POLITECNICO DI MILANO

FACOLTA' DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Dipartimento di Elettronica, Informazione e Bioingegneria

Corso di Laurea Magistrale in Ingegneria dell'Automazione

ANALISI DINAMICA DI DEVICES DI RETE: MODELLISTICA E PREDIZIONE

Relatore: Prof. SERGIO BITTANTI

Correlatori: Prof. Paolo Giuseppe Emilio Bolzern

Dott. Simone Formentin

Ing. Petrika Lluka

Tesi di Laurea di:

Paolo GIAMBI

Enea ZAPPELLA

Matr. 770191

Matr. 783230

Anno Accademico 2012-2013

Paolo: a Mauro, Chiara, Margi e Gugu
che mi hanno sempre dato la forza di non mollare mai.

Enea: alla mia famiglia

Indice

1

Introduzione

Nel corso degli anni Internet è diventato parte integrante della società moderna.

Viene usato con fini differenti, quali transazioni finanziarie, accesso alla conoscenza, scientifica e non, ma anche per coltivare relazioni e partecipare a comunità che difficilmente potrebbero esistere senza di esso [1].

Quest'organo vitale della società comprende un largo numero di reti di telecomunicazione, amministrate da diverse authorities e radicate in tutto il globo.

Data la centralità di questo strumento, si può capire come le problematiche ad esso collegate siano di fondamentale importanza e diversificate tra di loro, a causa del largo utilizzo che si fa della rete.

Per questi motivi è importante rendere tale strumento il più performante possibile: aldilà dell'infrastruttura fisica, link e devices che vanno a formare la rete globale, questi tesi va ad approfondire le attuali conoscenze nell'ambito dell'ottimizzazione delle prestazioni di una rete.

Per poter sfruttare al meglio le potenzialità della rete, è necessario disporre di un modello, il più possibile accurato, delle dinamiche delle risorse a disposizione. In particolare più importanti a questo scopo sono tre:

- L'analisi della relazione in un router, tra il traffico in entrata e la relativa allocazione di risorse.
- La predizione dell'andamento futuro del traffico, a partire dai valori assunti nel passato.
- La predizione dell'utilizzo delle risorse di un router, a partire dalla conoscenza dell'evoluzione futura del traffico in ingresso.

I dati utilizzati all'interno di questa trattazione provengono da una rete informatica di medie dimensioni e sono stati forniti da British Telecom, società con cui si è collaborato per lo sviluppo di questa tesi.

Questi temi sono già stati parzialmente affrontati in passato, come mostrato nello specifico nella prossima sezione.

Stato dell'arte

Nell'ambito del design e del mantenimento di reti di telecomunicazione esistono differenti problematiche, legate al loro utilizzo ottimale, ottenibile mediante un monitoraggio più o meno frequente e ad un aggiornamento periodico di hardware e software della rete, che nel corso degli anni sono state affrontate con diversi approcci. Chi cerca di dare una soluzione a queste problematiche ha come obiettivo la pianificazione degli interventi di manutenzione in una rete, attività meglio conosciuta come *capacity planning*.

L'obiettivo del cosiddetto "capacity planning" è quello di prevedere quali saranno gli andamenti futuri del traffico e delle risorse a disposizione dei device (router e switch [2] [3]), sulla base dei dati di traffico a disposizione, in modo da poter intervenire efficacemente in tutti e soli i punti della rete che necessitano potenziamenti. Solitamente, nella pratica corrente non si effettua questa analisi, e la politica di manutenzione di una rete informatica prevede un potenziamento periodico (per esempio annuale) delle risorse, senza uno studio rigoroso di quali siano le zone più a rischio all'interno della rete e gli interventi appropriati. In tal modo si ha il "vantaggio" di evitare di affrontare in via preliminare lo studio del problema, cosa che invece succede all'interno di un capacity plan, ma non si ha garanzia che gli investimenti fatti siano efficaci: potrebbero infatti venir potenziate aree che non necessitano di un aumento di risorse, oppure le risorse di un'area maggiormente sollecitata della rete potrebbero non essere incrementate a sufficienza. L'obiettivo che il capacity planning si pone, previo studio delle caratteristiche della rete, è quello di massimizzare i potenziamenti delle risorse di una rete andando ad intervenire nelle sole aree considerate a rischio, mantenendo un budget limitato.

Una suddivisione di massima in tematiche può essere la seguente:

- La modellizzazione del traffico in ingresso ad un router con scopo predittivo;
- L'analisi dell'occupazione delle risorse della rete in base alla quantità di traffico presente,

La prima tematica consiste nella modellizzazione del traffico e si pone come obiettivo finale la predizione del suo andamento futuro in condizioni nominali, cioè in assenza di eventi rari e imprevedibili. A seconda dell'orizzonte predittivo considerato, si può distinguere in predizione a breve termine (short-range) e predizione a lungo termine (long-range). In letteratura è possibile individuare numerosi modelli riguardanti la predizione a breve termine, mentre sono minori gli studi per lo sviluppo di modelli per la predizione di tipo a lungo termine.

La seconda tematica sull'occupazione delle risorse risulta essere meno trattata, nonostante la sua notevole importanza: peraltro, nel progetto di una rete è cruciale essere in grado di ricostruire e predire l'utilizzo delle risorse, in modo di poter intervenire tempestivamente in caso di congestioni, e pianificare con criterio gli interventi di potenziamento delle parti della rete che più ne necessitano.

Oltre alle tematiche presentate, esiste una terza problematica che ha come obiettivo la ricostruzione dei flussi OD (origine-destinazione) e dei traffici in una rete. Dal momento che non è conveniente, per ragioni economiche, monitorare ogni singolo link e che la rielaborazione di tutti i dati per determinare i flussi OD richiede un elevato utilizzo di risorse, si preferisce monitorare solamente alcuni link tra tutti quelli considerati e ricostruire a partire da quest'ultimi i restanti

traffici non osservati. Tale problematica risulta ben affrontata in letteratura e non sarà trattata nel corso di questa tesi.

La nostra trattazione si concentra maggiormente sullo sviluppo di un modello in grado di ricavare l'utilizzo delle risorse allocate (nello specifico la CPU) all'interno di un router a partire dal traffico in entrata dalla rete e sullo sviluppo di modelli validi per la predizione short-range e long-range.

In questo capitolo si presenta come prima cosa uno studio delle soluzioni sviluppate fino ad oggi per le varie problematiche considerate, per poi procedere, nei capitoli successivi, a mostrare i risultati sviluppati nell'ambito del lavoro di tesi.

1.1 **Predizione di traffico**

1.1.1 Short-Range Prediction

Ipotesi fondamentale alla base delle trattazioni su questo argomento è che l'andamento del traffico ad un dato istante sia dipendente dal valore che il traffico ha assunto negli istanti precedenti; in altre parole il traffico può essere rappresentato da un modello dinamico con equazione:

$$x(t) = \gamma_1 x(t-1) + \gamma_2 x(t-2) + \cdots + \gamma_p x(t-p) + e(t) + \theta_1 e(t-1) + \cdots + \theta_q e(t-q)$$

In cui $x(t)$ rappresenta il traffico misurato su un link, il cui valore dipende dai p valori assunti nei passi precedenti dalla variabile x e dai q valori assunti dal rumore $e(t) \sim WN(0, \sigma^2)$. I coefficienti γ_i e θ_i sono i pesi che definiscono la relazione tra i valori di $x(t)$ ed $e(t)$ a diversi istanti temporali.

Modello ARIMA

Un primo modello che potrebbe essere considerato è il modello ARMA (AutoRegressive Moving Average*)* [4], tuttavia per poter essere adoperato esso presuppone che la serie temporale da modellizzare sia stazionaria, mentre tipicamente il traffico, presentando trend e stagionalità, non lo è.

Si ovvia quindi a questo problema utilizzando un modello ARIMA (AutoRegressive Integrated Moving Average*),* rappresentato dall'equazione:

$$\Phi(z^{-1})\nabla^d X(t) = \Theta(z^{-1})e(t)$$

in cui:

- $\nabla X(t)$ rappresenta la differenziazione della serie temporale considerata, cioè:

$$\nabla X(t) = X(t) - X(t-1) = (1 - z^{-1})X(t)$$

- l'esponente d rappresenta il numero di differenziazioni effettuate sulla serie.

- i polinomi Φ e Θ corrispondono ai polinomi ϕ e θ del modello ARMA[4].

Il vantaggio di un modello ARIMA rispetto ad un ARMA sta nel poter ricondursi a dei segnali stazionari a partire da serie temporali che non lo sono: se ad esempio consideriamo un segnale avente un trend lineare, differenziando il segnale una volta (ponendo quindi $d = 1$) l'apporto del trend al segnale ricavato sarà pari ad una costante, rimuovendo quindi la causa della non stazionarietà.

Per determinare quindi il valore di d occorre differenziare ripetutamente la serie temporale, fino ad ottenere un segnale stazionario. In letteratura non sono state riscontrate trattazioni rigorose sul metodo con cui si ricava il valore di tale parametro, ma solo un'indicazione sul fatto che, generalmente, è sufficiente differenziare una o due volte per ottenere un segnale stazionario [6].

Identificato il valore del parametro d si può determinare l'ordine della parti AR e MA del modello e successivamente i valori dei parametri Φ_i e Θ_i attraverso le tecniche di Box-Jenkins, a conferma del fatto che una volta differenziato il segnale si possa intendere il modello ARIMA come un modello ARMA.

Modello ARIMA-GARCH

Il secondo modello di cui si è riscontrato un uso in letteratura per la predizione di tipo short-range è l'ARIMA-GARCH: si tratta di una combinazione tra un modello lineare (ARIMA) e una varianza condizionale (tempo variante) che viene assunta essere di tipo GARCH (Generalized Autoregressive conditional Heteroscedasticity). Nei modelli ARIMA appena descritti la varianza viene sempre considerata costante nonostante il traffico presenti un andamento fortemente bursty. All'interno di [7] viene introdotto il modello GARCH per modellizzare segnali di traffico cercando di descrivere la componente bursty dei dati con una varianza dinamica nel tempo.

Tale modello può essere espresso dalle formule di seguito riportate. Per quanto riguarda la parte ARIMA del modello:

$$\phi(z^{-1})(1 - z^{-1})^d X(t) = \theta(z^{-1})e(t)$$

In cui $X(t)$ rappresenta la serie temporale da modellizzare ed $e(t) \sim WN(0, \sigma(t)^2)$. La varianza condizionale è rappresentata da $\sigma(t)^2$ che per definizione vale:

$$Var\big(X(t)\big|X(t-1)\big) = \mathbb{E}[Z(t)^2|Z(t-1)] = \sigma(t)^2$$

Mentre i polinomi ϕ e θ corrispondono ai polinomi Φ e Θ utilizzati nel modello ARIMA precedentemente illustrato. La caratteristica più importante del modello è quella di basarsi su una varianza condizionale che indica una dipendenza esplicita dalle osservazioni passate. Nei modelli GARCH la distribuzione condizionale di $e(t)$ si ottiene introducendo una parametrizzazione che descriva la dipendenza della varianza condizionale dal rumore bianco $Z(t)$.

$$\sigma(t)^2 = \alpha_0 + \sum_{i=1}^{r} \gamma_i \sigma(t-i)^2 + \sum_{j=1}^{m} \alpha_j e(t-j)^2, \qquad \alpha_0 > 0, \alpha_j \geq 0, \gamma_i \geq 0$$

con il vincolo:

$$\sum_{i=1}^{r} \gamma_i + \sum_{j=1}^{m} \alpha_j < 1$$

Come descritto in [7].

Perciò il modello da identificare sarà un ARIMA(p, d, q)/GARCH(r, m), che descritto in modo compatto risulta essere:

$$\begin{cases} \nabla^d X(t) = \sum_{i=1}^{p} \phi_i \nabla^d X(t-i) + e(t) + \sum_{j=1}^{q} \theta_j e(t-j) \\ \sigma(t)^2 = \alpha_0 + \sum_{i=1}^{r} \gamma_i \sigma(t-i)^2 + \sum_{j=1}^{m} \alpha_j e(t-j)^2 \end{cases} \qquad e(t) \sim WN(0, \sigma(t)^2)$$

Per la stima dei parametri della parte GARCH, si suppone inizialmente che i parametri *p* e *q* della parte ARIMA siano stati impostati a zero, in modo che il modello regredisca alla sola parte GARCH, per poter stimare i parametri *r* e *m* ed i relativi coefficientiγ_i e α_j. Come descritto in [8] e [9], per esperienze di ricerca, impostare *r* e *m* ad un valore pari ad uno porta a descrivere la varianza con performance generalmente soddisfacenti. Determinata la parte GARCH del modello, un possibile metodo empirico per stimare gli ordini delle parti MA e AR della parte ARIMA del modello consiste nell'osservare le funzioni di autocorrelazione e autocorrelazione parziale, tenendo però conto del fatto che, come descritto in [7], tali funzioni sono ricavate a partire dal set di dati considerato e, perciò, si deve disporre di un set di dati adeguato (in termini di numero di samples e assenza di eventuali outliers).

1.1.2 Long-Range Prediction

Dal momento che l'obiettivo del capacity planning è l'ottimizzazione delle performance della rete presa in esame, non è possibile limitarsi unicamente a considerare un orizzonte temporale breve, ma si deve considerare come l'andamento del traffico (e quindi utilizzo di risorse) evolverà nel lungo periodo. Ciò è necessario per poter capire quali parti di una rete richiederanno un potenziamento e quali invece non ne avranno bisogno.

In questo caso, un modello ARIMA non è in grado di dare risultati soddisfacenti per la predizione long-range; infatti, visto che tipicamente i traffici presenti in una rete presentano una stagionalità settimanale e/o giornaliera, il modello ARIMA non è in grado di differenziare e sfruttare tali informazioni.

Per questo motivo un primo tipo di modello che si può riscontrate in letteratura è un evoluzione del modello ARMA, definito come SARIMA [10] [11] (Seasonal AutoRegressive Integrated Moving Average*)*. L'equazione che descrive il modello è:

$$\Phi(z^{-1})\varphi(z^{-1})\nabla^d X(t) = \Theta(z^{-1})\vartheta(z^{-1})e(t)$$

in cui:

- il parametro d e i vettori Φ e Θ sono i corrispondenti dell'ARIMA precedentemente descritto.
- il parametro D e i vettori φ e ϑ servono a modellizzare la componente stagionale del segnale.

Per una trattazione di come identificare i parametri si rimanda agli articoli citati nella bibliografia [10] [11], sottolineando però come non siano stati individuati trattazioni rigorose sulle modalità usate per tale identificazione.

I modelli SARIMA presentano una predizione in cui, a differenza del modello ARIMA, non ci si assesta su un valore costante dopo un iniziale transitorio, ma si prevede un andamento stagionale simile a quello avuto nei dati usati nell'identificazione del modello. Ciò porterà ad avere buone previsioni nel momento in cui la componente stagionale risulterà dominante.

Un secondo approccio per la predizione long-range del traffico, alternativo all'uso di un modello SARIMA è affrontato in [12]: la componente fondamentale del segnale viene identificata con il trend, in quanto si ritiene che, per poter ottenere una predizione a lungo termine sui livelli futuri di traffico, essa sia la componente principale, che permette di capire se l'utilizzo delle risorse (capacità dei link in questo caso) stia crescendo e, se necessario, effettuare una riorganizzazione preventiva della rete per evitare la saturazione dei link. Non si è quindi interessati a modellare le dinamiche giornaliere del traffico, ma solamente il suo andamento generale su un orizzonte temporale molto lungo, in modo da capire se la quantità di traffico stia aumentando e, in caso affermativo, con che velocità ciò stia accadendo.

Per ottenere tale predizione in [12] si presenta un modello che fa uso di una previa scomposizione dei dati (sottocampionati con periodo pari 90 minuti) attraverso il metodo wavelet MRA [13] [14] e suddivide i dati di traffico in una componente di trend ed deviazione dal trend stesso.

A partire dai dati considerati si ottengono delle serie temporali sempre più filtrate del segnale di traffico, da quella con maggior dettaglio a quella più filtrata, quest'ultima viene considerata la componente di trend, mentre per ottenere la componente di deviazione si utilizza la serie più filtrata tra quelle che ancora comprendono le oscillazioni dovute all'andamento stagionale.

Entrambe le componenti, prese separatamente, rappresentano una serie temporale che viene identificata attraverso modelli ARIMA. Con la predizione di ognuna delle due componenti è possibile ricomporre un segnale che predica l'andamento futuro del traffico medio sull'insieme di link tra due nodi.

I dati di traffico $x(t)$ vengono quindi così scomposti:

$$x(t) = c_p(t) + \sum_{j=1}^{p} d_j(t)$$

in cui:

- $c_0(t) = x(t)$
- $c_j(t) = \sum_{l=-\infty}^{\infty} h(l)\, c_{j-1}(t + 2^{j-1}l)$
- $d_j(t) = c_{j-1}(t) - c_j(t)$

dove $c_j(t)$ è l'approssimazione di $x(t)$ per il *j*-esimo filtraggio, $c_p(t)$ viene chiamato residuo, rappresentando il termine $c_j(t)$maggiormente filtrato, e viene sommato ai termini $d_j(t)$, detti dettagli, per ricomporre il segnale; infine $h(l)$ è un coefficiente di smoothing che mette in relazione i vari $c_j(t)$ tra di loro. In questo modo il segnale viene visto come una somma di contributi e, determinando il numero totale di dettagli, è possibile scegliere quanto rendere smoothed (o al contrario fedele ai dati di partenza) il segnale $x(t)$.

In particolare, in [12] l'ultima scala temporale scelta è quella per *p* pari a 6 poiché in tal modo il residuo $c_6(t)$ risulta essere sufficientemente filtrato da poter essere considerato un trend a lungo termine e rappresentare dal 95% al 97% dell'energia dei segnali su tutti i link (in cui l'energia viene calcolata come $E = \sum_{t=1}^{N} {c_6}^2(t)$). Tale scelta risulta inoltre comoda in quanto in questo modo la serie temporale del trend non risente delle fluttuazioni a 12 e 24 ore poiché avendo un dato ogni 96 ore ($2^6 \times 1.5 = 96$) rappresenta circa l'andamento settimanale (1 settimana = 168 ore). La quasi totalità della restante energia del segnale giace nel dettaglio alla terza scala temporale $d_3(t)$ ovvero quella che considera osservazioni con una passo di 12 ore.

Riassumendo viene creato un modello che descrive i dati composto solo dal trend $c_6(t)$ e dal dettaglio $d_3(t)$:

$$x'(t) = c_6(t) + m d_3(t)$$

in cui il coefficiente m che moltiplica il dettaglio viene identificato attraverso un uso dei minimi quadrati usando il segnale originale.

Infine all'interno di [12] viene fatta l'assunzione arbitraria che il comportamento giornaliero non abbia importanza in un'ottica di capacity planning, per cui i due termini $c_6(t)$ e $d_3(t)$ vengono sostituiti dai termini $l(j)$ e $dt_3(j)$, ottenuti dai precedenti mediando i dati di una settimana per ottenere delle serie temporali con un unico valore valido per tutta la settimana; quindi il modello diventa:

$$\hat{x}(t) = l(j) + m dt_3(j)$$

in cui $l(j)$ è considerato il trend di lungo termine mentre $dt_3(j)$ rappresenta la deviazione standard dal valore di trend. In figura 1.1 (come riportato in [12]) si confrontano i dati (ricavati come descritto sopra) ed il modello con valori per ogni settimana, comparando il solo trend $l(j)$ e il trend più e meno la deviazione standard.

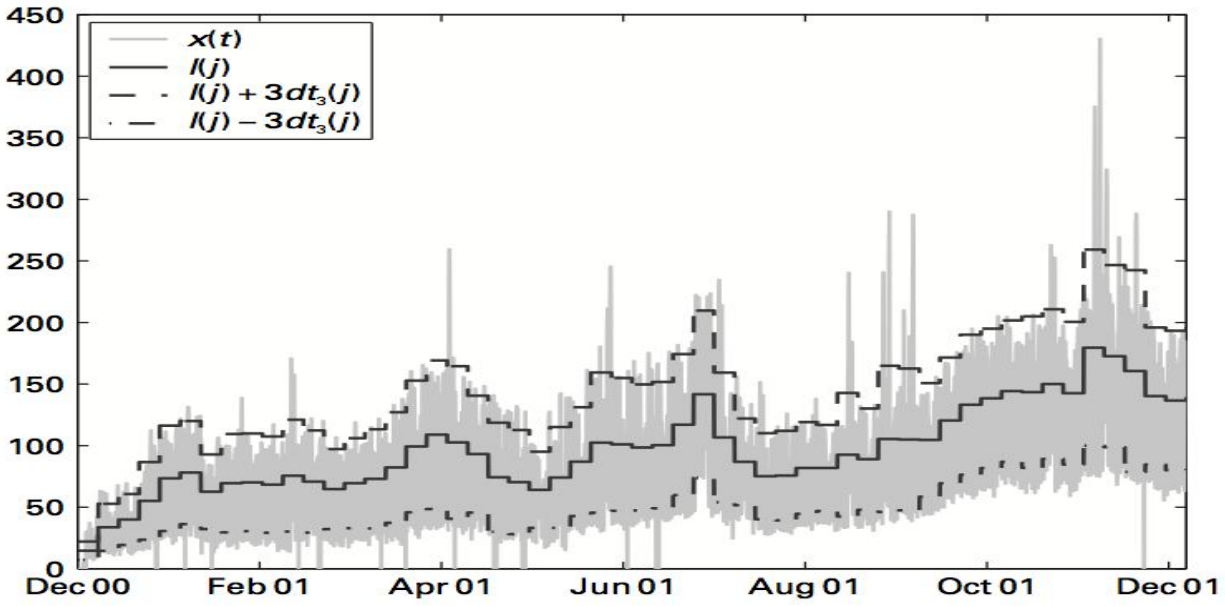


Figura 1.1: confronto tra gli andamenti del set di dati e del trend $l(j)$ (linea continua) a cui viene sommata e sottratta la deviazione standard

Individuate le 2 grandezze di interesse si procede poi in un secondo momento ad identificare per ognuna di esse un modello ARIMA secondo i metodi descritti a inizio capitolo, per poi ricavarne i predittori. In figura 1.2 (come riportato da [12]) è presente il risultato di tale operazione:

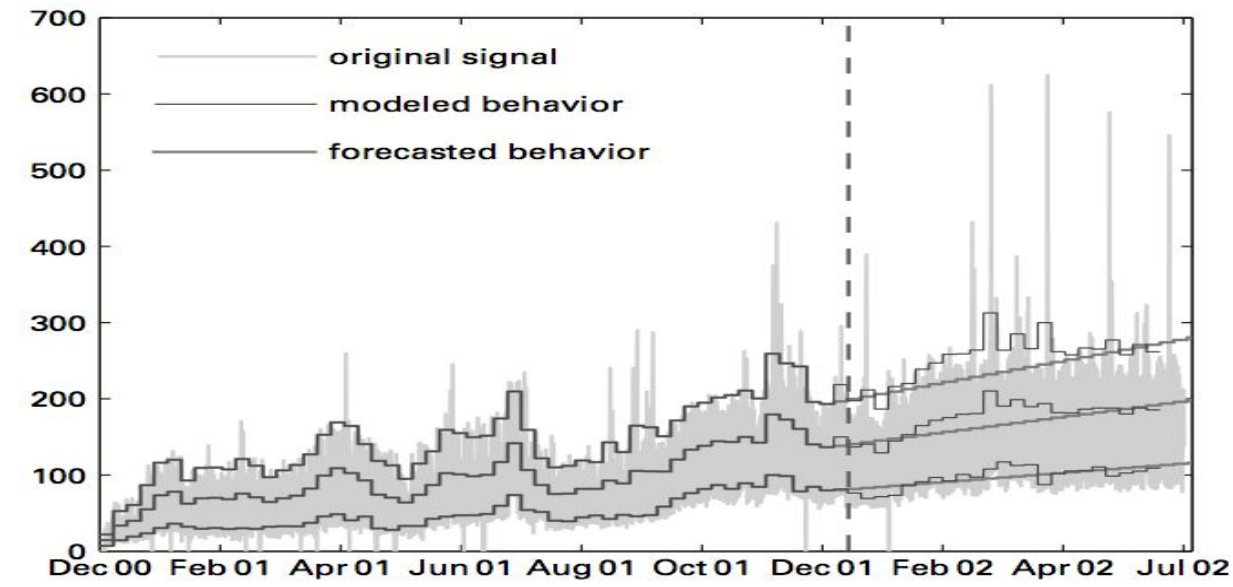


Figura 1.2: confronto tra la previsione del modello e l'andamento del set di dati

Concludendo, la metodologia presentata in [12] è, per quanto teoricamente non immediata, di implementazione relativamente semplice e può essere completamente automatizzata, contrariamente al modello SARIMA, di cui non sono stati riscontrati in letteratura metodi per l'identificazione di tutti i parametri del modello.

