gam(k+1,ii)*Pasn{ii}{k+1}+(1-gam(k+1,ii))*Pp_ca{ii};
        Z = Z + pinv(Pa_ca{ii}) - pinv(Pp_ca{ii});
        H = H + pinv(Pa_ca{ii})*xa_ca(:,ii) - pinv(Pp_ca{ii})*xp_ca(:,ii);
    end
    xpbs(:,k+1) = Av*xabs(:,k);
    Ppbs{k+1} = Av*Pabs{k}*Av'+Qv;
    Pabs{k+1} = pinv(pinv(Ppbs{k+1}) + Z);
    xabs(:,k+1) = Pabs{k+1}*(pinv(Ppbs{k+1})*xpbs(:,k+1) + H);

end
```

- Algoritmo 3: File Stima_sbs_3.m

```
%Algoritmo per il calcolo della stima centralizzata secondo articolo di
%Speyers:la stima globale è data dalla somma in ogni istante di tutte
%le "stime" locali. In presenza di perdita di pacchetto il filtro va
%in catena aperta.
% Av = matrice di stato
% Cv = matrice del processo di osservazione complessivo alla bs
% Lv = cell-array con guadagni Kalman tempo variante alla bs
% mu0v = media condizione iniziale x(0)
% yv = uscite messe in colonna (l'indice di colonna è il tempo)
% nout = vettore 1xnumsens i cui elementi sono il numero di uscite di ogni
% sensore
% gam = matrice con sequenze di arrivo per sensore
```

```
% z_loc = matrice numstatixnumsensori contenente la stima corrente ai
% sensori
% z_bs = matrice numstatixnumsensori contenente le stime locali inviate dai
% sensori (se arrivate), altrimenti le stime aggiornate alla bs in catena
% aperta
% xabs = somma delle stime locali, ossia stima ottima globale
function [xabs] = Stima_sbs_3(Av,Cv,mu0v,Lv,yv,nout,gam)
z_bs = repmat(mu0v,1,size(gam,2));
z_loc = (mu0v + Lv{1}*(yv(:,1)-Cv*mu0v))/size(gam,2); %è xagg(0)/numsensori
z_loc = repmat(z_loc,1,size(gam,2)); %matrice numstatixnumsensori
Gamma = repmat(gam(1,:),size(Av,1),1);
z_bs = Gamma.*z_loc + (ones(size(Gamma))-Gamma).*(Av*z_bs);
xabs(:,1) = sum(z_bs,2); %se non c'è perdita di pacchetto è xagg(0)
for kk = 1:size(gam,1)-1

    Gamma = repmat(gam(kk+1,:),size(Av,1),1);
    ys = yv(1:nout(1),kk+1);
    gg = 1;
    for ii = 2:size(gam,2) %numero di sensori

        gg = gg + nout(ii-1);
        ys = blkdiag(ys,yv(gg:gg+nout(ii)-1,kk+1));

    end
    %vengono calcolate le stime inviate da ogni sensore;
    %se c'è perdita di pacchetto il filtro va in catena aperta
    % aggiornamento ai sensori (senza perdita di pacchetto)
    z_loc = (Av - Lv{kk+1}*Cv*Av)*z_loc + Lv{kk+1}*ys;
    % aggiornamento del vettore [z_1(kk+1)....z_M(kk+1)]
    z_bs = Gamma.*z_loc + (ones(size(Gamma))-Gamma).*(Av*z_bs);
    %xabs è la somma in ogni istante di tutte le stime giunte dai sensori
    xabs(:,kk+1) = sum(z_bs,2);

end
```

- Algoritmo 4: File Kalmantv_pl_Nsens.m

```
%Funzione che fornisce le stime di Kalman con guadagno tempo variante nel
%caso di perdita di pacchetti; yv è il vettore di dati disponibili per
%effettuare la stima, in cui sono state aggregate tutte le osservazioni
%arrivate alla base station Cv e Qv sono matrici tempo varianti a seconda
%delle successioni delle variabili gamma che descrivono l'arrivo dei
%pacchetti (il calcolo viene effettuato in un file di pre_processing)
% Av = matrice di stato
% Cv = cell-array: Cv{k} è la matrice di osservazione completa (in ogni
% istante se un pacchetto non arriva la matrice relativa è posta a zero)
% Qv = matrice varianza dell'errore di modello
% Rv = cell-array: Rv{k} è la matrice diagonale a blocchi data dalle matrici
% varianza d'errore dei sensori (sono nulli gli elementi che non giungono
% alla bs)
% mu0v = media condizione iniziale x(0)
```