Va però sottolineato che il risultato di un metodo come quello appena descritto dà dei risultati soddisfacenti se l'obiettivo è prevedere l'andamento che seguirà generalmente il traffico nelle settimane future a quella da cui si comincia a fare predizione, ma non è soddisfacente se l'obiettivo è fare predizione long-range nell'ordine di tre-quattro settimane, in cui non si è interessati solo al trend che seguirà il traffico, ma anche alla stagionalità del segnale.

1.2 **Utilizzo delle risorse all'interno di un router**

La seconda problematica presente in quest'ambito consiste nella determinazione di un modello che sia in grado di descrivere l'occupazione delle risorse dei device presenti all'interno di una rete quali router e switch. Ciò interessa in particolar modo chi va a fare capacity planning: per capire quali sono i punti della rete che necessitano effettivamente di un potenziamento, è necessario avere a disposizione degli strumenti in grado di caratterizzare in modo soddisfacente l'allocazione di risorse in risposta alla mole di traffico della rete.

Considerando ad esempio switch e router, è noto come tali device non abbiamo performance indipendenti dalla quantità di risorse allocate, ma al contrario che le loro prestazioni siano fortemente dipendenti da quale percentuale di risorse sia utilizzata e quale risulti disponibile.

In particolare quando si parla di risorse di un device ci si riferisce a memoria e CPU. Mentre la memoria viene utilizzata per memorizzare le tabelle di routing, le VLAN (*Virtual Local Area Network*) e altre informazioni utili al router e la quantità totale che ne viene richiesta risulta essere prevedibile (almeno qualitativamente), l'utilizzo di CPU varia invece in base al numero di pacchetti da rielaborare, ed è quindi fortemente dipendente dal traffico. Tra le due grandezze, quella più critica è quindi sicuramente la seconda: non è infatti pensabile progettare una rete i cui i device risultino avere percentuali di utilizzo di CPU costantemente ad alti livelli. Quando l'allocazione di CPU oltrepassa un valore soglia, le attività del device (processing dei dati in ingresso, conseguente immissione dei dati processati sulla rete, riaggiornamento periodico delle tabelle di routing) rallentano, portando ad una diminuzione delle prestazioni; ciò non fa altro che far salire la mole di dati da processare, in quanto il numero di Megabyte di traffico in ingresso al device è indipendente dal device stesso, generando quindi un ulteriore aumento del carico di lavoro a cui il device risulta sottoposto. A seguito di questo aumento, il router (o lo switch) rallenta ulteriormente la propria attività, fino a quando si arriva ad un utilizzo delle risorse talmente alto per cui il device cessa di rispondere agli stimoli provenienti dalla rete e di fatto risulta essere a tutti gli effetti "down". Ciò porta quindi ad avere una zona della rete congestionata, generando dei sovraccarichi che il capacity plan si pone di evitare.

Nel caso di uno switch, la soglia di CPU da non superare è pari al 60-70%, mentre per i router risulta essere più bassa, pari a circa il 45-50%. Si capisce quindi come i router siano l'anello più a rischio della catena, e che ci sia quindi bisogno di poter avere a portata di mano degli strumenti in grado di caratterizzare l'occupazione delle risorse in funzione del traffico presente nella rete considerata.

In letteratura non sono però stati trovati degli studi che si pongano come obiettivo la creazione di tali modelli, lasciando quindi irrisolto il problema. Il primo obiettivo di questa tesi sarà quindi la ricerca di un modello, possibilmente semplice, in grado di poter essere usato per operare una conversione traffico-utilizzo di CPU, in modo di poter garantire, posta nota la conoscenza del traffico in ingresso ad un router, il conseguente valore percentuale di allocazione di CPU.

1.3 **Kriging: ricostruzione di flussi e traffici della rete**

Nell'ambito del monitoring e dell'ottimizzazione delle prestazioni delle reti sono inoltre presenti ulteriori problematiche; una di queste è la ricostruzione di flussi e traffici non osservati.

Allo scopo di poter fare capacity planning è sicuramente utile essere a conoscenza di quali siano i link della rete più usati per il trasferimento dei dati, dal momento che tale conoscenza permette di potenziare la capacità dei link maggiormente stressati, il potere computazionale dei nodi, oppure fare manutenzione identificando da subito le cause di un possibile malfunzionamento.

Nonostante la quantità dei Mbps trasferiti dalle varie interfacce dei dispositivi attraverso i link della rete possa essere misurata, ciò richiede una quantità di tempo non indifferente: si può ovviare a questo problema a patto che solo i link principali della rete vengano monitorati diminuendo il tempo di attesa per i dati. Gli obiettivi del Kriging sono due: ricostruire l'andamento del traffico (in Mbps) sui link in cui non viene misurato e ricostruire l'andamento dei flussi OD (definiti di seguito).

Per poter ricostruire i dati sui link non osservati è necessario non solo conoscere quanto traffico attraversa i link osservati, ma anche avere una conoscenza di dove questi dati verranno smistati dai vari nodi; deve quindi essere noto quali nodi stanno comunicando tra loro, formando parte del traffico che attraversa i link che li collegano. A questo scopo, oltre alla grandezza traffico, s'introduce una nuova grandezza: il flusso OD (Origin-Destination).

Il traffico, misurato in Mbps, è definito come la quantità Megabyte per secondo che attraversano il link ed è misurato ai buffer d'uscita dei nodi della rete.

Il flusso, anch'esso Mbps, è definito come traffico scambiato tra due nodi, dove il flusso da A a B è diverso dal flusso da B ad A. Ogni pacchetto dati passante all'interno della rete contiene l'informazione di quali sono gli IP che si stanno scambiando i dati contenuti nel pacchetto; quindi a partire da ciò è possibile ricostruire i flussi OD all'interno della rete. Quest'operazione tuttavia è più complessa del semplice ottenimento della misura di traffico, per cui richiede più tempo per essere calcolata.

In [12], [15] e [16] il problema del Kriging viene affrontato utilizzando la definizione di *Routing Equation,* che mette in relazione tutti i flussi e i traffici della rete:

$$y(t) = Ax(t)$$

Dove $y(t)$ è il vettore dei traffici, $x(t)$ è il vettore dei flussi e la matrice A viene chiamata Routing Matrix; i suoi elementi possono assumere valori pari a 0 o 1, poiché il singolo elemento a_{lj} indica l'appartenenza o meno di un flusso x_j ad un certo traffico y_l.

In [15] e [16] ci si dedica alla ricostruzione dei traffici non osservati a partire dai dati di traffico misurati, utilizzando inizialmente per la taratura le informazioni dei flussi; all'interno di [12] ci si concentra invece sul ricostruire i flussi OD a partire dai traffici a disposizione, usando anche in questo caso i dati di flusso per la taratura iniziale.

In letteratura tale problema è affrontato approfonditamente con risultati positivi che forniscono una soluzione soddisfacente al problema. Per questo motivo, all'interno di questa tesi non è di interesse approfondire quest'ambito.

CONCLUSIONI

Dallo studio dell'arte presentato si può concludere come il problema della predizione dell'andamento futuro del traffico sia stato affrontato approfonditamente con differenti approcci, facendo uso di diversi modelli per tentare di descriverne le dinamiche.

In particolare si possono trarre le seguenti conclusioni:

- In letteratura non vengono sviluppati lavori che utilizzano il modello ARMA, dal momento che esso non risulta adatto per un segnale non stazionario. In realtà quando è possibile scomporre il segnale in trend, stagionalità e parte stazionaria, si ottengono 3 contributi, di cui i primi due risultano deterministici. Il modello ARMA può essere perciò applicato se si considera avente in ingresso non il segnale complessivo, ma solo la sua componente stocastica.
- Il modello ARIMA risulta invece adatto a descrivere il segnale senza bisogno di pre-processing. Tuttavia non sono presenti, in letteratura, modalità rigorose con cui determinare il numero di differenziazioni necessarie (espresso dal parametro d) per ottenere un segnale stazionario. Di fatto la scelta di tale parametro risulta totalmente dipendente dai dati che si considerano, non essendo perciò determinabile a priori.
- L'utilizzo del metodo Wavelet MRA consente di avere predizioni di traffico per un orizzonte predittivo esteso, ma a patto di poter prevedere solamente l'andamento del trend che forma il segnale, non riuscendo a fornire indicazioni sulla stagionalità del traffico.
- Il modello SARIMA consente invece la descrizione del traffico in un ambito di predizione long-range, ma non sono stati riscontrate in letteratura trattazioni su come poter identificare in modo rigoroso i parametri aggiuntivi presenti nel modello rispetto ad un modello ARIMA.

Per quanto riguarda il problema di come gli utilizzi delle risorse di un device possano essere collegati alla quantità di traffico in esso entrante, tale campo risulta tutt'ora inesplorato, non essendo stati riscontrati in letteratura studi su di esso. Non si è a conoscenza quindi di trattazioni che permettano di risalire, a partire dalle conoscenze del traffico presenti in una rete, alla misura di quanto le risorse fisiche di un device (CPU, memoria) siano adoperate, non permettendo quindi di poter capire, in un'ottica di capacity planning, quali siano i punti maggiormente stressati all'interno di una rete.

I contributi innovativi presentati all'interno di questa tesi riguardano l'analisi del rapporto esistente tra il traffico entrante in un router e il relativo utilizzo di CPU e la predizione di quest'ultimo, attraverso l'impiego del metodo di Holt-Winters, a partire dall'andamento del traffico.

All'interno del capitolo 2 ci si concentra sul secondo problema elencato, andando ad individuare un modello che possa descrivere in maniera adeguata il legame tra traffico e CPU allocata all'interno di un router. Dapprima si procede con l'analisi dei dati a disposizione, relativi ad un case study di una rete di medie dimensioni, ricavandone le diverse componente di trend, stagionalità e parte stocastica. In un secondo momento si ricavano due modelli basati su due diversi approcci: il primo stabilisce una relazione tra le singole componenti che vanno a formare traffico e CPU; il secondo descrive una relazione che considera i segnali complessivi. Ottenuti tali modelli per il

router preso in considerazione, se ne applicano i risultati ai restanti device, per poter verificare che essi siano adatti a tutti gli elementi che formano la rete e non solamente al router considerato.

Nel capitolo 3 ci si concentra sulla predizione di tipo short-range:

- Per prima cosa si utilizza un modello di tipo ARMA: la scelta di utilizzare tale modello è dovuta al fatto che, essendo tale tipologia descritta approfonditamente in letteratura, essa risulta largamente utilizzata in altri campi e quindi ben nota; inoltre può essere utilizzata come benchmark per i modelli successivamente sviluppati.
  Andando a riutilizzare il pre-processing fatto sui dati nel capitolo precedente, si identifica il modello considerato utilizzando la componente stocastica del segnale, la cui uscita viene poi sommata alle componenti deterministiche del traffico. Si passa quindi alla predizione del segnale considerando diversi orizzonti temporali per poter predire l'andamento nell'immediato futuro del traffico. Infine, utilizzando la relazione traffico-CPU sviluppata nel capitolo 2, si ottiene una predizione per l'utilizzo di CPU.
- Si considera poi un modello ARIMA: per prima cosa, studiando il set di dati a disposizione si determina il valore del parametro d non predeterminabile;in seguito dopo aver differenziato il segnale si procede ad identificare i restanti parametri, per poi ricavare i predittori di traffico e CPU.
- Da ultimo si considera il metodo di Holt-Winters [21] [22]: un modello di questo tipo permette di poter tener conto delle componenti di trend e stagionalità senza dover fare pre-processing dei dati; inoltre permette di tener traccia delle variazioni che tali contributi possono presentare in un certo lasso di tempo, riaggiornando automaticamente le due componenti. In letteratura tale modello risulta impiegato in ambito finanziario, per la previsione di titoli e simili, ma non se ne è riscontrato un utilizzo nel campo delle telecomunicazioni. Anche in questo caso si procede inizialmente con l'identificazione dei parametri del modello, per poi andare a ricavare i predittori per traffico e CPU.

Nel capitolo 4 si procede con la predizione di tipo long-range, con l'obiettivo di poter fornire una previsione dell'utilizzo di CPU per un lasso di tempo pari ad un mese, periodo sufficiente, in ottica di capacity planning, per poter andare ad intervenire su una rete nel caso se ne dimostrasse la necessità. Vengono sviluppati nuovamente i modello ARMA e di Holt-Winters seguendo gli stessi passi del capitolo precedente (identificazione del modello, predittore di traffico, passaggio alla predizione di CPU), mostrando inoltre come un modello ARIMA sia inadeguato per predire orizzonti di questa ampiezza.

Infine nel capitolo 5 si traggono le conclusioni di quanto si è sviluppato nei capitoli precedenti, riassumendo i risultati ottenuti e le innovazioni introdotte da questa tesi, per poi andare a mostrare i possibili sviluppi futuri.

2

Relazione statica lineare fra traffico in ingresso e CPU di un router

L'obiettivo di questo capitolo è la creazione di un modello che caratterizzi la relazione fra traffico in ingresso ad un router attraverso due o più interfacce e il corrispondente utilizzo di CPU, in condizioni di basso utilizzo di quest'ultima.

Si è scelto, inizialmente, di considerare una relazione statica e lineare che leghi le due grandezze nella forma: $y = Ku$, in cui si associa il traffico alla variabile d'ingresso u e l'utilizzo di CPU alla variabile d'uscita y. L'ipotesi che la relazione sia statica è dovuta al fatto che campionando con periodo pari ad un minuto si considera che tutte le dinamiche interne al router si esauriscano in un lasso di tempo inferiore a quello di campionamento.

Ulteriore ipotesi fondamentale per la validità del modello è che, nel caso di router a più interfacce, i contributi di traffico dati dalle singole interfacce possano essere sommati in un unico contributo, riconducendoci quindi ad un "router ideale" con un unico ingresso da cui proviene tutto il traffico.

Vengono sviluppati 2 diversi approcci al problema: in un primo caso si scompongono i segnali u e y; in un secondo caso si considerano i segnali complessivi, al fine di verificare se la divisione dei segnali nelle proprie componenti porti ad un vantaggio effettivo. Per poter avere un riferimento con cui confrontarsi, si considera infine un terzo caso, in cui si ricava unicamente la relazione tra componente stocastica di traffico e di CPU e, per ottenere il segnale totale di CPU, si sommano il trend e la stagionalità ricavati a partire dai dati di CPU in modo da poter avere un upper bound di riferimento.

In un primo momento si considera un router a 2 interfacce, in cui cioè tutto il traffico in entrata dal primo link viene ritrasmesso sul secondo (a meno di una minima parte che non viene ritrasmessa in quanto informazione scambiatasi periodicamente tra i router relativa allo stato della rete); generalizzando poi in un secondo momento il modello ai router dotati di più interfacce.

I dati sono relativi ad un case study per una rete di medie dimensioni. Futuri lavori potranno generalizzare i dati ottenuti in questo capitolo.

2.1 **I dati**

I dati a nostra disposizione provengono da una multinazionale la cui rete viene gestita da British Telecom e ne usa il backbone MPLS (*multiprotocol label switching*) come rete di trasporto. Per motivi di privacy tale multinazionale sarà indicata con il nome fittizio “Acme”, non potendone menzionare il nome. In Italia essa è presente in più regioni attraverso diverse filiali, a ciascuna delle quali è associata un router che funge da ingresso alla sottorete dedicata alla singola filiale; attraverso questo router transita quindi il traffico da e verso la sottorete.

Considerando tutti e soli i router che fungono da ingresso alle varie sottoreti, la configurazione della rete da essi formata è la seguente:

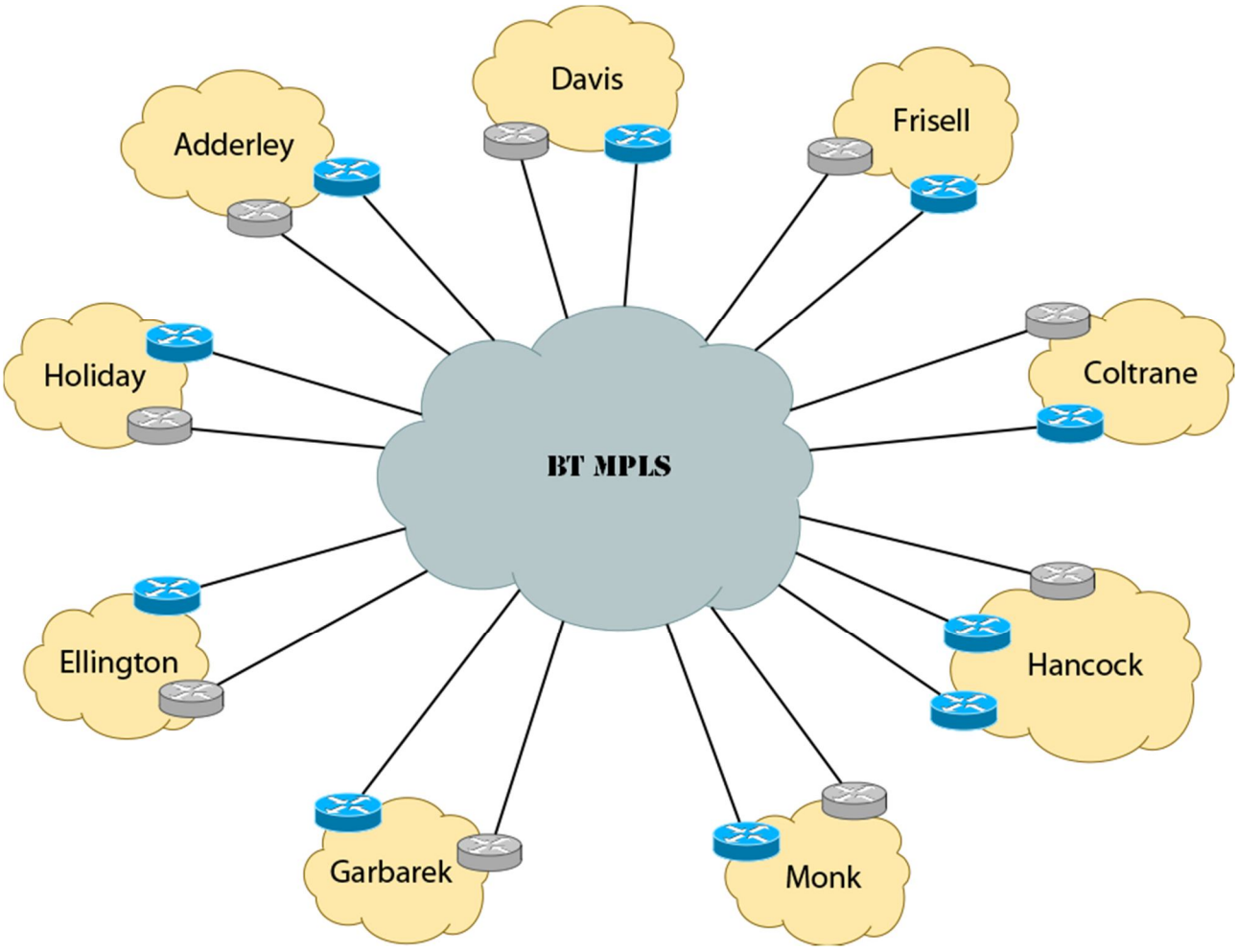


Figura 2.1: schema della rete ACME

In cui i router di colore grigio sono router in stand-by (e perciò non attraversati da traffico), mentre i router di colore blu sono i router attivi che verranno considerati nel corso di questa tesi.

2.1.1 Modalità di raccolta dati

Per il monitoraggio della rete e la raccolta dei dati si utilizza Netspyglass [17], tool utilizzato da British Telecom, proprietaria della rete stessa e responsabile del servizio di telecomunicazione offerto ad Acme.

Le principali funzionalità di Netspyglass sono:

-monitoraggio delle connettività

-monitoraggio del traffico di rete

-monitoraggio delle risorse (CPU, memoria, hard disk, banda)

-rilevamento di allarmi ed eventi e notifica tramite e-mail

-raccolta, analisi e storing di traffici SNMP e netflows

-capacity monitoring

Particolarmente importanti per lo sviluppo di questa tesi sono il secondo, il terzo e il quinto punto, in quanto permettono di avere a disposizione i dati con i quali si lavorerà.

In figura è mostrata l'architettura del sistema:

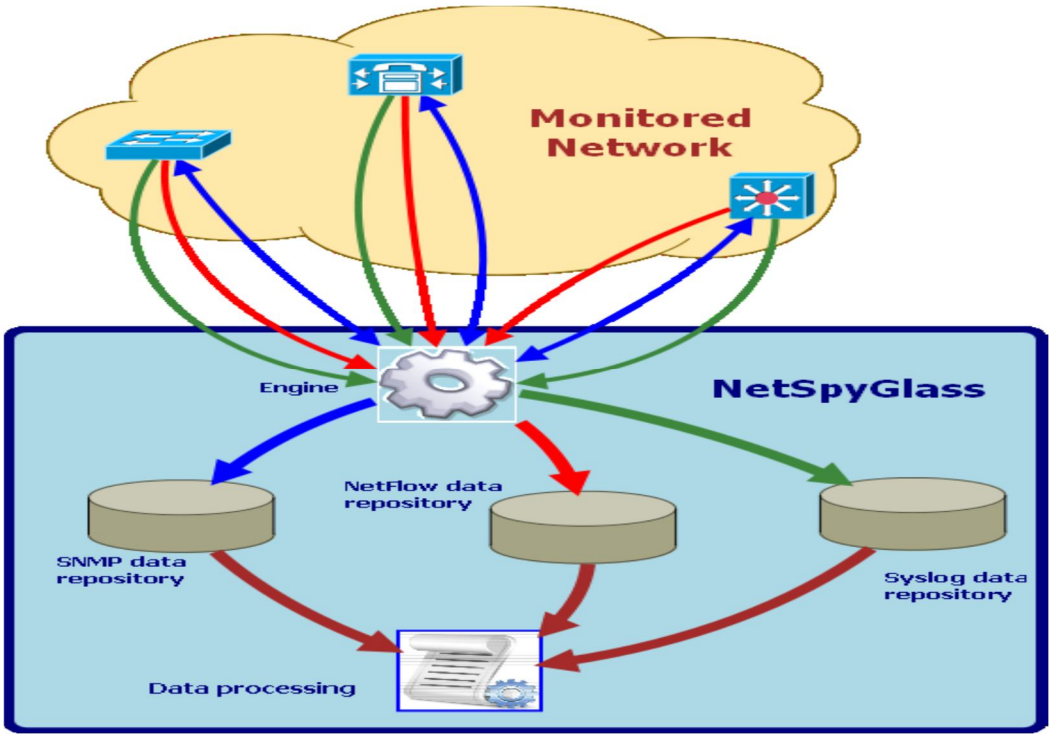
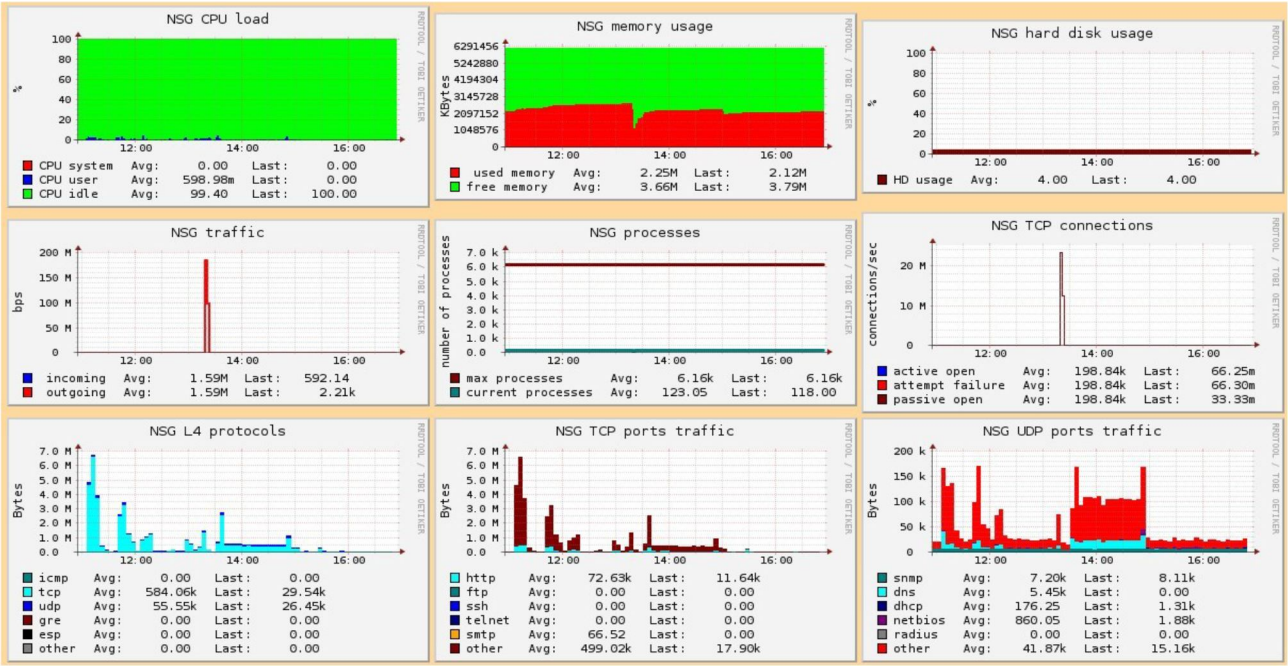


Figura 2.2: schema del funzionamento di Netspyglass

Come si può vedere, ogni device della rete in esame comunica con Nestpyglass, più precisamente con l'engine, che gestisce tutte le procedure relative alle funzioni elencate in precedenza. L'engine raccoglie periodicamente i dati di ogni device e dopo averli analizzati li immagazzina negli appositi repository, dai quali possono essere usati dall'utente, oltre ad inviare eventuali messaggi di allarme/avviso.

Attraverso una GUI inclusa nel tool è possibile accedere agli stati di ogni singolo device, la relativa interfaccia, le informazioni sugli andamenti del traffico, le diverse componenti di ogni singolo traffico (html, ftp…), eventuali eventi e messaggi inviati.

Per facilitare il capacity monitoring è inoltre possibile creare dei grafici relativi ai dati raccolti, in modo da poter visualizzare l'andamento di grandezze quali l'utilizzo di risorse (nel nostro caso CPU) o l'andamento del traffico.



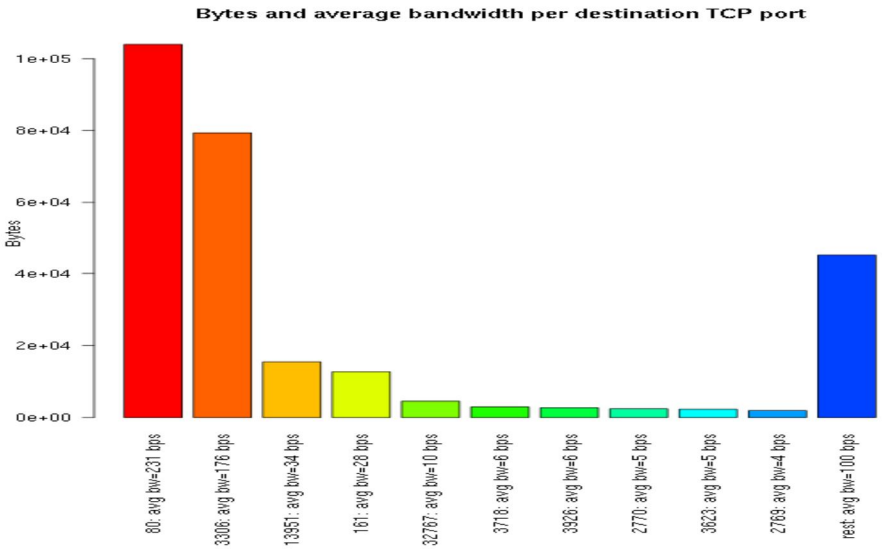


Figura 2.3 esempio delle funzionalità di Netspyglass: dashboard su traffico e allocazione di risorse e percentuali dei diversi protocolli formanti il traffico

2.1.2 Processing dei dati: RRDtool

Per la raccolta dei dati vera a propria Netspyglass fa uso di un ulteriore programma, chiamato RRDtool.

RRDtool, acronimo per Round Robin Database tool, è uno strumento sviluppato da Tobias Oetiker, utile nel nostro caso per poter registrare dati provenienti da serie temporali. I dati vengono inseriti in un database circolare a dimensione fissa (da qui il termine Round Robin).

Normalmente, se si volesse salvare indefinitamente i dati raccolti da una serie temporale, si dovrebbe incrementare periodicamente la grandezza dell'archivio, in modo da aver continuamente spazio sufficiente per immagazzinare i samples via via ottenuti, oppure eliminare i dati meno recenti. In un archivio di tipo Round Robin invece, quando i dati stanno per riempire completamente il database, si procede a liberare parte del database sostituendo i dati meno recenti con la loro media. Ad esempio, all'interno di questo capitolo, i dati considerati vengono raccolti con un periodo di 60 secondi; quando l'archivio risulterà essere quasi completamente occupato, i primi samples verranno mediati a gruppi di 6 elementi: in questo caso quindi ogni 6 dati precedentemente immagazzinati si otterranno 5 ulteriori posti liberi per i dati successivi. Quando anche con questo modo il database sarà quasi pieno si provvederà a mediare nuovamente i dati meno recenti, incrementando il numero di dati per ciascuna media; in questo modo il database sarà sempre almeno in parte libero, pur mantenendo una dimensione costante, dipendente dal numero di campionamenti che si vogliono memorizzare.

Il lato negativo di questo procedimento consiste quindi nella perdita dei dati raccolti meno recentemente in favore della loro media. Un metodo intuitivo per evitare questo problema consiste nel fare in modo che la grandezza del database sia equivalente al numero di samples che si è interessati ad avere: nel nostro caso si useranno dati per quattro settimane (per fare previsione short-range) e li si raccoglierà con periodicità pari a 60 secondi, quindi si sceglie di usare un database che riesca a contenere 40320 samples (numero totale di minuti in 4 settimane) prima di dover ricorrere all'operazione di mediazione. Per quanto riguarda la previsione long-range, dovendo predire un orizzonte settimanale, utilizzare dati mediati non risulta essere un problema, in quanto ci si focalizzerà sull'andamento generale del segnale di volta in volta considerato e non sul valore puntuale assunto minuto per minuto.

Esistono diversi tipi di database definibili da RRDtool, quali:

- COUNTER: usato per grandezze di cui interessa in particolare l'incremento ad ogni istante di tempo. La grandezza misurata viene salvata come incremento/tempo, quindi nel nostro caso sarà misurata in Mb/s.
- DERIVE: database i cui elementi rappresentano ad ogni istante la derivata del segnale misurato, partendo dall'istante iniziale a quello corrente. Ogni sample rappresenta quindi la derivata aggiornata all'ultimo istante temporale disponibile.
- ABSOLUTE: usando questo tipo di database i valori che si leggono vengono resettati ad ogni nuova lettura. Si usa questo tipo di database quando si stima che il counter da cui si legge la misura abbia la tendenza ad avere overflow (non sarà impiegato in seguito).
- GAUGE: l'unico tipo di database a non lavorare sugli incrementi dei valori misurati ma sui valori stessi. Usato quindi quando ciò che interessa non è tanto il valore incrementale di una grandezza quanto la sua misura.

- COMPUTE: database usato per salvare il risultato di un operazione fatta su valori di altri database. Se, ad esempio, a partire da un database di tipo counter si vogliono ottenere dei valori che non possono essere ottenuti tramite le consolidaton function (definite in seguito), si definisce l'operazione desiderata tramite una RPN-expression (termine proprio di RRDtool che sta ad indicare una sequenza di operazioni non elementare) e si crea un nuovo database attraverso il comando COMPUTE per salvare i risultati voluti.

Nel nostro caso scegliamo di usare database di tipo GAUGE per l'utilizzo di CPU e COUNTER per il traffico. Questo poiché per quanto riguarda la CPU siamo interessati ai valori puntuali e non agli incrementi: ciò che deve essere monitorato non è l'incremento minuto per minuto, ma il valore che viene assunto; risulta utile quindi immagazzinare il valore totale in quanto esso è direttamente confrontabile con un eventuale soglia che si stabilisce (ad esempio all'interno di un capacity plan) la CPU non debba oltrepassare. Per il traffico si è scelto COUNTER in quanto tipicamente grandezze come i flussi e i traffici SNMP vengono sempre immagazzinati considerando i valori incrementali. In appendice 2.1 è presente il codice usato per creare il database desiderato usando RRDtool.

In questo modo vengono immagazzinati i dati grezzi, cioè così come sono raccolti, che vengono definiti RRDtool primary data. A partire da essi si possono ottenere, mediante funzioni dette consolidation function, dei dati rielaborati, detti consolidated data. Le consolidation function sono:

- AVERAGE: la media dell'incremento dei dati. Se ad esempio vogliamo calcolare la media lungo 4 elementi, la funzione prenderà il primo e il quarto elemento, sottrarrà il quarto al primo e dividerà per il lasso di tempo trascorso (quindi ad esempio nel nostro caso 4 minuti).
- MIN: restituisce il minimo incremento dei dati tra quelli presenti nella finestra considerata. Se ad esempio consideriamo come sopra l'incremento ogni 4 elementi e una finestra di 10 step, la funzione calcolerà l'incremento medio (come AVERAGE) scorrendo tutta la finestra, selezionando poi, tra tutti i valori ottenuti, quello minimo.
- MAX: analoga alla funzione MIN, restituisce il valore massimo al posto del minimo.
- LAST: restituisce il valore dell'ultimo incremento.

Come si può notare tutte le consolidation function restituiscono valori relativi all'incremento dei dati raccolti e non relativi ai dati raccolti stessi, per cui, a partire dai dati consolidati, si è dovuto ricavare i dati grezzi attraverso l'uso di uno script Matlab.

Si è scelto di usare sempre consolidation function di tipo AVERAGE. Di nuovo si rimanda all'appendice 2.1 per maggiori informazioni sulla sintassi dei comandi da usare in ambiente RRDtool.

2.1.3 Conversione dei dati

Immagazzinati i dati necessari per lo sviluppo di questa tesi, si deve ora procedere a convertirne il formato per poterli utilizzare in Matlab. RRDtool fornisce i dati in file con un formato proprio (rrd), attraverso una funzione integrata da rrdtool stesso (funzione fetch) è possibile ricondursi ad un formato txt. Ottenuto tale formato, si ha bisogno di un ulteriore modifica affinché i dati possano essere caricati correttamente in Matlab; per poter giungere a un formato finale appropriato si è utilizzato un apposito script.

2.2 **Determinazione delle componenti dei segnali**

Convertiti i dati in un formato utilizzabile in ambiente Matlab si può ora procedere a studiare gli andamenti di traffico e CPU.

Si ricorda l'ipotesi, fatta a inizio capitolo, che si possa considerare un unico traffico, ottenuto dalla somma dei traffici delle singole interfacce.

Come detto si sceglie di usare i dati provenienti da un router a due interfacce, in particolare si è scelto il router denominato “Adderley”. In router di questo tipo, generalmente tutto il traffico entrante nel router dall’interfaccia 1 viene immesso sull’interfaccia 2 e viceversa, essendo le interfacce percorribili in entrambi i sensi (full duplex), anche se nulla impedisce che il traffico in uscita venga immesso sulla stessa interfaccia da cui proviene, in base alle tabelle di routing memorizzate dal router considerato. Considerando un unico traffico in entrata e uno in uscita non risulta di interesse quale sia l’interfaccia su cui un pacchetto viene indirizzato, ma solo che tale pacchetto debba essere processato dal router.

L’andamento di traffico e CPU per questo particolare router è il seguente:

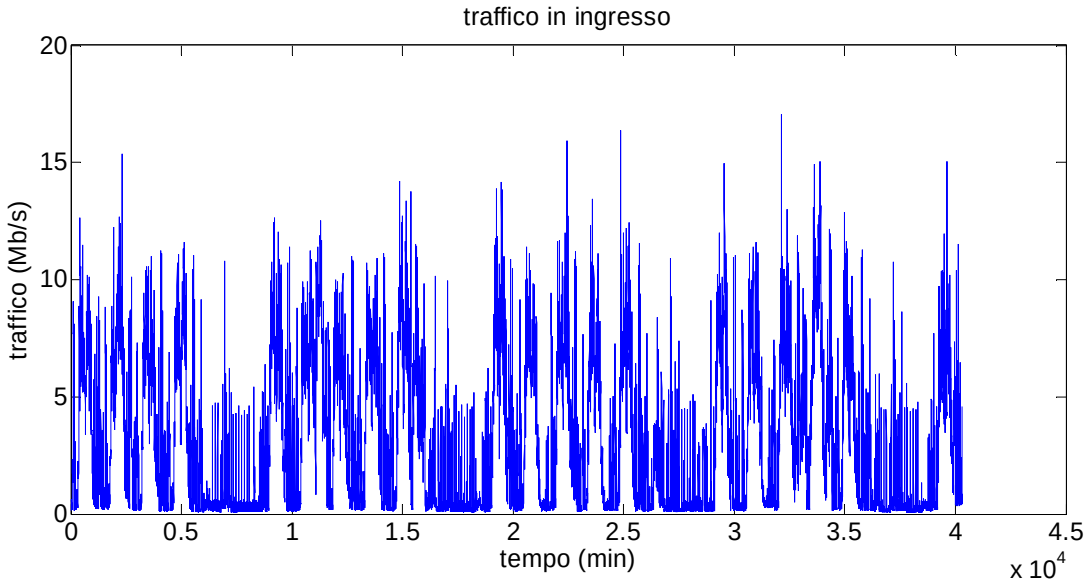


Figura 2.4: traffico in ingresso al router Adderley in un lasso di tempo pari a 4 settimane

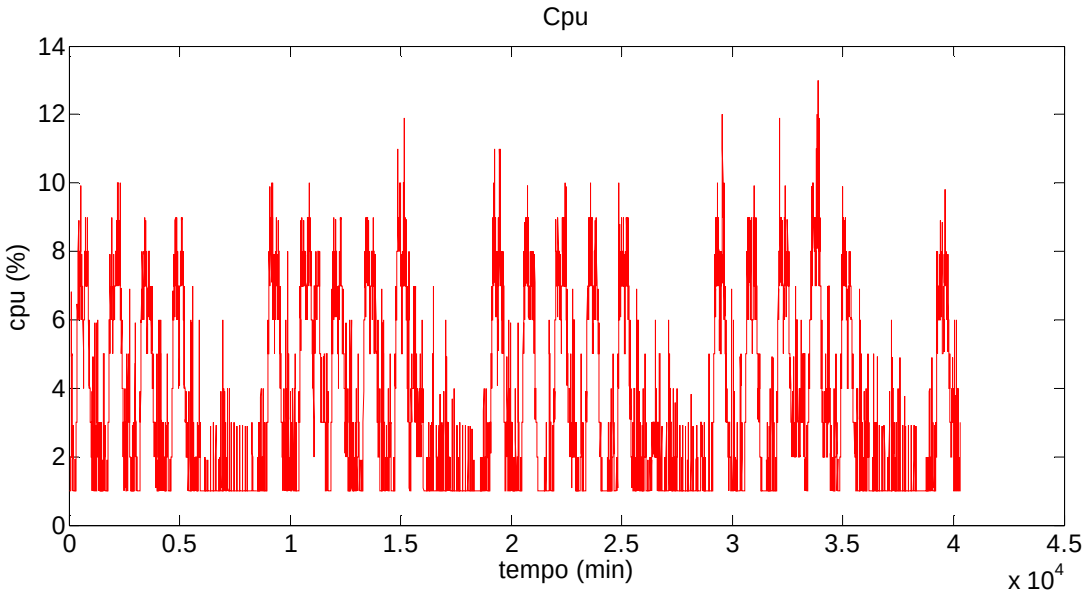


Figura 2.5: utilizzo di CPU relativo al router Adderley in un lasso di tempo pari a 4 settimane

I dati sono relativi a un mese di monitoring, campionati ogni 60 secondi (da qui il fatto che sull'ascissa si abbiano 40320 istanti di tempo, corrispondenti al numero di minuti in 4 settimane). Il periodo considerato va dalle 2:31 di martedì 7 maggio alle 2:30 lunedì 4 giugno, per questo nel grafico si ha in corrispondenza dell'ultimo giorno un andamento del traffico pari a quello di un giorno lavorativo e non festivo, mentre la prima settimana sembra avere un giorno in meno.

Come si può notare è presente una marcata stagionalità settimanale: nei giorni dal lunedì al venerdì il traffico cresce nelle prime ore del mattino, ha una diminuzione nelle ore centrali della giornata, cresce di nuovo nel pomeriggio per poi scendere nelle ore serali e mantenersi a livello basso nelle ore notturne. Questo fenomeno è considerevolmente meno marcato nei dati corrispondenti al week-end, in cui il traffico si mantiene pressoché costante, fatta eccezione per alcune spikes di breve durata (si veda la figura 2.6).
Considerazioni analoghe valgono per l'utilizzo di CPU.

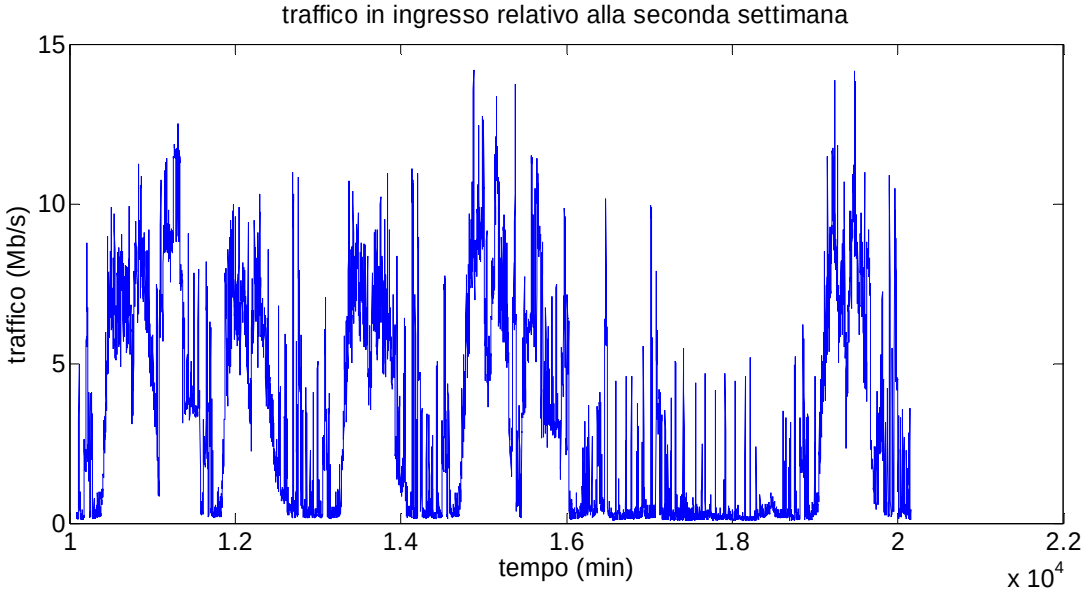


Figura 2.6: seconda settimana di traffico in ingresso al router, in cui è possibile osservare la presenza di andamenti giornalieri nei giorni lavorativi

Si è perciò scelto di considerare sia traffico (u) che CPU (y) come la somma di 3 contributi:

$$u = \bar{u} + s_u + \delta u$$

$$y = \bar{y} + s_y + \delta y$$

in cui:

- $\bar{y}$ e $\bar{u}$ corrispondono ai trend presenti nei dati.
- s_y e s_u corrispondono alle componenti stagionali di cui si è parlato.
- δy e δu identificano le componenti rimanenti di CPU e traffico.

Le prime due componenti sono deterministiche e possono essere ricavate direttamente dai dati, la terza è invece di natura stocastica e viene va ricavata come differenza tra il segnale totale e le componenti deterministiche. Dal momento che l'obiettivo è caratterizzare la relazione tra tutte le componenti, dobbiamo per prima cosa determinare i valori di trend e stagionalità di traffico, in modo da poter ottenere dei dati depurati, per poi andare a identificare la relazione ingresso-uscita del modello avente come per entrata il traffico e come uscita l'utilizzo di CPU.