```
% P0v = varianza condizione iniziale x(0)
% yv = matrice di uscite complessive (ogni riga i è una traiettoria completa
% dell'osservazione effettuata dal sensore i; sono nulli gli elementi che non
% giungono alla bs)
function [xagg, Ppred] = Kalmantv_pl_Nsens(Av,Cv,Qv,Rv,mu0v,P0v,yv)
Ppred = cell(1,length(yv));
Ppred{1} = P0v;
Kv = cell(1,length(yv));
Kv{1} = Ppred{1}*Cv{1}'*pinv(Cv{1}*Ppred{1}*Cv{1}'+Rv{1});
xagg = zeros(length(mu0v), length(yv));
%inizializzazione dell'algoritmo
xagg(:,1) = mu0v + Kv{1}*(yv(:,1)-Cv{1}*mu0v);
for k = 1:size(yv,2)-1


    Ppred{k+1} = Av*Ppred{k}*Av' + Qv - Av*Kv{k}*Cv{k}*Ppred{k}*Av';
    Kv{k+1} = Ppred{k+1}*Cv{k+1}'*pinv(Cv{k+1}*Ppred{k+1}*Cv{k+1}'+Rv{k+1});
    xagg(:,k+1) = Av*xagg(:,k) + Kv{k+1}*(yv(:,k+1)-Cv{k+1}*Av*xagg(:,k));

end
```

Bibliografia

[1] L.Schenato, "Optimal sensor fusion for distributed sensors subject to random delay and packet loss", *IEEE Conference on Decision and Control* (CDC 07), New Orleans, USA, December 2007.

[2] A. A. Abidi, G. J. Pottie, and W. J. Kaiser, "Power-conscious design of wireless circuits and systems", *Proceedings of the IEEE*, vol. 88, pp. 1528-1545, October 2000.

[3] J. D. Wolfe, J. L. Speyer, "A low-power filtering scheme for distributed sensor networks", *Proceedings of the 42nd IEEE*, Conference on Decision and Control, Maui, Hawaii USA, December 2003.

[4] G. Picci, "Filtraggio statistico (Wiener, Levinson, Kalman) e applicazioni", Edizioni Libreria Progetto Padova, 2007.

[5] A. Agnoli, A. Chiuso, P. D'Errico, A. Pegoraro, L. Schenato, "Sensor fusion and estimation strategies for data traffic reduction in rooted wireless sensor networks".

[6] E. Song, Y. Zhu, J. Zhou, Z. You, "Optimal Kalman filtering fusion with cross-correlated sensor noises", *Automatica*, 43(2007), pp. 1450-1456.

[7] L. Schenato, "Optimal estimation in networked control systems subject to random delay and packet drop", to appear in *IEEE Transactions on Automatic Control.*

[8] A. S. Willsky, M. G. Bello, D. A. Castanon, B. C. Levy, G. C. Verghese, "Combining and updating of local estimates and regional maps along sets of one-dimensional tracks", *IEEE Transactions on Automatic Control*, vol. AC-27, NO. 4, August 1982.

[9] L. Schenato, Dispense di Progettazione di Sistemi di Controllo, 2007;

[10] P. Baldi, "Calcolo delle probabilità e statistica", McGraw Hill;

[11] M. Huang, S. Dey, "Stability of Kalman filtering with Markovian packet losses", *Automatica*, Volume 43, Issue 4, April 2007.

[12] O.L.V Costa, M.D. Fragoso and R.P Marques, “Discrete-Time Markov Jump Linear Systems “.

[13] S. Craig Smith, "Estimation with lossy measurements: jump estimators for jump systems".