2.2.1 Identificazione del trend

Per determinare il trend (o in assenza di esso, la media dei dati) si utilizzano le funzioni Polyfit e Polyval presenti in Matlab. La prima consente di caratterizzare i trend di un ordine voluto (lineare, quadratico, …) restituendo i coefficienti dei polinomi che ne descrivono il relativo andamento; la seconda, a partire dai coefficienti restituiti dalla prima, ricrea tale andamento per il periodo di tempo voluto.

Osservando i dati, ci si aspetta un trend praticamente nullo; scegliamo quindi di considerare un trend lineare, utilizzando tutte le 4 settimane di dati che si ha a disposizione.

I coefficienti (da intendere nella forma *y=at+b,* dove *t* è il tempo in secondi) ottenuti per il traffico in ingresso sono:

$$\text{a} = -3.4 \cdot 10^{-5}[Mb/s^2] \quad \text{b} = 3.53[Mb/s]$$

Mentre quelli ottenuti per l'utilizzo di CPU:

$$\text{a} = -2.6 \cdot 10^{-5}[\%/s] \quad \text{b} = 3.64[\%]$$

Come ipotizzato osservando i grafici, il trend presente è praticamente nullo in entrambi i casi (si noti che la differenza di ordine tra il primo coefficiente, che rappresenta il coefficiente angolare, e il secondo che rappresenta l'intercetta); dal momento che si considerano 4 settimane, però, esso ha comunque un contributo sulla lunga distanza, per cui risulta corretto considerarlo all'interno di questa trattazione.

Per essere sicuri che il trend sia effettivamente lineare, si sceglie di riutilizzare la funzione richiedendo un trend di ordine 2: se il primo coefficiente restituito fosse zero, ciò significherebbe che il polinomio corrispondente sarebbe di nuovo del prim'ordine.

I nuovi coefficienti ottenuti sono ($y = at^2 + bt + c$):

- per il traffico in ingresso: $a = -1.31 \cdot 10^{-9} \left[\frac{Mb}{s^3}\right]$ $b = 5.46 \cdot 10^{-5} \left[\frac{Mb}{s^2}\right]$ $c = 3.33 \left[\frac{Mb}{s}\right]$
- per l'utilizzo di CPU: $a = -9.97 \cdot 10^{-9} \left[\frac{\%}{s^2}\right]$ $b = -4.39 \cdot 10^{-5} \left[\frac{\%}{s}\right]$ $c = 3.47[\%]$

Da cui si può notare come in entrambi i polinomi il primo coefficiente siano circa pari a zero, confermando quindi quanto detto sopra. Identificato il trend, possiamo depurare i segnali di traffico e CPU, ottenendo i seguenti risultati:

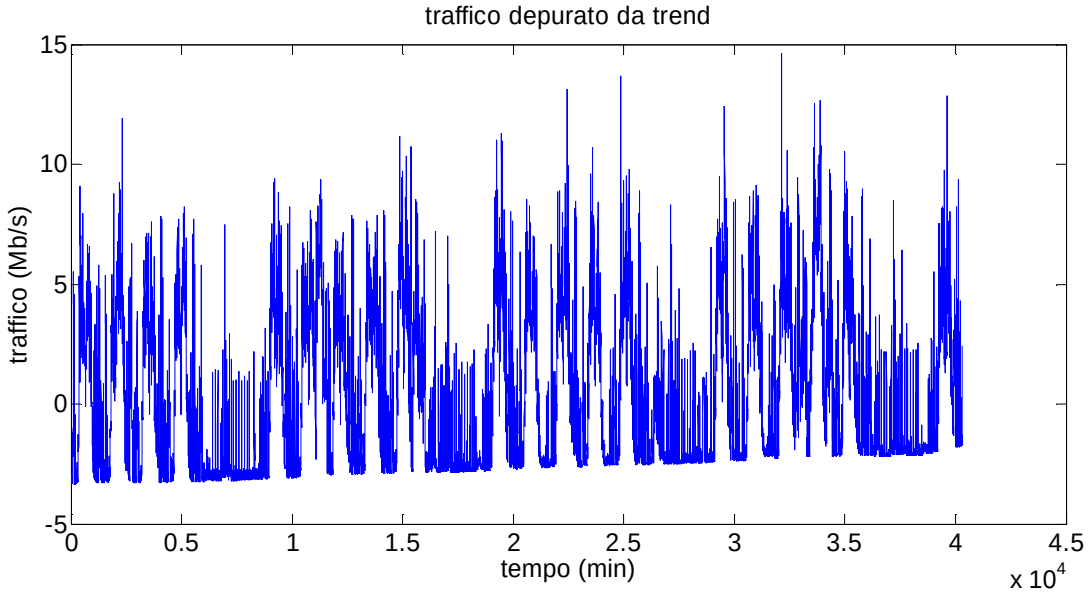


Figura 2.7: traffico depurato dal trend lineare precedentemente identificato

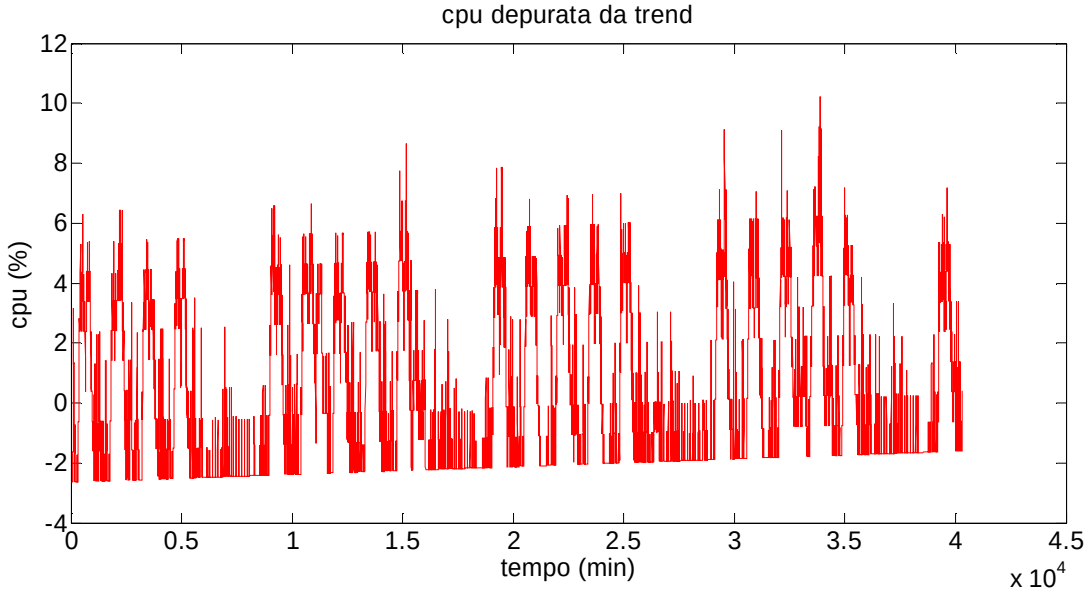


Figura 2.8: CPU depurata dal trend lineare precedentemente identificato

Terminata questa parte siamo ora in grado di calcolare la stagionalità dei segnali.

2.2.2 Identificazione della stagionalità

Sapendo che la stagionalità ha periodicità settimanale e che abbiamo a disposizione dati per 28 giorni, calcoliamo la media di ogni minuto nella settimana sommando il contributo di ogni settimana e dividendo poi per il numero di settimane.

In questo modo otteniamo la stagionalità sia per il traffico che per l'utilizzo di CPU, come mostrato in figura 2.9 e 2.10.

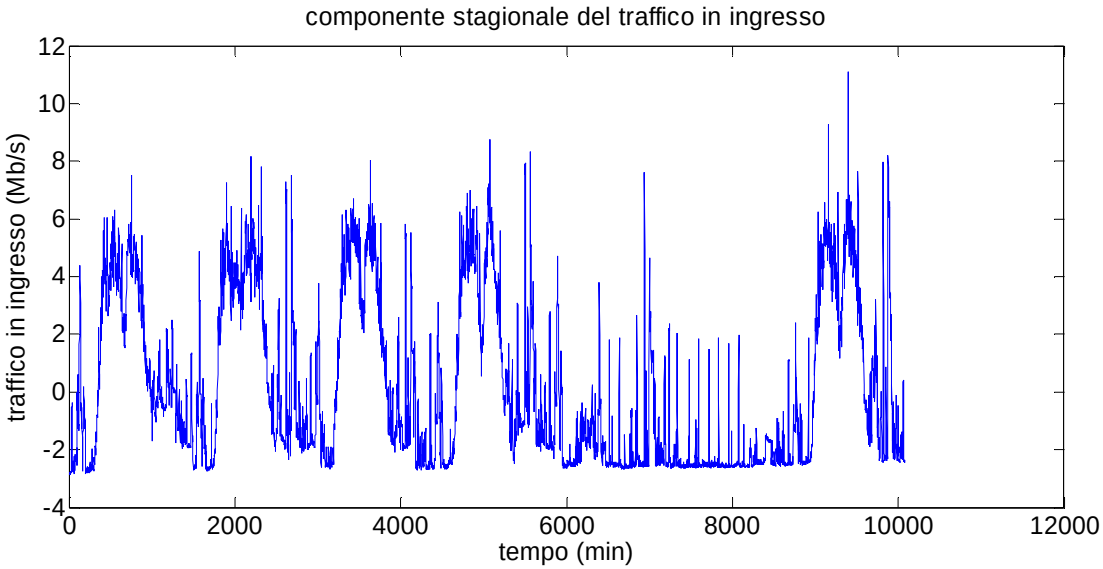


Figura 2.9: contributo stagionale del traffico, identificato utilizzando le 4 settimane a disposizione

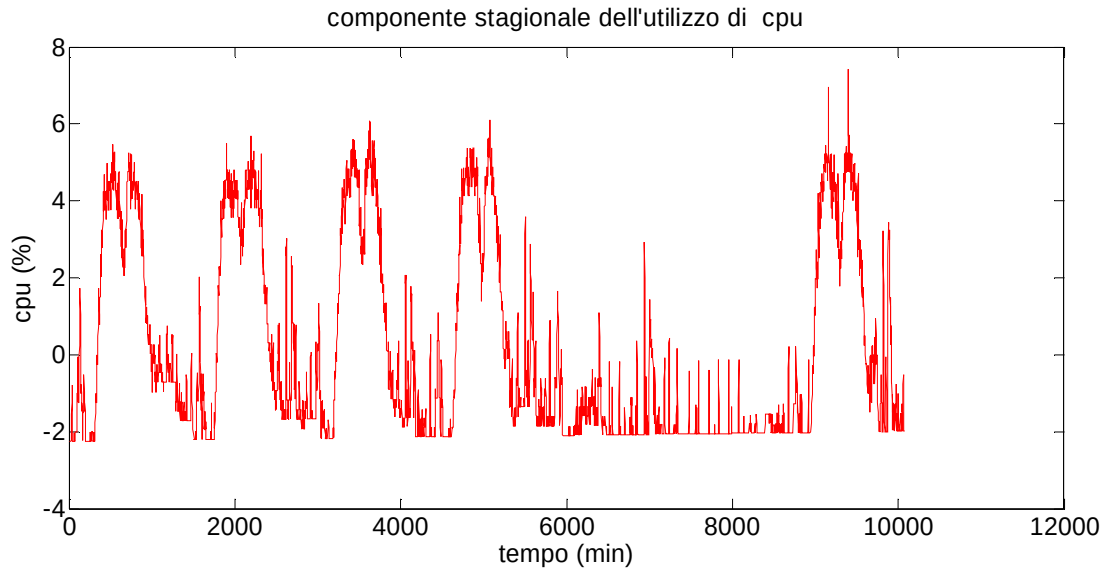


Figura 2.10: contributo stagionale dell'utilizzo di CPU, identificato utilizzando le 4 settimane a disposizione

Di seguito sono riportati i confronti delle stagionalità con i segnali depurati solamente dal trend:

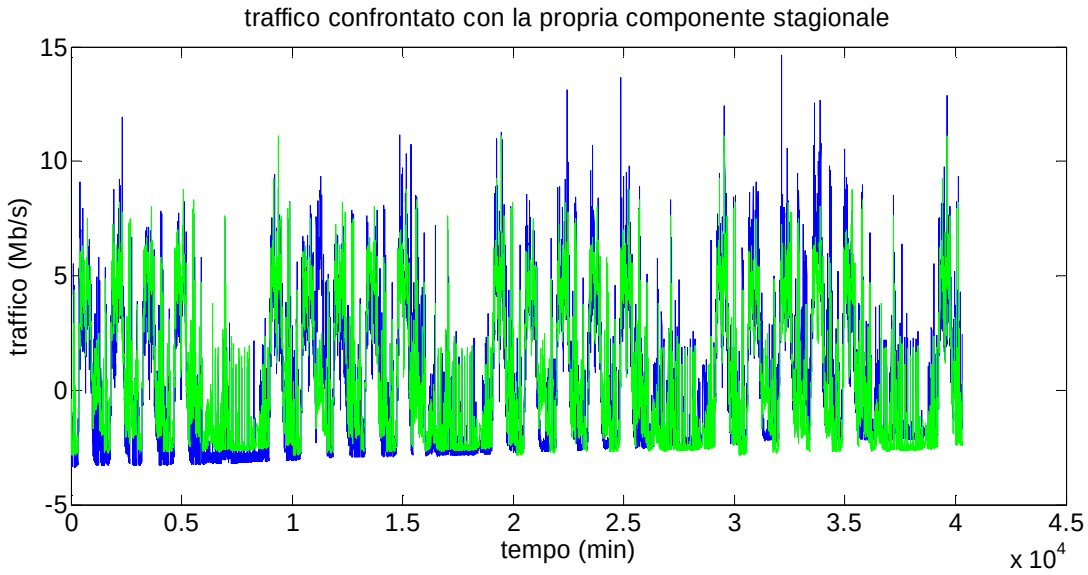


Figura 2.11: confronto tra traffico depurato da trend e relativa componente stagionale

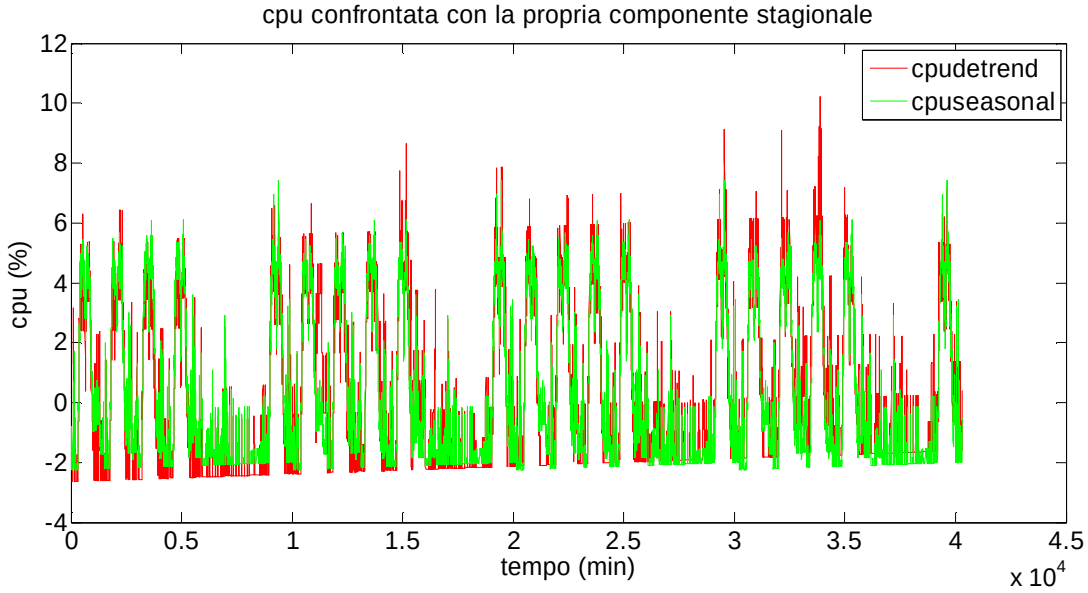


Figura 2.12: confronto tra CPU depurata da trend e relativa componente stagionale

A questo punto siamo in grado di ricavare le componenti δy e δu, cioè la parte stocastica di traffico e CPU:

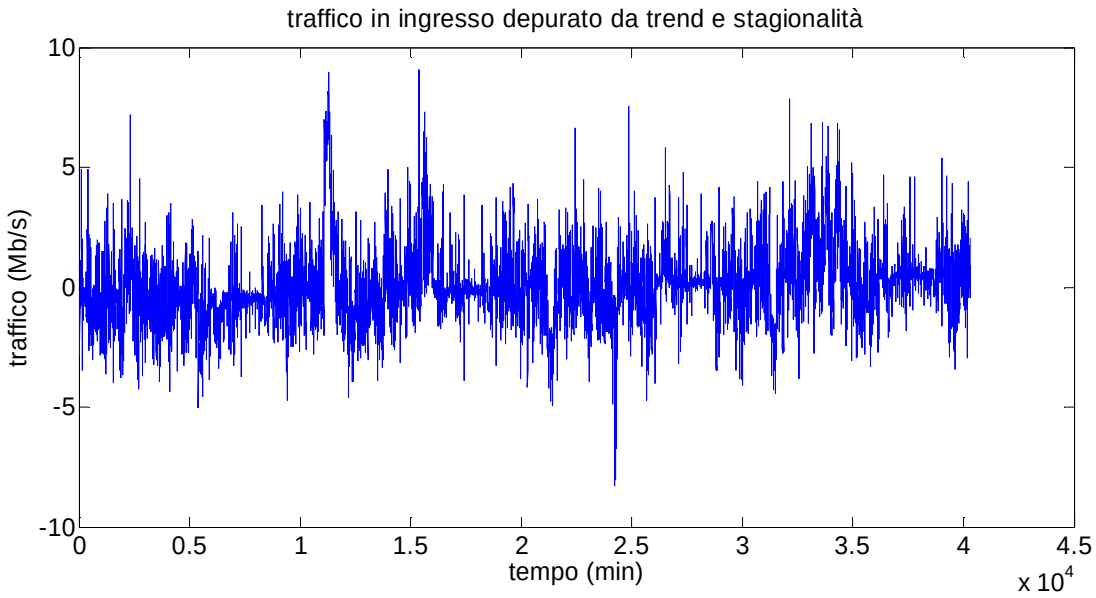


Figura 2.13: componente stocastica del traffico, ricavata come differenza tra segnale originale e le componenti deterministiche

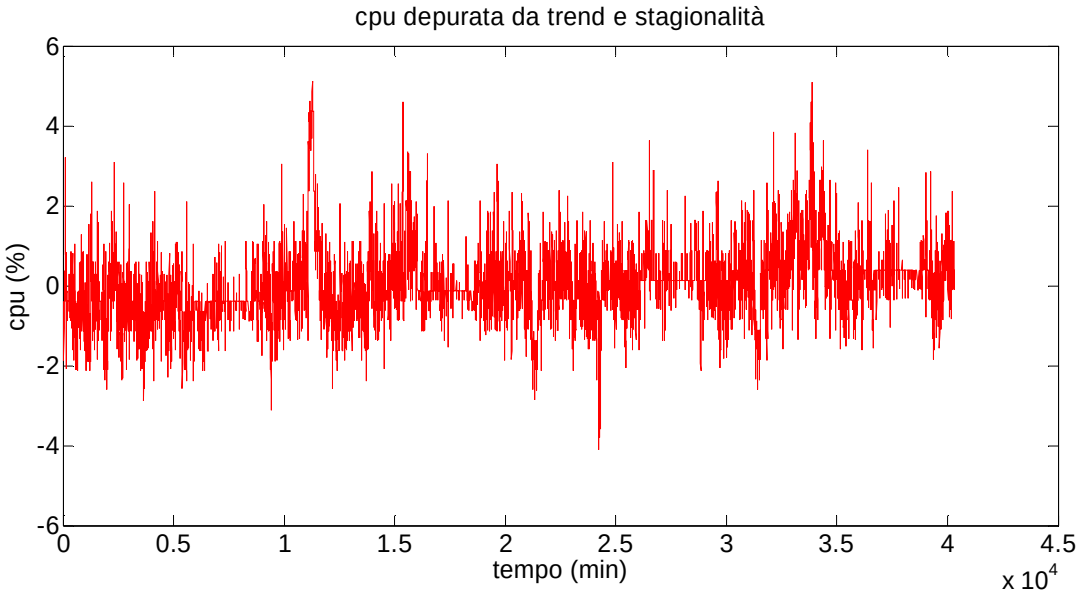


Figura 2.14: componente stocastica dell'utilizzo di CPU, ricavata come differenza tra segnale originale e le componenti deterministiche

È verificabile che quest'ultima componente del segnale abbia ora una media pari a zero e sia un processo stazionario.

2.3 **Modello a tre coefficienti**

2.3.1 Identificazione

Suddivisi i segnali nelle loro componenti si può ora procedere ad identificare il modello che caratterizzi la relazione tra traffico in entrata (espresso in Mb/s) e CPU utilizzata (in percentuale).

Per far questo, a partire dalle relazioni:

$$\delta y = K_c \delta u$$

$$s_y = K_s s_u$$

$$\bar{y} = K_t \bar{u}$$

si devono identificare i parametri K_c, K_s, K_t e calcolare l'errore tra le uscite $\delta\hat{y}$, $\hat{s}_y$, $\hat{\bar{y}}$ e i dati reali δy, s_y e $\bar{y}$. Per prima cosa viene decisa la quantità di dati da usare per l'identificazione e quella per la validazione.

Si considerano 3 casi:

	Settimane per l'identificazione	Settimane per la validazione
Caso 1	3	1
Caso 2	2	2
Caso 3	1	3

Si è scelto di trattare nello specifico il primo dei 3 casi, per poi riassumere più sinteticamente i 2 successivi.

Dividiamo quindi il set di dati in un primo insieme di 30240 sample, (pari a 3 settimane di dati) da usare per identificare i 3 parametri, e in un secondo formato dai restanti 10080 samples per la validazione.

Per poter identificare i 3 K_i^o si sceglie di considerare la cifra di merito *J (Loss function)* [4]:

$$J = \frac{1}{N}\sum_{i=1}^{N}(\delta y_i - \delta\hat{y}_i)^2$$

$$J = \frac{1}{N}\sum_{i=1}^{N}\left(s_{y,i} - \hat{s}_{y,i}\right)^2$$

$$J = \frac{1}{N}\sum_{i=1}^{N}(\bar{y}_i - \hat{\bar{y}}_i)^2$$

In cui N pari a 30240.

Minimizzando tali cifre si ricavano i valori ottimiK_c^o, K_s^o e K_t^o.

Considerando per esempio la prima cifra di merito:

$$J = \frac{1}{N}\sum_{i=1}^{N}(\delta y_i - \delta\hat{y}_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(\delta y_i - K\delta u_i)^2$$

Derivando si ottiene:

$$\frac{\partial y}{\partial K} = \frac{2}{N}\sum_{i=1}^{N}(\delta y_i - K\delta u_i)(-\delta u_i) = \frac{2}{N}\sum_{i=1}^{N}\left(K\delta {u_i}^2 - \delta y_i \delta u_i\right)$$

E ricavando infine i valori ottimali dei coefficiente:

$$K_c^o = \sum_{i=1}^{N}\delta y_i \delta u_i \Big/ \sum_{i=1}^{N}\delta {u_i}^2 = 0.5707[\%/(\text{Mb/s})]$$

$$K_s^o = \sum_{i=1}^{N} s_{y,i} s_{u,i} \Big/ \sum_{i=1}^{N} {s_{u,i}}^2 = 0.8655[\%/(\text{Mb/s})]$$

$$K_t^o = \sum_{i=1}^{N}\bar{y}_i \bar{u}_i \Big/ \sum_{i=1}^{N}{\bar{u}_i}^2 = 1.0755[\%/(\text{Mb/s})]$$

In questo modo abbiamo ottenuto i 3 valori ottimali K_c^o,K_s^o e K_t^o, corrispondenti ai valori di K_i tale per cui è minimo l'errore tra le varie componenti dell'utilizzo di CPU e le uscite del modello.

2.3.2 Validazione

Passiamo ora a considerare il set di dati di validazione, calcolando l'errore percentuale come:

$$err = \frac{1}{N}\frac{\sum_{i=1}^{N}(y-\hat{y})^2}{\text{var}(y)} \cdot 100 \qquad fitting = \left(1 - \frac{1}{N}\frac{\sum_{i=1}^{N}(y-\hat{y})^2}{\text{var}(y)}\right) \cdot 100$$

con N che in questo caso vale 10080 (corrispondente a una settimana di samples). Si calcola l'errore per ogni componente (andando quindi a sostituire ad y nella cifra di merito di volta in volta $\bar{y}$, s_y e δy) e per il segnale totale, come mostrato in tabella.

Fitting componente di trend (%)	99.04%
Fitting componente stagionale (%)	95.02%
Fitting componente stocastica (%)	88.35%
Fitting segnale totale (%)	93.84%

In figura 2.15, 2.16 e 2.17 si hanno i confronti grafici per quanto riguarda la componente stocastica e il segnale totale.

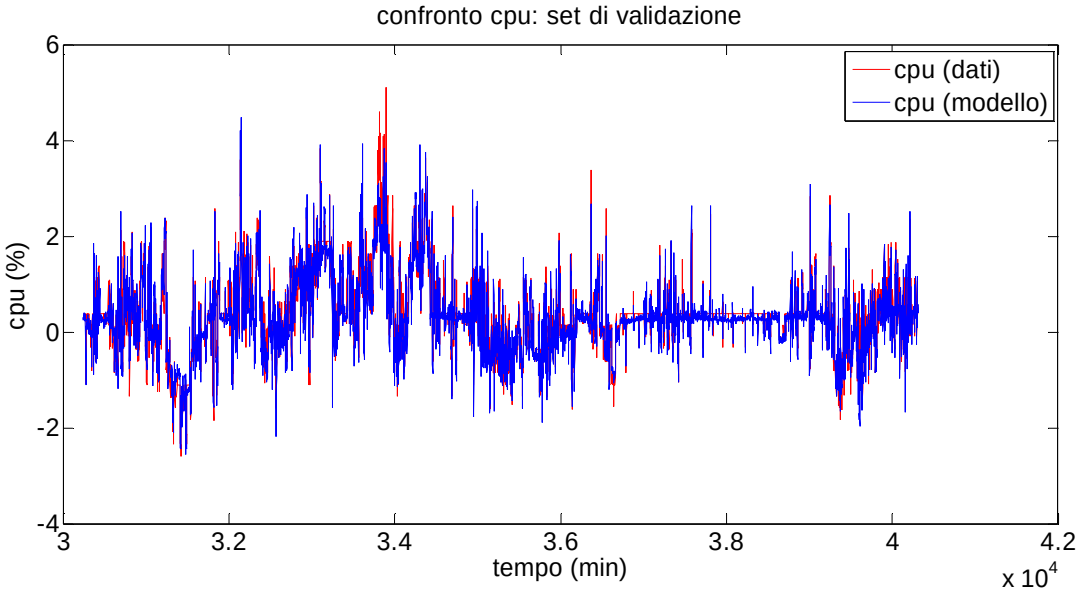


Figura 2.15: confronto tra la componente stocastica dell'utilizzo di CPU ricavata dai dati e l'uscita del modello

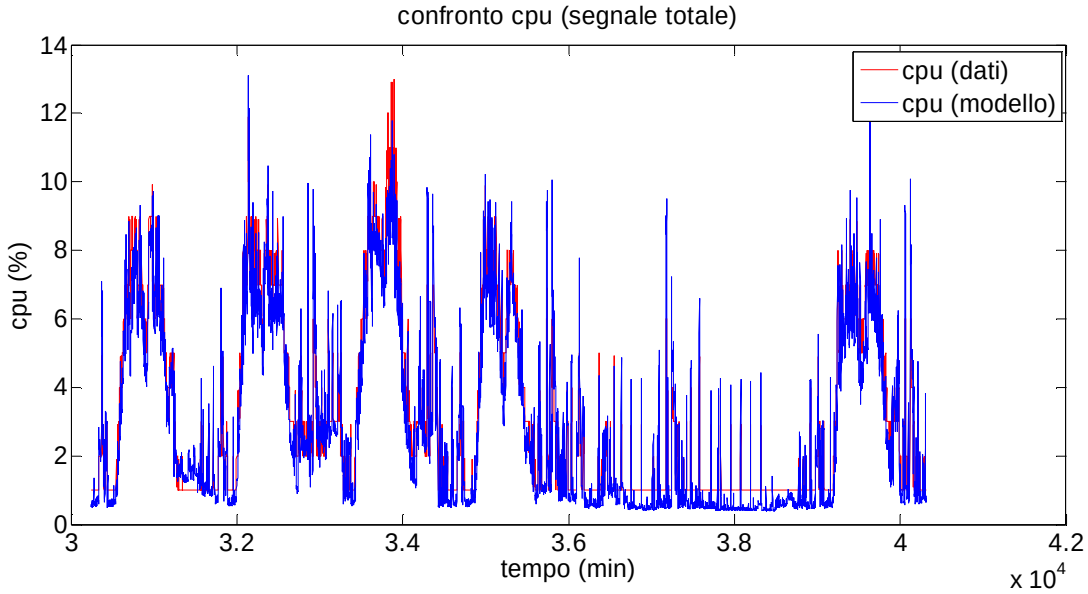


Figura 2.16: confronto tra CPU ed l'uscita del modello considerando il segnale totale

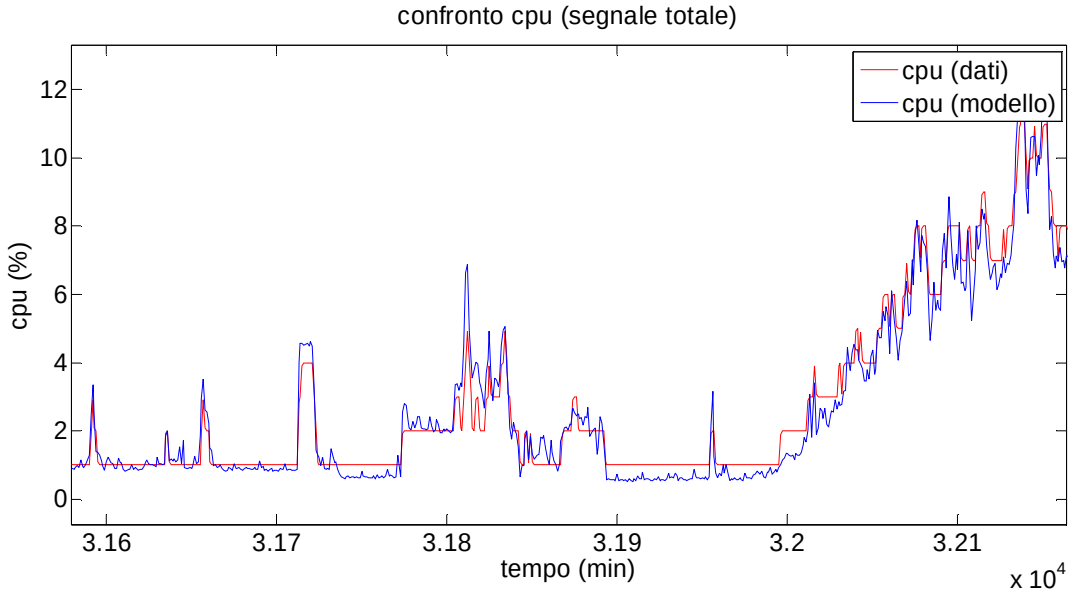


Figura 2.17: particolare del confronto (segnale totale), è possibile vedere come nei transitori il modello si sovrapponga ai dati

Si può notare come i due segnali siamo simili, in molti casi sovrapposti, e che in corrispondenza di picchi improvvisi del segnale a disposizione si hanno altrettanti picchi del segnale dato dal modello, a prova del fatto che il fitting del modello è particolarmente buono, a conferma di come sia possibile considerare statica la relazione tra traffico e CPU, poiché le dinamiche si esauriscono in un periodo inferiore a quello di campionamento. Da sottolineare il fatto che, mentre l'utilizzo di CPU non assume mai valori sotto l'1%, a causa del fatto che si ha sempre un utilizzo minimo di CPU dovuto ai processi interni di un router e quindi indipendenti dal traffico in ingresso, i valori assunti dal modello assumono valori in alcuni casi prossimi allo zero. Una possibile soluzione consiste nell'introduzione nel modello di una saturazione, che impedisca all'uscita del modello di assumere i valori nella fascia compresa tra zero e uno:

$$\hat{Y}(t) = \begin{cases} \hat{Y}(t) & \text{se } \hat{Y}(t) \geq 1 \\ 1 & \text{se } \hat{Y}(t) < 1 \end{cases}$$

Dal punto di vista pratico, tale discrepanza va ad impattare in maniera minima sulle prestazione del modello, essendo lo scarto tra i 2 segnali comunque ridotto.

Si può quindi concludere che, per il router in considerazione, l'assunzione fatta a inizio capitolo di considerare un modello statico e lineare per descrivere la relazione tra somma del traffico in ingresso al router e conseguente utilizzo di CPU sia corretta.

2.3.3 Caso 2 e Caso 3

Terminato il caso 1, trattiamo ora sinteticamente i casi 2 e 3, in cui si considerano rispettivamente 2 e 1 settimana per l'identificazione e 2 e 3 settimane per la validazione. Ci si aspetta che, avendo meno dati per l'identificazione, le prestazioni del modello siano inferiori ma, visto la percentuale di errore e fitting del primo caso, comunque buone.

Il procedimento utilizzato è analogo a quello usato per il primo caso, riportiamo quindi in questa parte solamente i risultati ed i grafici relativi al confronto tra i segnali raccolti e in uscita dal modello (segnale totale), si rimanda all'appendice 2.2 per ulteriori approfondimenti.

	K_c^O [%/(Mb/s)]	K_s^O [%/(Mb/s)]	K_t^O [%/(Mb/s)]
Caso 2	0.5839	0.8655	1.0597
Caso 3	0.5957	0.8655	1.0447

	Fitting componente stocastica [%]	Fitting stagionalità [%]	Fitting trend [%]	Fitting segnale totale [%]
Caso 2	86.60	95.02	99.08	93.34
Caso 3	88.38	95.02	99.05	87.71

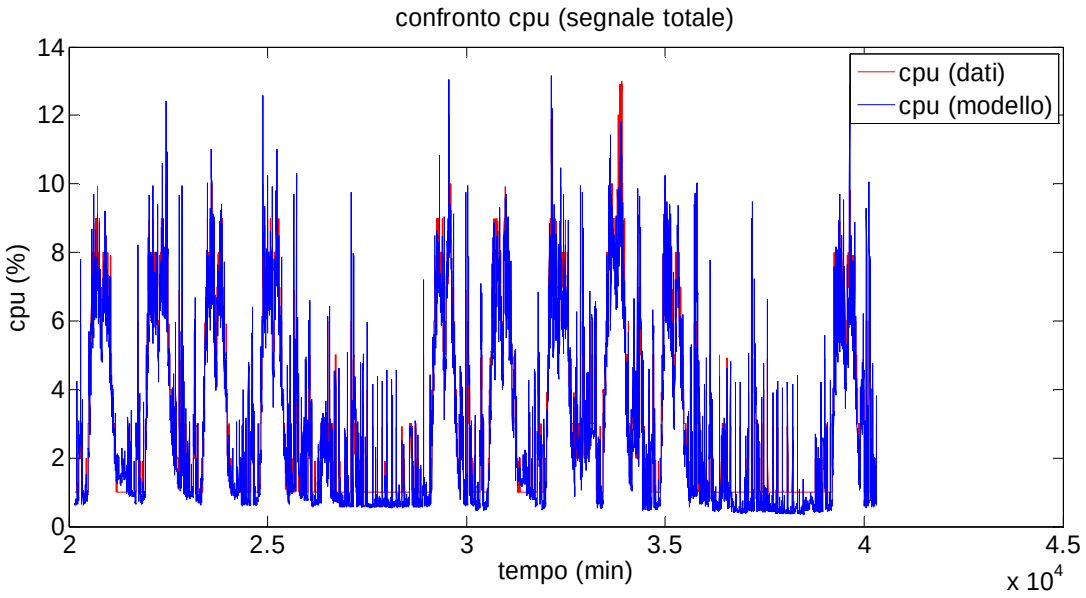


Figura 2.18 Caso 2: confronto utilizzando 2 settimane per l'identificazione e 2 settimane per la validazione

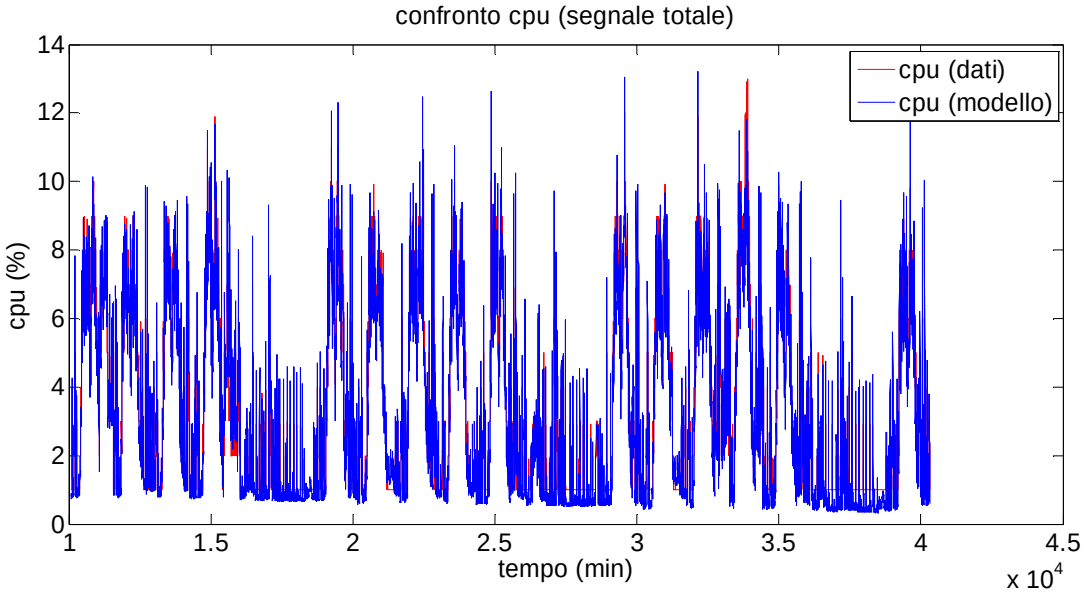


Figura 2.19 Caso 3: confronto utilizzando 1 settimane per l'identificazione e 3 settimane per la validazione

Osservando i valori di fitting sia nel caso in cui si tiene conto di stagionalità e trend, sia in quello in cui si considera il segnale depurato da queste due componenti, si può notare come essi siano simili al primo caso sopra trattato (3 settimane di identificazione).

Questo ci porta a concludere che ridurre il tempo dedicato all'identificazione del modello permette di ottenere, utilizzando meno dati e dovendo quindi attendere un lasso di tempo inferiore affinché essi siano disponibili, un modello con prestazioni pressoché identiche.

2.4 **Modello a coefficiente unico**

Si considera ora una variazione del problema presentato: dal momento che scegliendo di dividere il traffico si ottengono percentuali di fitting particolarmente soddisfacenti, ci si chiede quali sarebbero le prestazioni di un modello che considerasse il traffico entrante complessivo. Tale studio risulterà utile nel corso dei capitoli successivi. Ci si aspetta che, dal momento che si andrà ad identificare un unico coefficiente, le prestazioni siano inferiori a quelle appena ottenute.

Si sceglie perciò di considerare i segnali u e y e la relazione $y = Ku$ che abbiamo dimostrato caratterizzante il rapporto tra le due grandezze.

Per poter confrontare i risultati con i precedenti consideriamo gli stessi 3 possibili casi, in cui si considerano rispettivamente tre settimane per l'identificazione nel primo caso, due nel secondo e infine una nel terzo.

La cifra di merito da considerare è ora:

$$err = \frac{1}{N}\frac{\sum_{i=1}^{N}(y-\hat{y})^2}{\text{var}(y)} \cdot 100$$

Che risulta essere simile alla cifra di merito considerata alla fine della precedente trattazione. La differenza sta nel termine $\hat{y}$: mentre prima essa era visto come la somma delle 3 componenti di CPU, in cui ognuna era ricavata dalla corrispondente componente di traffico, ora essa viene ricavata direttamente dal segnale totale.

Ripercorrendo i passaggi per ricavare il coefficiente K, si ottiene (considerando il primo caso con 3 settimane di identificazione):

$$K^O = \frac{\sum_{i=1}^{N} y_i u_i}{\sum_{i=1}^{N} {u_i}^2} = 0.9385[\%/(\text{Mb/s})]$$

In figura 2.20 viene mostrato il confronto tra il segnale $\hat{y}$ ricavato attraverso il coefficiente e i dati reali y:

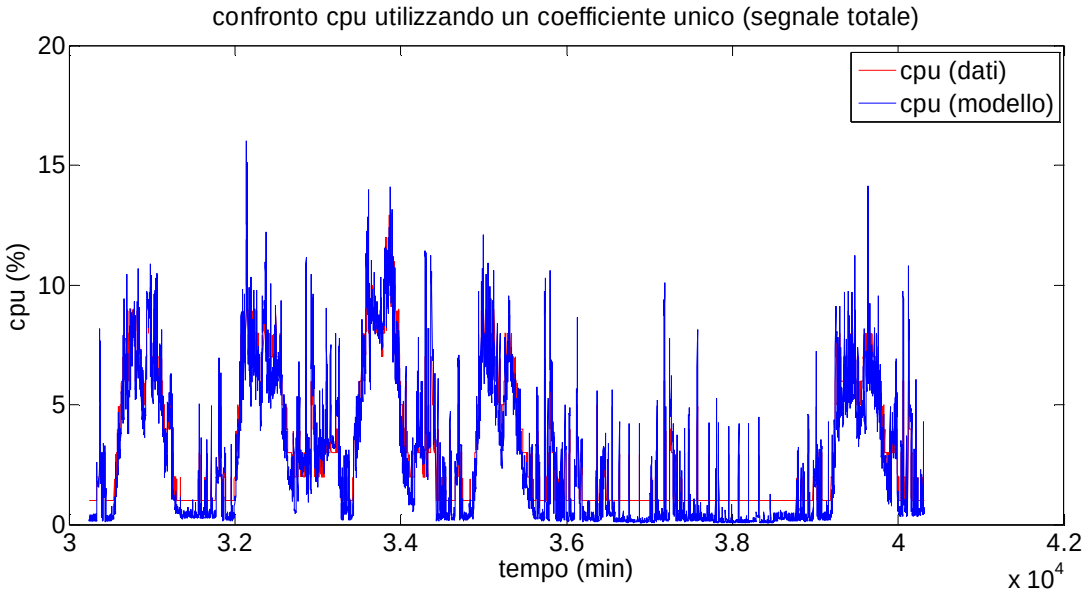


Figura 2.20: confronto dell'utilizzo di CPU ottenuto dal modello (a un unico coefficiente) con il set di dati a disposizione

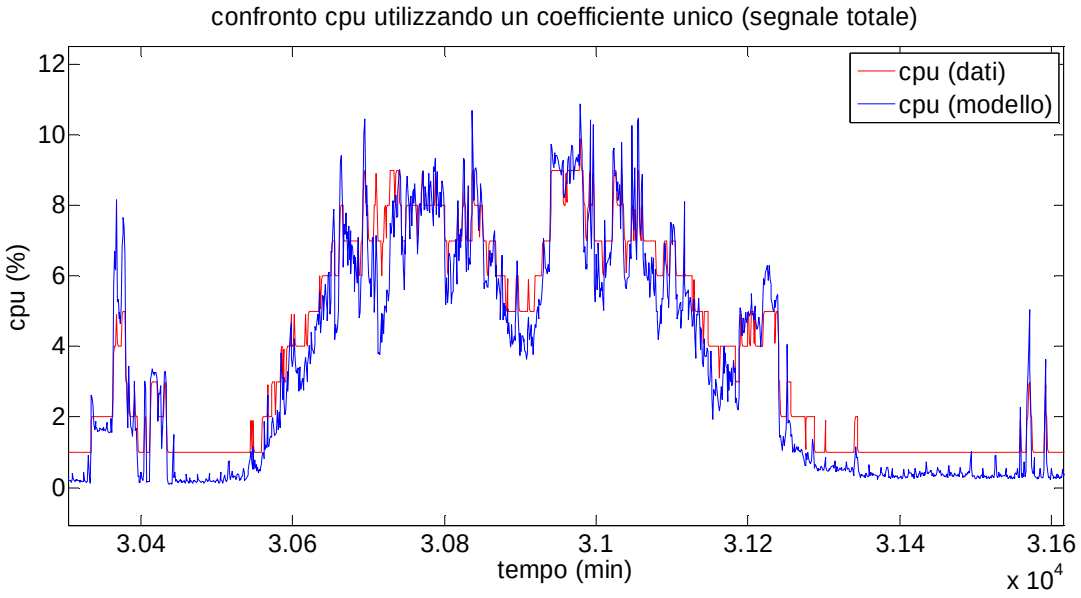


Figura 2.21: particolare del confronto tra modello e dati

Dal confronto grafico si nota come anche in questo caso si approssimi con fedeltà la realtà, avendo uno scarto dai dati ridotto. Rimane inoltre anche in questo caso la discrepanza tra modello e dati in corrispondenza di utilizzi di CPU sotto l'1%.

In tabella sono mostrati i risultati per tutti i 3 casi considerati:

	K^O [%/(Mb/s)]	Fitting [%]
Caso 1 (3 sett. / 1 sett.)	0.9385	87.49
Caso 2 (2 sett. / 2 sett.)	0.9245	87.71
Caso 3 (1 sett. / 3 sett.)	0.9361	87.31

Per terminare la trattazione, si confrontano i due diversi approcci tra di loro. Si sceglie come riferimento ottimale un ulteriore caso: si considera un coefficiente per la componente stocastica, in modo da ricavare a partire dal traffico il corrispondente utilizzo di CPU, e per le componenti deterministiche si considerano quelle ricavate dai dati di CPU, in modo da avere un segnale nella forma $\hat{y} = \delta\hat{y} + s_y + \bar{y}$. In questo modo, essendo due componenti coincidenti con i dati che si stanno usando, il relativo indice di fitting sarà sicuramente più alto di quelli relativi ai due approcci descritti. Tale aggiunta risulta utile per capire quanto i due approcci si discostino da questo caso considerato ottimale.

In tabella sono presenti i confronti tra le percentuali di fitting:

	Approccio 1 (3 coeff.)	Approccio 2 (1 coeff.)	Approccio ottimale
Caso 1 (3 sett. / 1 sett.)	92.91%	87.49%	98.75%
Caso 2 (2 sett. / 2 sett.)	93.34%	87.71%	98.82%
Caso 3 (1 sett. / 3 sett.)	93.39%	87.31%	98.83%

Come ci si poteva aspettare il terzo approccio ottiene i risultati migliori, avvicinandosi al 100%. Tra i due approcci sviluppati il primo, facendo uso di due coefficienti in più rispetto al secondo, permette di avere prestazioni superiori, avendo un errore pari a circa la metà. Tuttavia, vista la percentuale accettabile di fitting, anche il secondo risulta valido.

Per questo possiamo considerare entrambi gli approcci validi, e nel proseguimento di questa tesi saranno impiegati entrambi, a seconda che si stia considerando il segnale nella sua totalità o analizzando le singole componenti.

2.5 **Generalizzazione nel caso di router a più interfacce**

Fino ad ora sono stati considerati i dati provenienti dal router Adderley, router a 2 interfacce; lo stesso procedimento di identificazione e validazione viene ora applicato al caso di router con 3, 5 e 6 interfacce, poiché si vuole verificare che la conclusione che esista una relazione statica tra traffico e utilizzo di CPU in un router valga anche per router a più interfacce e non solamente a due.

Di seguito, per mostrare come ciò sia vero, vengono presi in esame i dati del router denominato Monk, avente 6 interfacce.

Anche in questo caso viene fatta l'assunzione che sia ancora possibile considerare come input del modello la somma dei traffici in ingresso su ogni interfaccia. Pertanto il traffico in ingresso appare come in figura 2.22.

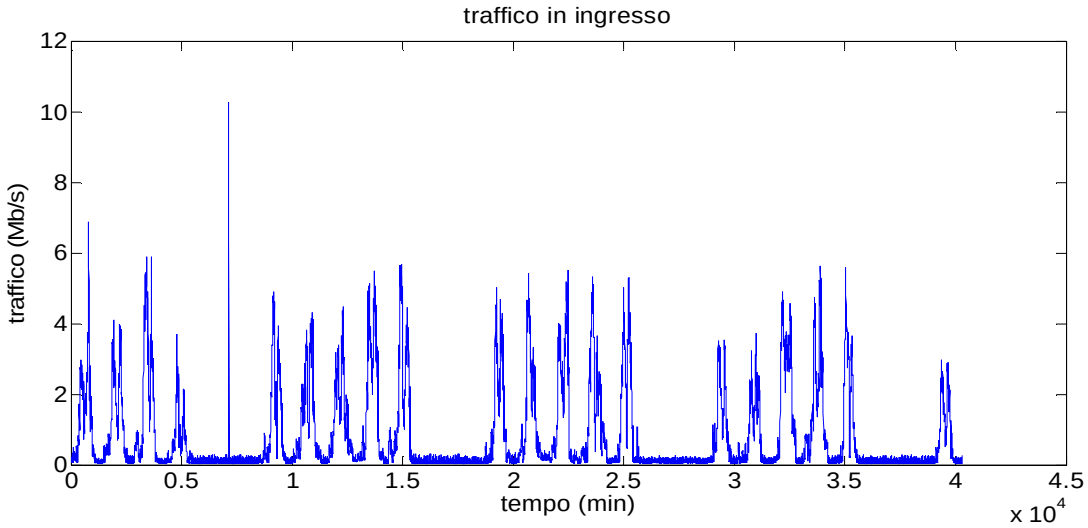


Figura 2.22: traffico in ingresso al router considerato

Mentre in figura 2.23 è mostrato l'andamento della CPU.

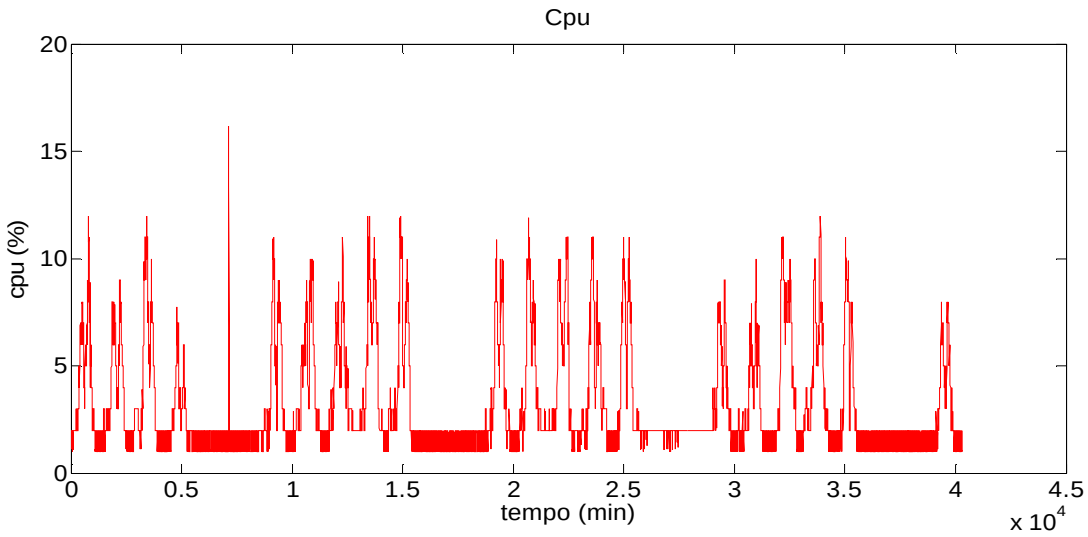


Figura 2.23: andamento della CPU per il router considerato

Similmente al caso del router Adderley è possibile riconoscere nei dati una stagionalità settimanale. Si considerano nuovamente due possibili modelli, il primo con tre coefficienti che modellizzano la relazione traffico-CPU per ogni componente e il secondo che tratta il segnale nella sua totalità. Inoltre anche in questo caso consideriamo tre diversi situazioni, in cui si varia il numero di settimane usate per l'identificazione e per la validazione come nel caso del router Adderley. Viene trattato qui il caso in cui si considerano 3 settimane per l'identificazione. Per questo motivo, per prima cosa traffico e CPU vengono scomposti nelle tre componenti di trend, stagionalità e stocastica (figure 2.24 e 2.25).

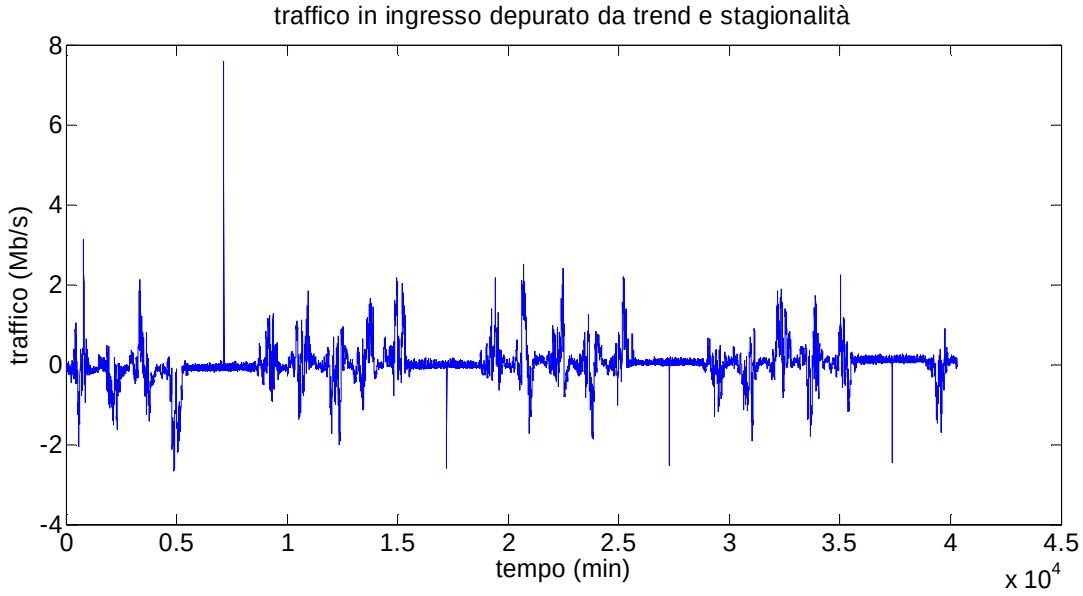


Figura 2.24: andamento della componente stocastica del traffico in ingresso

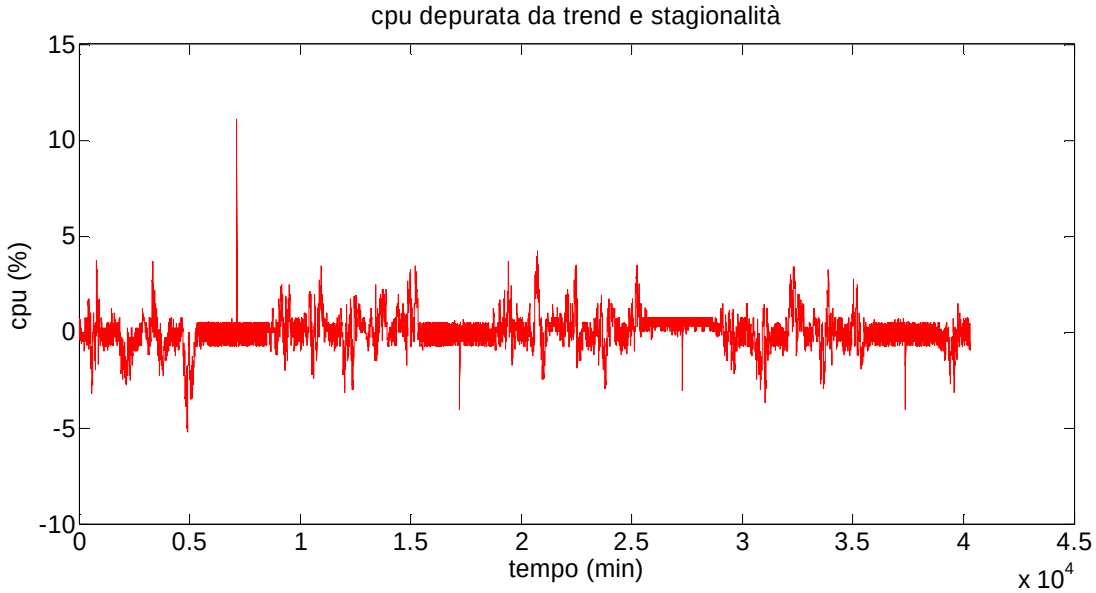


Figura 2.25: andamento della componente stocastica dell'utilizzo di CPU

Si considera per prima cosa il modello che tratta il segnale nelle sue componenti. Le relazioni tra le diverse componenti sono:

$$\delta y = K_c \delta u$$

$$s_y = K_s s_u$$

$$\bar{y} = K_t \bar{u}$$

Come visto in precedenza per il caso di Adderley si vogliono identificare i coefficienti K_i ottimali attraverso il metodo dei minimi quadrati.

Svolgendo i passaggi già mostrati precedentemente si ottiene:

$$K_c^o = \frac{\sum_{i=1}^{N} \delta y_i \delta u_i}{\sum_{i=1}^{N} \delta u_i{}^2} = 1.7354[\%/(\mathrm{Mb/s})]$$

$$K_s^o = \frac{\sum_{i=1}^{N} s_{y,i} s_{u,i}}{\sum_{i=1}^{N} s_{u,i}{}^2} = 2.0608[\%/(\mathrm{Mb/s})]$$

$$K_t^o = \frac{\sum_{i=1}^{N} \bar{y}_i \bar{u}_i}{\sum_{i=1}^{N} \bar{u}_i{}^2} = 3.7380[\%/(\mathrm{Mb/s})]$$

Ricavati i tre coefficienti K_i^o è possibile effettuare la validazione sui dati dell'ultima settimana; in figura 2.26 e figura 2.27 si hanno i confronti tra i dati e i valori ottenuti dal modello:

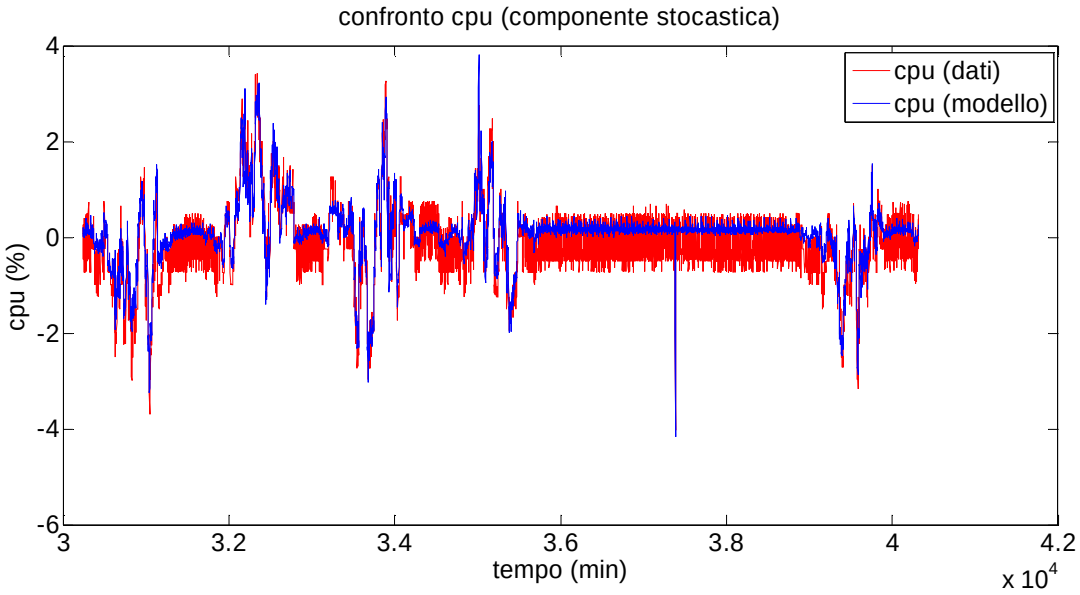


Figura 2.26: confronto tra la componente stocastica ricavata dai dati e l'uscita del modello

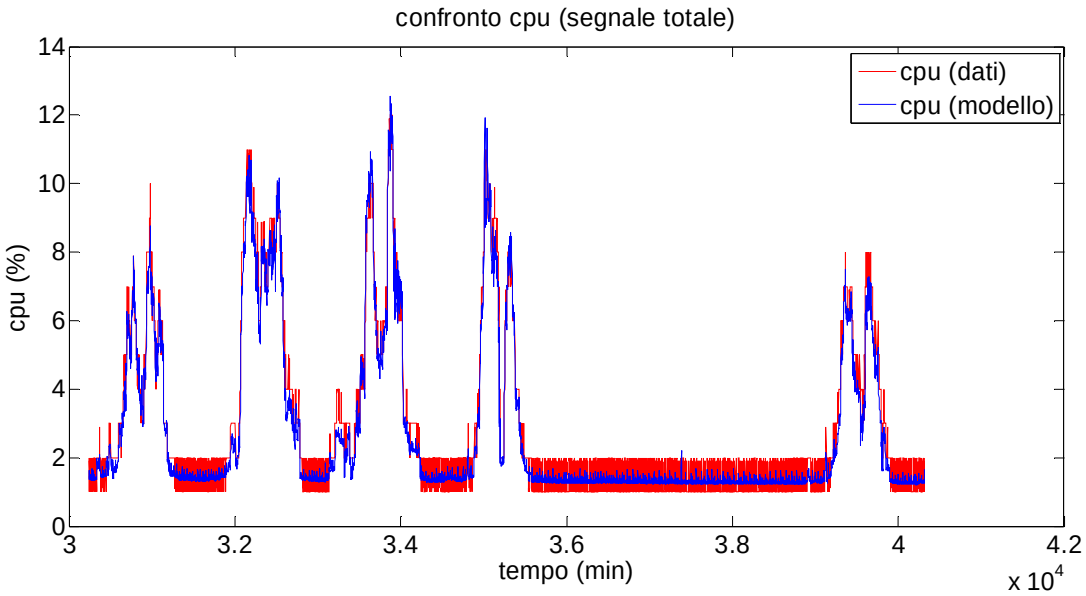


Figura 2.27: confronto tra l'uscita del modello e il set di dati considerando il segnale totale

L'errore percentuale del modello, considerando il segnale totale, risulta essere:

$$err = \frac{1}{N}\frac{\sum_{i=1}^{N}(y-\hat{y})^2}{\mathrm{var}(y)} \cdot 100 = 3.59\%$$

Che corrisponde ad un fitting pari al 96.41%

Dall'ingrandimento della settimana di validazione in figura 2.28 è possibile vedere come effettivamente il modello segua in maniera soddisfacente l'andamento dei dati.

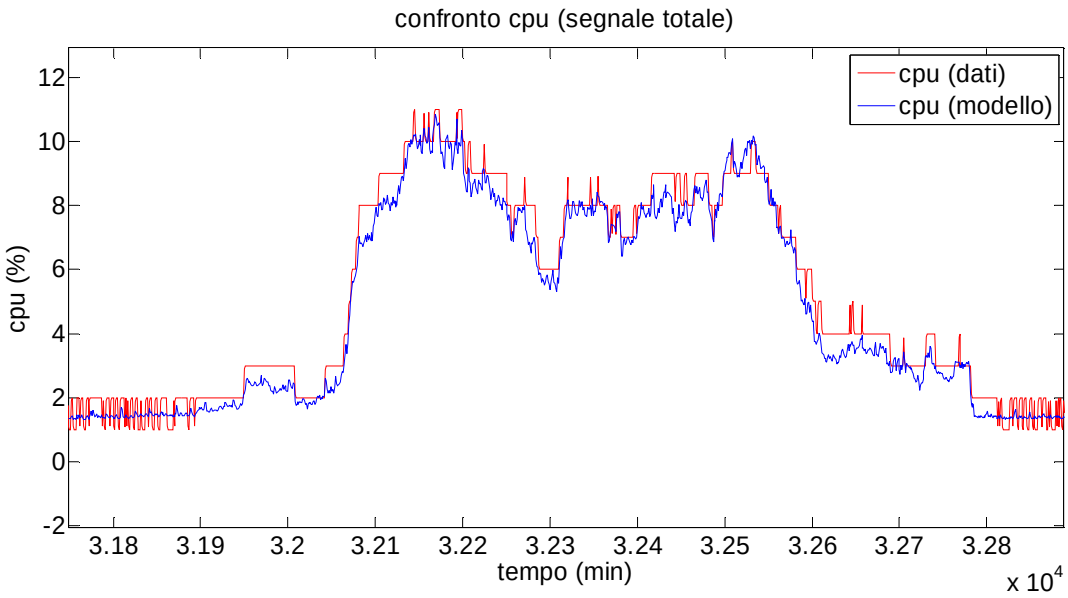


Figura 2.28: particolare del confronto tra dati e uscita del modello (segnale totale)

Terminato il modello a 3 coefficienti, ricaviamo ora il coefficiente K^O che caratterizzi la relazione tra traffico e CPU totali. Svolgendo i passaggi illustrati nel paragrafo 1.4, si ottiene:

$$K^O = \sum_{i=1}^{N} y_i u_i \Big/ \sum_{i=1}^{N} {u_i}^2 = 2.5520\ [\%/(\mathrm{Mb/s})]$$

Che porta ad avere, considerando la settimana di validazione, un fitting pari a:

$$fitting = \left(1 - \frac{1}{N}\frac{\sum_{i=1}^{N}(y-\hat{y})^2}{\mathrm{var}(y)}\right) \cdot 100 = 88.72\%$$

In questo caso si può vedere come un modello con 1 coefficiente dedicato ad ogni componente sia in grado di approssimare in maniera sensibilmente migliore i dati rispetto ad un modello con un unico coefficiente: mentre l'errore nel primo caso risulta inferiore al 4%, nel secondo risulta essere più del doppio, essendo circa il 10%. Ottenere una percentuale di fitting pari al 90% è un risultato comunque soddisfacente, ma, se confrontato con il fitting del primo modello, ci si rende conto come valga la pena di utilizzare 2 coefficienti in più, essendo minimo lo sforzo computazione aggiuntivo che un eventuale applicazione run-time richiederebbe (dal momento che i coefficienti ottimali si ricavano con una semplice operazione di moltiplicazione e divisione) . Per dare prova

dell'ultima affermazione, a fine capitolo vengono mostrati i risultati per ulteriori i router della rete considerata.

2.6 **Generalizzazione nel caso di router che presentano comportamenti anomali**

All'interno della rete Acme, in uno dei router considerati, chiamato Garbarek e avente 5 interfacce, i dati a disposizione non si comportano secondo un andamento stagionale ben delineato, ma presentano degli aumenti di traffico e quindi di occupazione della CPU nelle ore notturne o nei weekend (figure 2.29 e 2.30).

Si è potuto notare come questo comportamento venga identificato da una differenza tra il traffico in ingresso ed il traffico in uscita: il traffico totale in ingresso al router risulta più elevato del traffico totale in uscita proprio in corrispondenza delle anomalie (si veda figura 2.31). Questo indica che non tutti i byte in ingresso al router vengono inoltrati sulle interfacce d'uscita; questo tipo di situazione si può verificare nel caso in cui le informazioni contenute nei pacchetti non ritrasmessi abbiano come destinazione il router in cui si verifica tale comportamento. Poiché è stato verificato che le quantità di traffico in ingresso senza corrispondenza nel traffico d'uscita sono contenute nei pacchetti di livello IP, queste anomalie nei dati possono essere ricondotte ad un periodo di ricalcolo delle tabelle di routing da parte del router interessato.

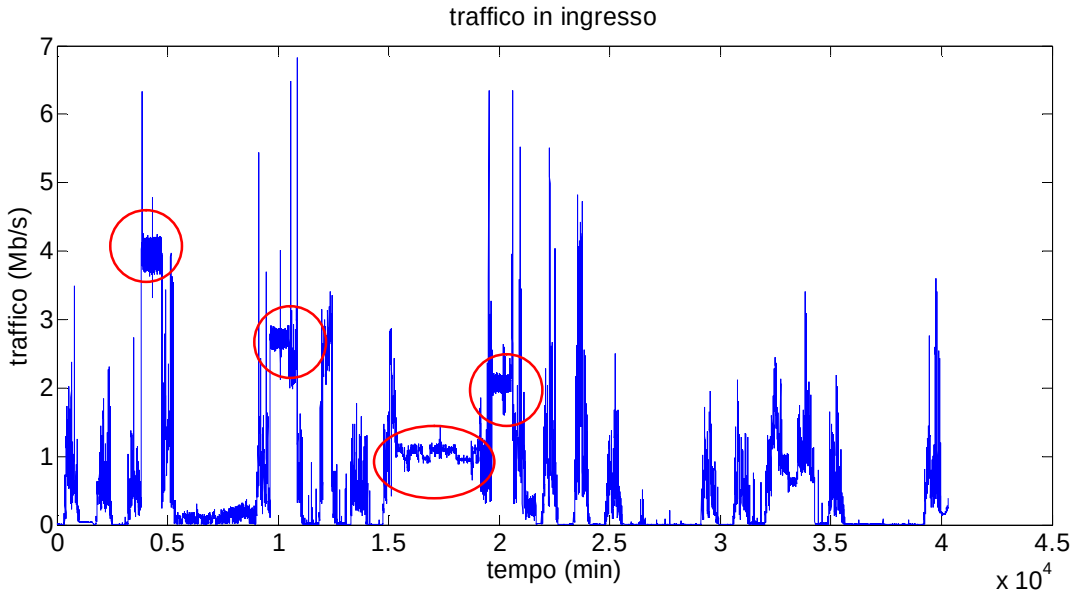


Figura 2.29: individuazione delle anomalie nel segnale di traffico

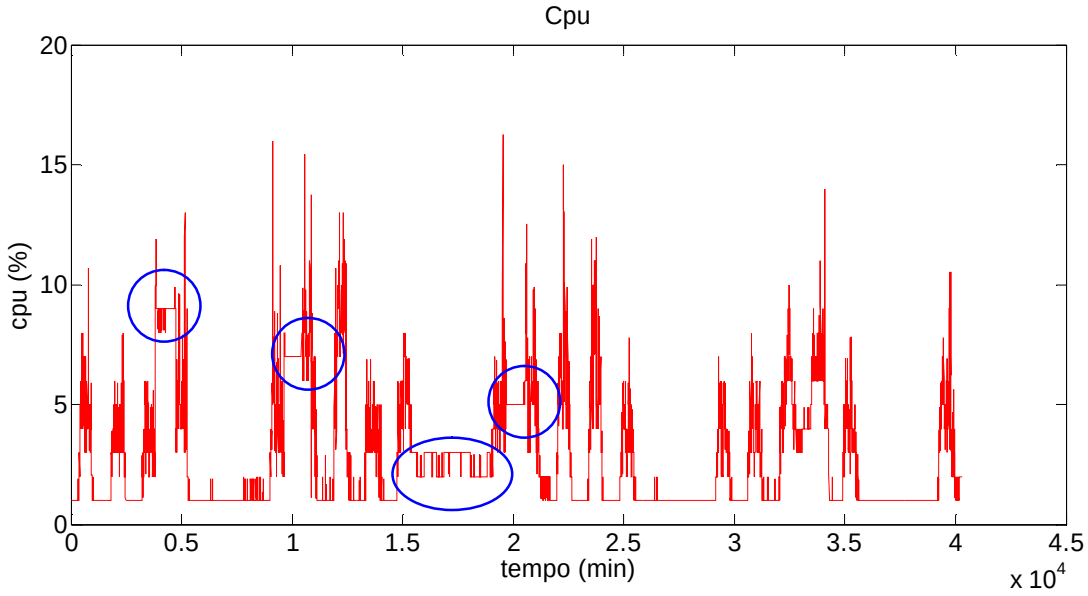


Figura 2.30: individuazione delle anomalie nel segnale di CPU, causate da livelli insolitamente alti del traffico nelle ore notturne

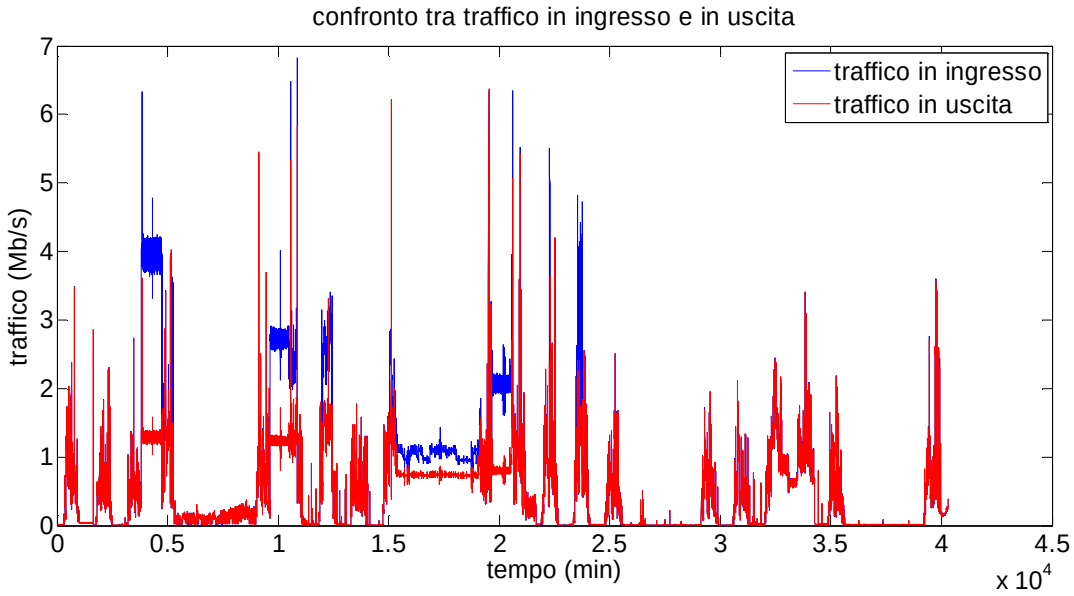


Figura 2.31: metodo di riconoscimento delle anomalie: una differenza tra traffico in ingresso e traffico in uscita diversa da zero indica la presenza di un andamento anomalo

In questo caso si sono adottate due scelte ad hoc:

-per prima cosa si è scelto di identificare, per la componente stocastica, due valori di K_c^o, uno per i dati con comportamento standard ed uno per i dati con anomalie; per le componenti deterministiche l'approccio rimane invariato. Per fare questo è necessario separare i valori che fanno parte del primo insieme da quelli del secondo. Se la differenza tra traffico in ingresso e traffico in uscita è maggiore di un certo valore stabilito allora ci troviamo all'interno del caso anomalo e per i valori negli istanti di tempo in cui vale questa condizione viene identificato K_0^o, mentre per i valori negli altri istanti di tempo viene identificato K_1^o.

-come secondo accorgimento si è scelto di ricavare i coefficienti K_i^O a partire non dal traffico in ingresso ma da quello in uscita. Ciò potrebbe sembrare illogico, in quanto l'utilizzo di CPU è una conseguenza del traffico in entrata e non di quello uscente. Dal punto di vista pratico, in un router in cui tutto il traffico in ingresso è pari a quello in uscita (come tutti gli altri da noi considerati) ciò non porta a differenze sostanziali. In questo caso particolare, invece, utilizzare il traffico uscente porta a non avere le anomalie evidenziate in figura 2.31; in modo da avere un corrispondente segnale più regolare.

Procediamo ora a ricavare i coefficienti, ricavate sotto le assunzioni appena descritte.

I valori dei coefficienti ricavati attraverso l'utilizzo dei minimi quadrati sono:

$$K_0^o = 3.489[\%/(\text{Mb/s})]$$

$$K_1^o = 4.010[\%/(\text{Mb/s})]$$

L'errore sul segnale stocastico (figura 2.32) calcolato sulla settimana di validazione è pari a 11.65%, corrispondente ad un fitting del 88.35%

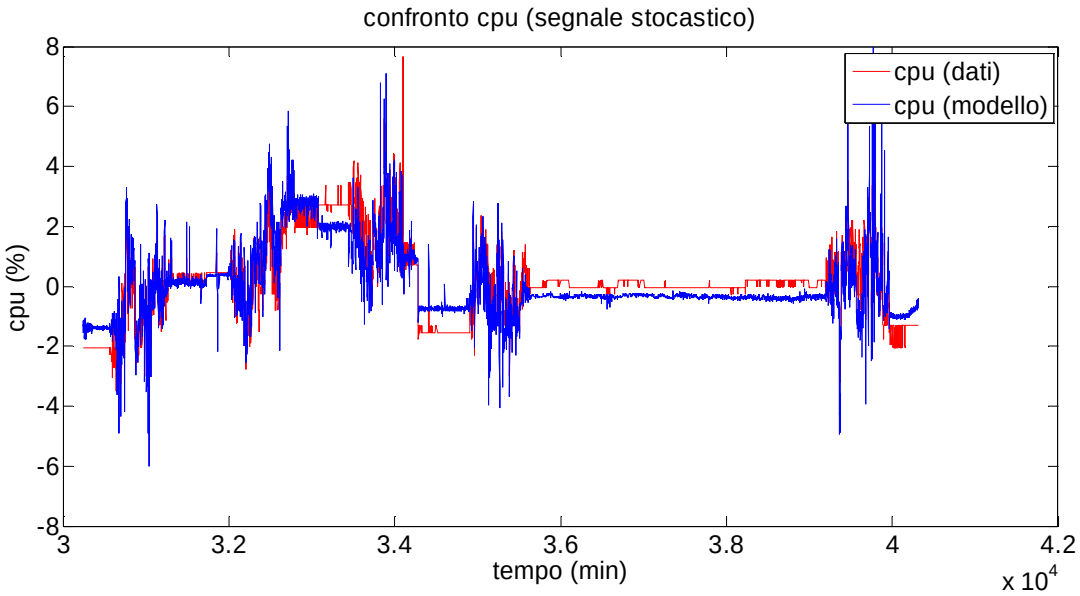


Figura 2.32: confronto tra l'uscita del modello e il set di dati considerando la componente stocastica del segnale

L'errore sul segnale totale (figura 2.33) per la settimana di validazione, calcolati i coefficienti per trend e stagionalità, è pari a 13.85% corrispondente ad un fitting del 86.15%. Da sottolineare il fatto che, nonostante l'utilizzo del traffico d'uscita precedentemente adottato, non si abbia una stagionalità marcata comune a tutte le 4 settimane considerate; ciò porta ad ricavare un segnale stagionale che descrive in modo approssimativo la corrispondente componente reale, portando a performance più basse che negli altri router considerati.

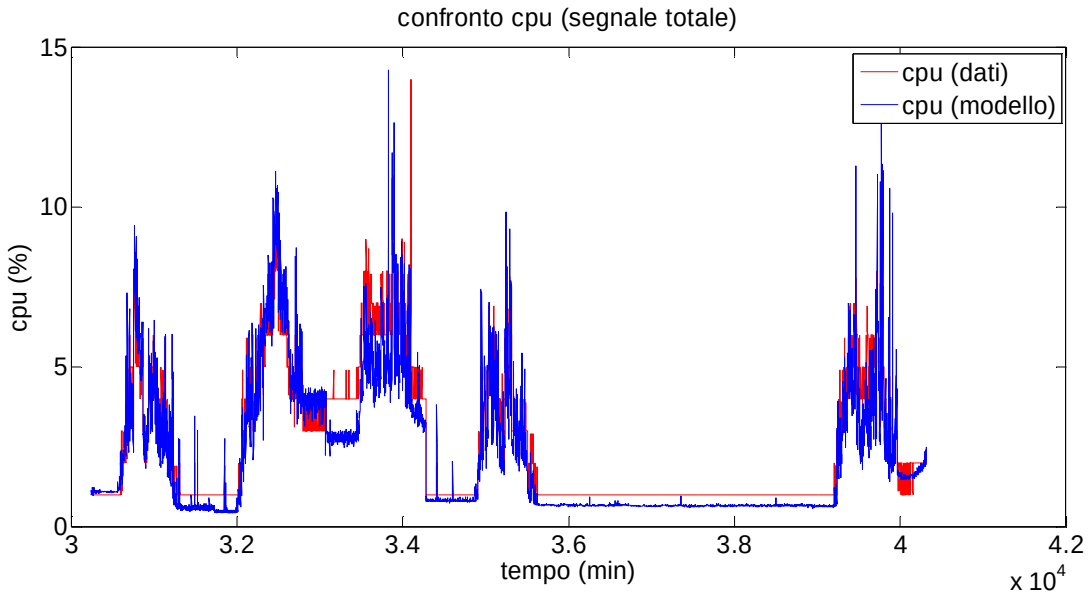


Figura 2.33: confronto tra l'uscita del modello e il set di dati considerando il segnale totale

Per questo modello non viene calcolato un coefficiente K^O che metta in relazioni i segnali totali, in quanto non sarebbe in grado di descrivere le anomalie riscontrate. Vantaggio del modello a tre coefficienti, oltre ad avere prestazioni migliori, è quindi una flessibilità maggiore, dal momento che permette di modificare l'approccio per una delle tre componenti considerate lasciando invariate l'approccio per le altre. L'obiettivo di questo paragrafo non è infatti quello di fornire una modifica che abbia validità generale per tutti i possibili router che presentino anomalie, ma di mostrare come, con pochi accorgimenti, si possa andare a modificare la relazione ricavata per ottenere risultati soddisfacenti.

2.7 **Risultati finali**

Per ottenere ulteriori conferme della bontà di questi risultati il modello è stato identificato per altri router della stessa rete, i risultati (relativi al caso 1 in cui si considerano 3 settimane per l'identificazione) sono riportati in tabella:

	Num. interfacce	K_c^O (componente stocastica) [%/Mb/s]	K_s^O (stagionale) [%/Mb/s]	K_t^O (trend) [%/Mb/s]	K^O(segnale totale) [%/Mb/s]
Adderley	2	0.571	0.866	1.076	0.939
Frisell	2	1.199	1.657	1.642	1.607
Hancock	3	7.727	12.318	15.023	13.01
Garbarek	5	K1: 3.489 K2: 4.010	4.772	6.195	-
Monk	6	1.700	2.066	3.802	2.5520

	Fitting (componente stocastica)	Fitting segnale totale (con 3 coeff)	Fitting segnale totale (con 1 coeff)	Fitting segnale totale (a partire dai dati)
Adderley	88.35%	92.91%	87.49%	98.75%
Frisell	84.54%	95.07%	94.13%	95.65%
Hancock	81.48%	94.9%	91.5%	96.56%
Garbarek	74.79%	85.95%	-	90.34%
Monk	75.51%	96.27%	88.72%	97.95%

Per i casi 2 e 3 si veda l'appendice 2.3.

Si può verificare come, aumentando le interfacce del router considerato, si abbia un valore di fitting inferiore considerando l'utilizzo di CPU (componente stocastica). Inoltre i router con stagionalità meno stabile nel tempo presentano percentuali di fitting minori dovute al fatto che tale componente è prevalente sulle altre. La prestazione del modello risulta essere comunque soddisfacente portando alla conclusione che la relazione statica e lineare sia valida indipendentemente dal numero di interfacce considerate.

2.8 **Condizioni di validità del modello**

Come ultimo argomento del capitolo si presentano le condizioni di validità del modello.

Il modello risulta valido in condizioni standard di utilizzo del router, cioè quando la CPU presenta percentuali di utilizzo inferiori al 45-50%. Oltre tale soglia si ha una congestione in cui un router tipicamente presenta rallentamenti nello smistamento del traffico, creando perciò una dinamica che presenta tempi non trascurabili. Avendo identificato un modello statico, tali dinamiche non possono essere descritte. Non è stato possibile identificare un modello valido per una condizione di sovraccarico di router in quanto tutti i dati a nostra disposizione presentano utilizzi di CPU bassi, compresi tra 0 e 20% circa.

Per percentuali di utilizzo sotto il 2% prolungate nel tempo il comportamento del modello differisce in maniera sostanziale rispetto a quello dei dati. Questo può essere spiegato dal modo con cui viene misurata la percentuale di CPU utilizzata: per valori di traffico molto bassi, l'andamento del valore di utilizzo di CPU viene nascosto dalla risoluzione con la quale viene misurata la percentuale di CPU. In altre parole la quantità di CPU misurata di volta in volta è quantizzata, non assumendo mai valori frazionari, ma solo interi (come si può vedere dai grafici mostrati nel capitolo). Quando il traffico oscilla in una fascia di valori molto bassi, la variazione di

pochi kilobytes può far sì che il sensore apposito misuri un punto percentuale di CPU in più, quando in realtà la quantità realmente usata è inferiore. Per questo una relazione statica tra traffico e CPU non è più da considerarsi valida, in quanto l'andamento dell'utilizzo di CPU è "nascosto" dalla risoluzione del sensore. Da questo il lower bound del modello. Tale fenomeno è trascurabile invece quando si hanno valori di CPU più elevati, in quanto l'incremento dovuto all'aggiunta di un singolo blocco è trascurabile rispetto alla quantità totale. Si può quindi concludere che il modello approssima con risultati soddisfacenti la realtà quando l'utilizzo di CPU è compreso in una fascia tra il 2-3% e il 45-50%.

Tali condizioni non inficiano l'utilità del modello se si considera quali sono le percentuali di utilizzo di un router all'interno di una rete aziendale: prendendo in esame le rete Acme, non si hanno router con valori di CPU al di sopra del 20% circa. Ciò vuol dire che in nessuno caso l'upper bound impedisce l'utilizzo del modello. Per quanto riguarda attività come il capacity plan (ossia la previsione di quali parti della rete richiederanno potenziamenti e il conseguente intervento futuro), l'obiettivo di quest'ultimo è prevenire il fatto che le prestazioni di una rete raggiungano livelli non soddisfacenti. Per far ciò, i metodi impiegati fino ad ora consistono nel prevedere l'andamento del traffico e in base a questo trarre conclusioni sul possibile andamento futuro di CPU. Si capisce quindi come sia necessario non tanto un modello che descriva il comportamento di un router congestionato, quanto un modello che descriva il comportamento del router quando ancora non presenta rallentamenti e che sia in grado di predire l'eventuale violazione di una soglia di allarme di occupazione di CPU. Tale violazione rappresenta un campanello di allarme che dovrebbe essere seguito da un intervento manuale avente come obiettivo il ridurre il carico del router in esame e, conseguentemente, far ritornare i valori di CPU in un range in cui il modello sia valido.

Per lo stesso motivo il lower bound ha un impatto trascurabile per gli obiettivi che il capacity plan si propone: se l'allocazione di CPU rimane costantemente sotto il 2%, il router non corre rischio di entrare in zona di saturazione. Tale evento potrebbe verificarsi solo attraverso l'incremento dei valori di CPU allocata, portando il segnale entro il range di validità del modello; perciò un eventuale saturazione futura di un router che presenta per un certo periodo scarso utilizzo verrebbe comunque notata dal modello.

CONCLUSIONI

In questo capitolo si è studiato la relazione che intercorre tra il traffico in ingresso ad un router e la corrispondente allocazione di CPU. Si è potuto concludere che un modello statico nella forma $y = Ku$ approssima in modo soddisfacente tale relazione. In particolare, sono stati consideranti 2 diversi approcci: un primo, in cui per ogni componente del segnale si ricava il corrispondente coefficiente che metta in relazione traffico e CPU, ed un secondo, in cui si considerano i segnali complessivi. Entrambi i metodi portano alla conclusione che il modello statico approssima con buoni risultati la realtà; in particolare il primo modello, utilizzando un numero maggiore di gradi di libertà, è in grado di descrivere meglio tale relazione.

Si è poi generalizzato tale risultato ai router aventi più di due interfacce, giungendo alla conclusione che il modello sviluppato è adeguato, pur con prestazioni leggermente inferiori, anche per router a più interfacce.

Si è considerato un caso anomalo, in cui i segnali di traffico in ingresso ed uscita mostravano disuguaglianze, dovuto al fatto che il traffico ricevuto non veniva re-indirizzato, concludendo che esso era formato da informazioni destinate al router stesso, caso anomalo se presente in quantità come quelle evidenziate. Modificando leggermente il primo modello considerato, si è mostrato come questo modello possa descrivere con buona approssimazione casi anomali.

Sono state mostratele condizioni di validità del modello, evidenziando la presenza di bound superiori e inferiori; il primo causato dal rallentamento del router a causa di utilizzi di CPU considerate alte (sopra il 45-50%), il secondo in corrispondenza di valori di CPU molto bassi, in cui la dinamica del segnale si perde nella risoluzione con cui esso viene misurato; si è evidenziato come tali limiti non compromettano la validità del modello.

Una possibile applicazione dei risultati ottenuti consiste nella previsione dell'occupazione di CPU: mentre in letteratura si sono potuti riscontrare studi sulla previsione dell'andamento futuro del traffico, non sono state individuate trattazioni sulla previsione dell'utilizzo di CPU. Grazie alla relazione statica lineare identificata si può riformulare tale problema non più come la ricerca di un modello in grado di prevedere l'occupazione futura di CPU, ma in un problema suddiviso in due step: un primo in cui si ricava un predittore per il traffico, seguito da un secondo in cui attraverso le relazione identificata ci si riconduce alla CPU; ciò risulta di particolare interesse in diverse applicazioni.

Nel corso dei prossimi capitoli il modello ricavato verrà quindi utilizzato per poter fare, a partire dal traffico, previsione a breve e lungo termine (rispettivamente capitolo 3 e 4). In altre parole, a partire dalle considerazioni fatte a conclusione dello stato dell'arte, si affronterà in maniera rigorosa il problema di come identificare un modello per il traffico in grado di descriverne con buone prestazioni le dinamiche, per poi ricavare un predittore per il traffico e infine ricondursi, attraverso i risultati di questo capitolo, ad una predizione per l'allocazione di CPU.

3

Predizione short-range di traffico e utilizzo di CPU di un router

Nel capitolo precedente si è modellizzata la relazione esistente tra traffico in ingresso ad un router e conseguente utilizzo di CPU. L'obiettivo di questo capitolo è invece, usando il modello ricavato, la predizione della CPU per un orizzonte temporale breve (short-range).

Una possibile applicazione per i risultati di questo capitolo è il "load balancing" delle risorse all'interno di una server farm [18]: una server farm consiste in un insieme di server, non visibili dalla rete pubblica, che condividono gli stessi contenuti e su cui sono installate le stesse applicazioni. Dalla rete pubblica un client non si connette al singolo server, non essendo visibile, ma all'applicazione, per esempio per richiedere contenuti, salvare informazioni o rielaborare dati. Se si immagina quindi di avere un insieme di server che hanno come compito il soddisfare richieste di informazioni provenienti dall'esterno attraverso la rete internet, l'obiettivo è il dare risposta a tutte le domande nel minor tempo possibile.

Per far ciò è importante che le richieste siano equamente ripartite tra tutti i server disponibili. In un caso ideale in cui tutti i server sono identici basterebbe assegnare a tutti i server lo stesso carico di lavoro; in realtà difficilmente le macchine presentano le stesse caratteristiche. Per poter garantire un servizio efficiente si ha quindi bisogno di ripartire il carico in modo che l'utilizzo delle risorse di ciascuna macchina sia bilanciato, dando un numero maggiore di richieste ai server più liberi in un dato momento. Si può quindi capire come l'importanza della predizione entro un lasso temporale di qualche minuto acquisti peso: ciò che conta in questo caso non è il trend caratterizzante l'occupazione di CPU, quanto il suo valore puntale; potrebbe essere ad esempio vantaggioso attribuire momentaneamente maggior carico di lavoro ad un server che mostra un trend in salita, ma che momentaneamente presenta bassa utilizzazione, piuttosto che ad un secondo server con una media di utilizzo bassa ma che in un dato momento presenta un picco di lavoro dovuto ad una particolare richiesta.

Un secondo ambito in cui una predizione a breve termine può risultare utile è il rilevamento di attacchi telematici alla rete presa in esame: un possibile metodo con cui una rete può essere attaccata consiste nel richiedere una quantità molto elevata di informazioni (nel caso di una server farm) o nel "bombardamento" del device attraverso un numero di pacchetti sufficientemente alto da portarlo ad un sovraccarico e a farlo congestionare (nel caso di un router). Dal momento che in genere tali attacchi generano un aumento del traffico e di conseguenza della CPU con trend di

salita molto elevati, avere a disposizione la previsione dell'andamento per 10-15 minuti permette di avere un riferimento con cui confrontare l'andamento reale del segnale: in caso di una differenza marcata tra predizione e dati reali, in cui quest'ultimi presentano trend di salita molto superiori a ciò che si era predetto, si può concludere che sia in atto un attacco alla rete, allertando quindi in anticipo gli operatori e permettendo di agire tempestivamente per cercare di riportare i valori di traffico e CPU a livelli accettabili.

Come nel capitolo precedente i dati utilizzati provengono dalla rete Acme. Sono disponibili dati per un totale di 4 settimane di traffico e CPU, di cui le prime 3 vengono usate per l'identificazione del modello, l'ultima sia per la validazione che per la predizione. Il tempo di campionamento con cui i dati sono stati raccolti è pari a 1 minuto, dal momento che l'obiettivo è prevedere l'andamento di traffico e successivamente CPU nell'immediato futuro.

All'interno del capitolo vengono usati una gamma modelli diversi, con lo scopo di individuare quale, tra le diverse tipologie, sia quella più indicata a modellizzare le dinamiche assunte dal traffico.

I modelli utilizzati sono:

- un modello ARMA: tale modello non viene solitamente utilizzato, in quanto il traffico risulta essere un segnale non stazionario; tuttavia, individuando trend e stagionalità come avvenuto nel capitolo precedente, è possibile ricondursi ad un segnale stazionario e quindi poter impiegare un modello ARMA.
- Un modello ARIMA, utilizzato frequentemente in letteratura per poter prevedere l'andamento del traffico in una rete.
- Il modello di Holt-Winters, dal momento che, rispetto ai due modelli precedentemente considerati, presenta il vantaggio di poter aggiornare la stagionalità ed il trend in modo automatizzato.

Per ognuno di essi si divide la trattazione in 3 parti: la prima, in cui si identifica un modello in grado di caratterizzare adeguatamente l'andamento del traffico; la seconda, che ha come obiettivo la creazione di un predittore per il traffico a partire dal modello individuato; infine la terza in cui, sfruttando la relazione traffico-CPU del capitolo precedente, si ottiene la predizione dell'utilizzo di CPU.

3.1 **Modello arma**

3.1.1 Identificazione del modello

Il primo modello preso in considerazione è un modello ARMA.

Si considera il traffico nella sua sola componente depurata da trend e stagionalità (δu), ricavata come nel precedente capitolo; una volta identificato il modello quest'ultime saranno risommate in modo da ottenere un segnale $\hat{u}$ che verrà confrontato con il segnale totale u. In figura 3.1 è mostrato il traffico depurato (δu):

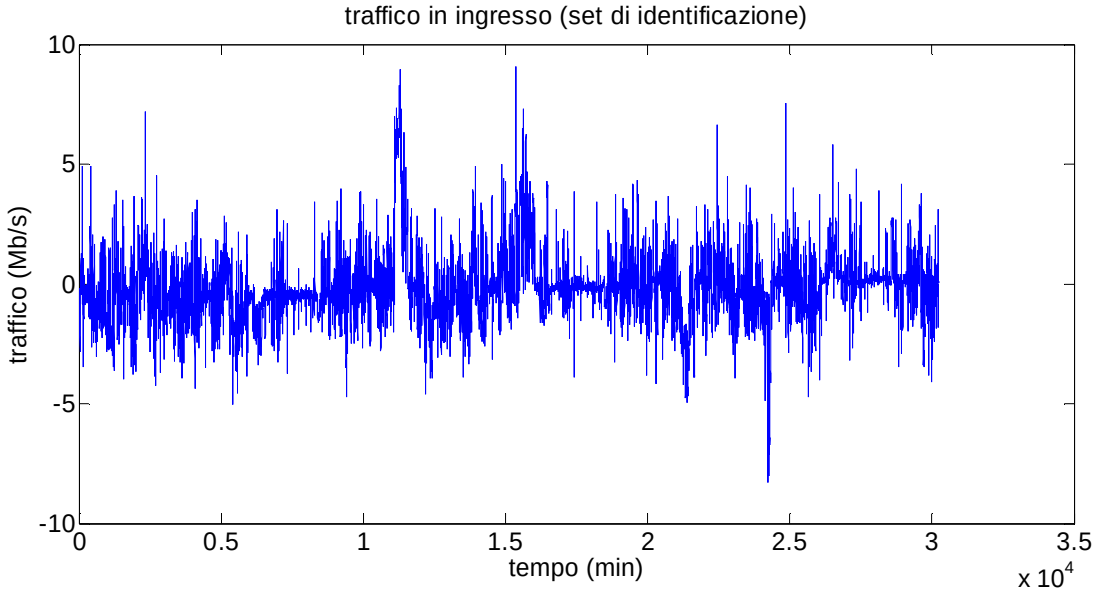


Figura 3.1: andamento assunto dal traffico in ingresso depurato da trend e stagionalità considerando il set di identificazione (3 settimane)

L'equazione caratterizzante un modello arma è la seguente:

$$\varphi(z^{-1})\delta u(t) = \vartheta(z^{-1})e(t)$$

Di cui si devono identificare: p (grado del polinomio φ, parte auto-regressiva del modello), q (grado del polinomio ϑ, parte a media mobile) e i parametri dei due polinomi.

Per prima cosa si determina l'ordine del modello, per far questo si procede nel seguente modo:

-si considerano diverse cifre di merito (Loss function, AIC, FPE (come definiti in [3], [19], [20]) e la cifra di merito definita nel capitolo precedente) che forniscano indicazioni sul fitting del modello di volta in volta considerato.

-si ricavano le mappe di poli e zeri per ogni ordine considerato.

-si studia la funzione di trasferimento di ogni modello, paragonandola a quella ricavata a partire dai dati.

Studio dell'andamento delle cifre di merito

Consideriamo inizialmente un modello AR dal momento che, per qualsiasi cifra di merito si prenda in esame, esiste sempre un ordine del modello per cui si ha un minimo globale; cioè, considerando la funzione cifra di merito al variare dell'ordine del modello, tale funzione è convessa. Ciò fornisce un'indicazione per quanto riguarda l'ordine della parte AR che il modello ARMA dovrà avere.

Le cifre di merito considerate per valutare la bontà del modello al variare dell'ordine sono:

-Loss function: $J = \frac{1}{N}\sum_{i=1}^{N}(\delta u_i - \delta\hat{u}_i)^2$

-Loss function normalizzata sulla varianza: $error = \frac{1}{N}\frac{\sum_{i=1}^{N}(\delta u_i - \delta\hat{u}_i)^2}{\mathrm{Var}(\delta u)}$

-Akaike information criterion: $AIC = 2\frac{n}{N} + \ln(J)$

-Final prediction error: $FPE = \frac{N+n}{N-n} J$

In cui N è pari a 30240 (numero di minuti in tre settimane), *n* è invece l'ordine del modello che di volta in volta si considera.

Nella figura 3.2 è mostrato l'andamento della seconda cifra di merito considerata (in valore percentuale) al variare dell'ordine considerato:

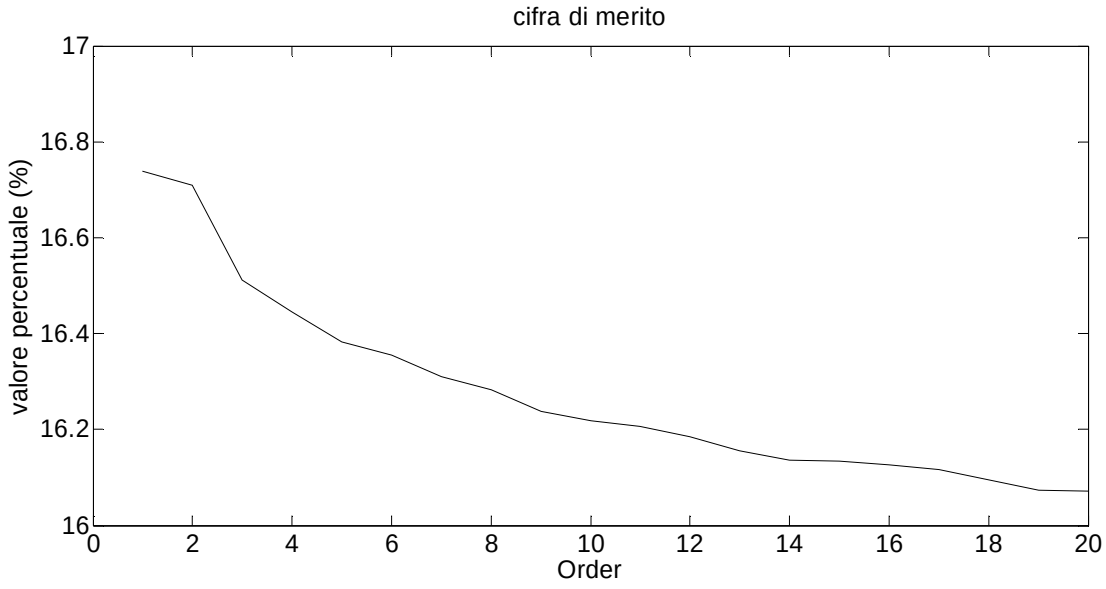


Figura 3.2: andamento della cifra di merito J normalizzata sulla varianza del segnale considerato (δu)

In appendice 3.1 sono mostrati i risultati relativi alle cifre di merito J,AIC e FPE, che presentano andamenti simili a quello mostrato qui sopra.

Si può notare come tutte le cifre di merito non convergano ad un valore minimo per un ordine basso, ma tendano indefinitamente a valori via via più bassi all'aumentare di quest'ultimo, facendo intuire che sia abbia un minimo globale in corrispondenza di valori molto alti del modello. Tuttavia, oltre un certo valore, le cifre di merito presentano diminuzione sempre più lievi, portando alla conclusione che optare per un modello di ordine alto porti ad un aumento della complessità senza un sostanziale miglioramento delle performance.

Ricalcolando ora le cifre di merito considerando un modello ARMA si ottiene:

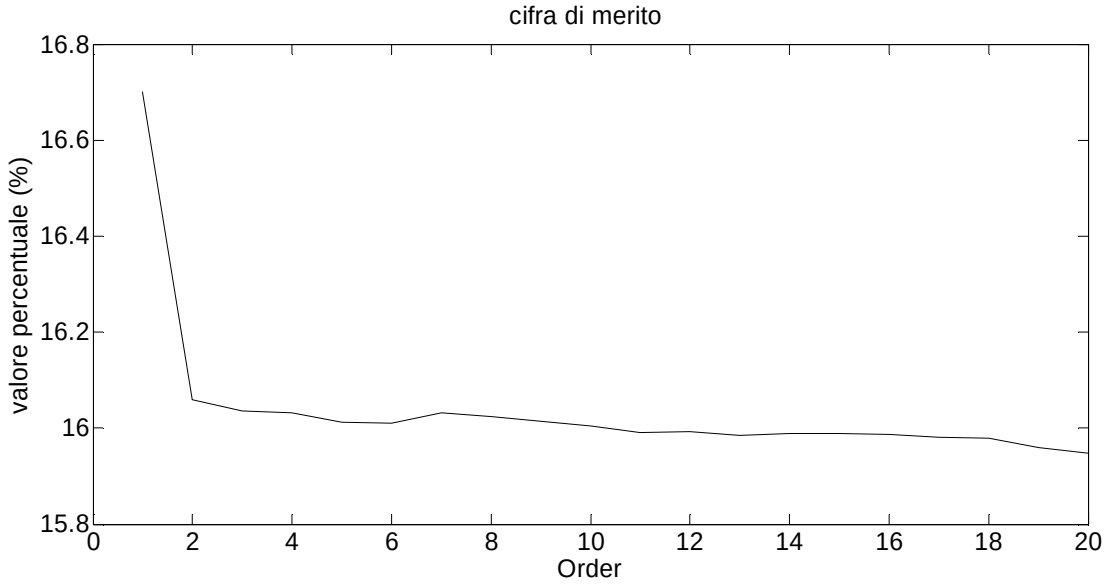


Figura 3.3: andamento della cifra di merito J normalizzata sulla varianza del segnale considerato (δu) considerando un modello ARMA

Di nuovo si rimanda all'appendice 3.1 per i grafici relativi alle cifre di merito J, AIC e FPE.

Anche in questo caso le cifre di merito non convergono ad un minimo, nemmeno locale; tuttavia si può osservare che oltre un ordine pari a 3 la diminuzione sia nuovamente minima.

Valutazione delle mappe di poli e zeri

Una seconda motivazione che porta a propendere per una scelta bassa dell'ordine è la posizione di poli e zeri dei modelli. Da questo momento in poi si prenderanno come esempio 4 diversi ordini; rispettivamente gli ordini 3 e 4 (presi come esempio di ordine basso), 6 e 20 (preso come riferimento per modelli con ordine elevato).

In figura se ne possono osservare le mappe di poli e zeri:

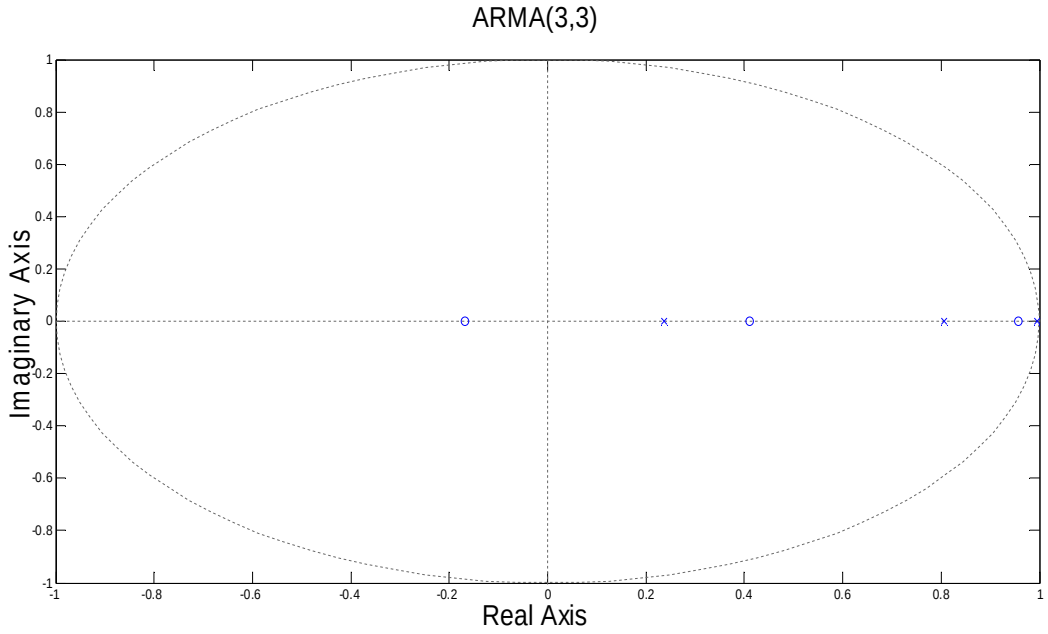


Figura 3.4: Mappa di poli e zeri relativa al modello ARMA (3,3)

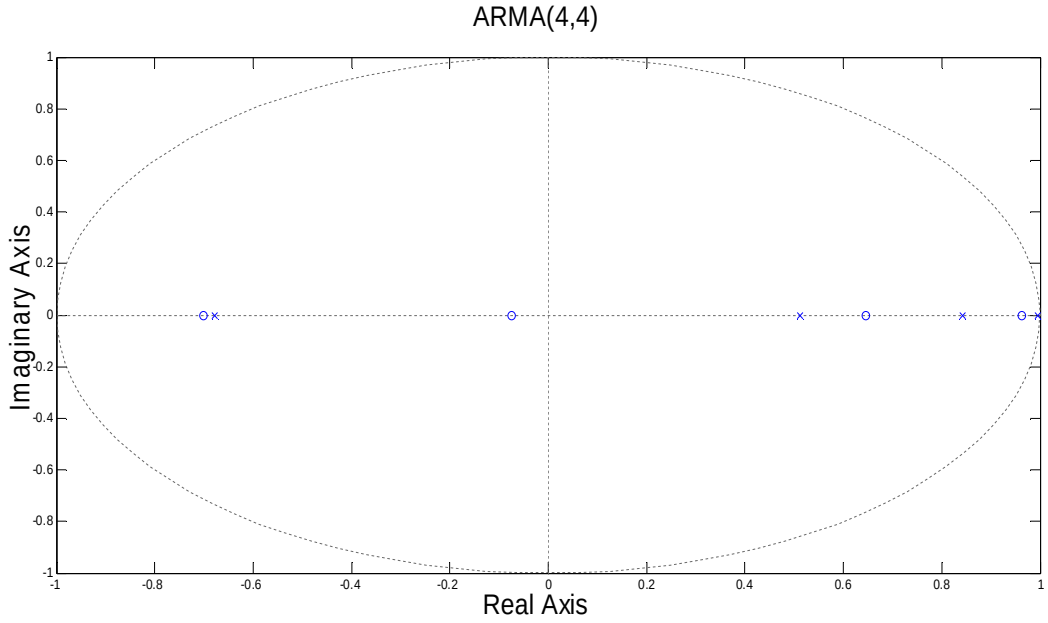


Figura 3.5: Mappa di poli e zeri relativa al modello ARMA (4,4)

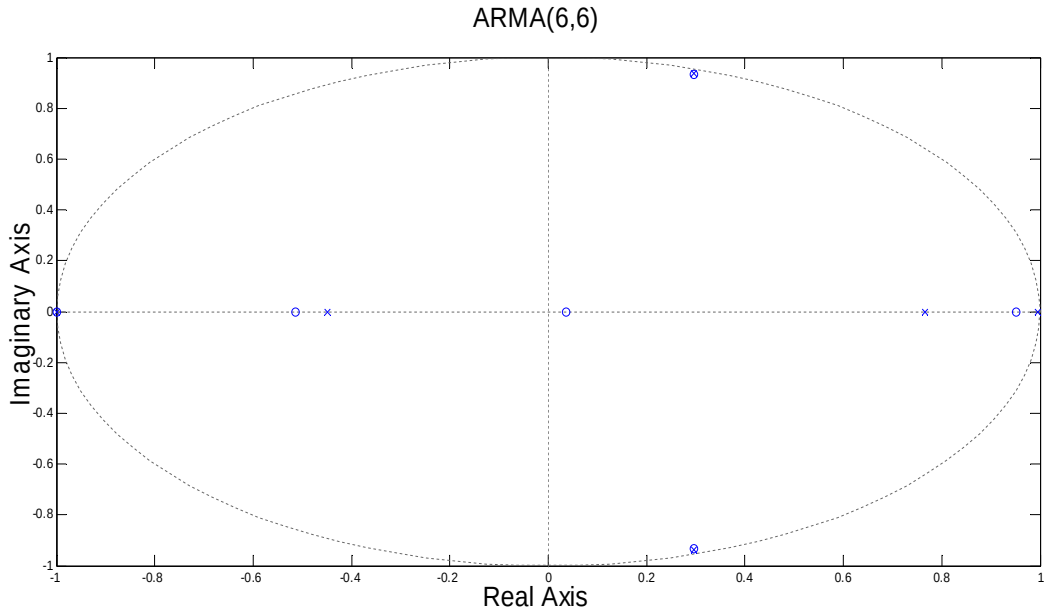


Figura 3.6: Mappa di poli e zeri relativa al modello ARMA (6,6)

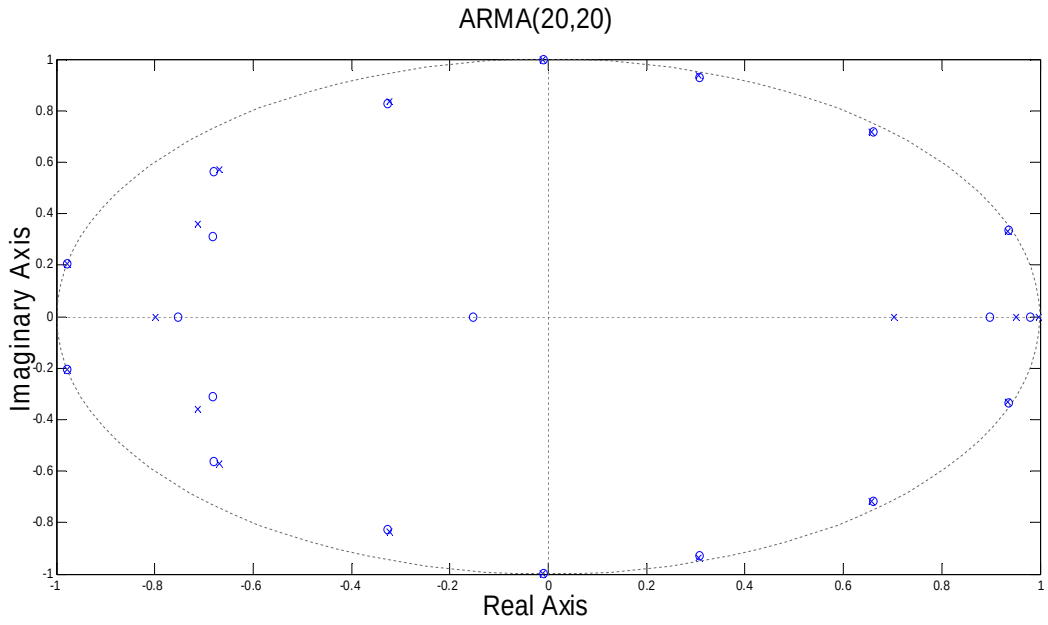


Figura 3.7: Mappa di poli e zeri relativa al modello ARMA (20,20)

Si può osservare come all'aumentare dell'ordine aumentino le cancellazioni; il numero di coppie poli-zeri che si elidono è tale per cui i poli non eliminati risultano pari (a meno che nell'ARMA(20,20)) a 3. Si è verificato che per ordini elevati diversi dal ventesimo i poli non cancellati sono pari a 3. Da questo si può concludere che un ordine elevato non porta ad una modellizzazione migliore l'evoluzione del traffico, ma ad un elevato numero di cancellazioni polo-zero che fanno sì che un modello di ordine superiore si comporti in modo simile ad un modello di ordine 3.

Studio della funzione di trasferimento

Si ricavano ora le funzioni di trasferimento dei modello per gli ordini considerati, e le si confronta con la f.d.t. ricavata dai dati, come ulteriore indice di bontà di modellizzazione: ci aspettiamo che i modelli con funzione di trasferimento maggiormente simile a quella proveniente dai dati abbiano un fitting migliore rispetto agli altri.

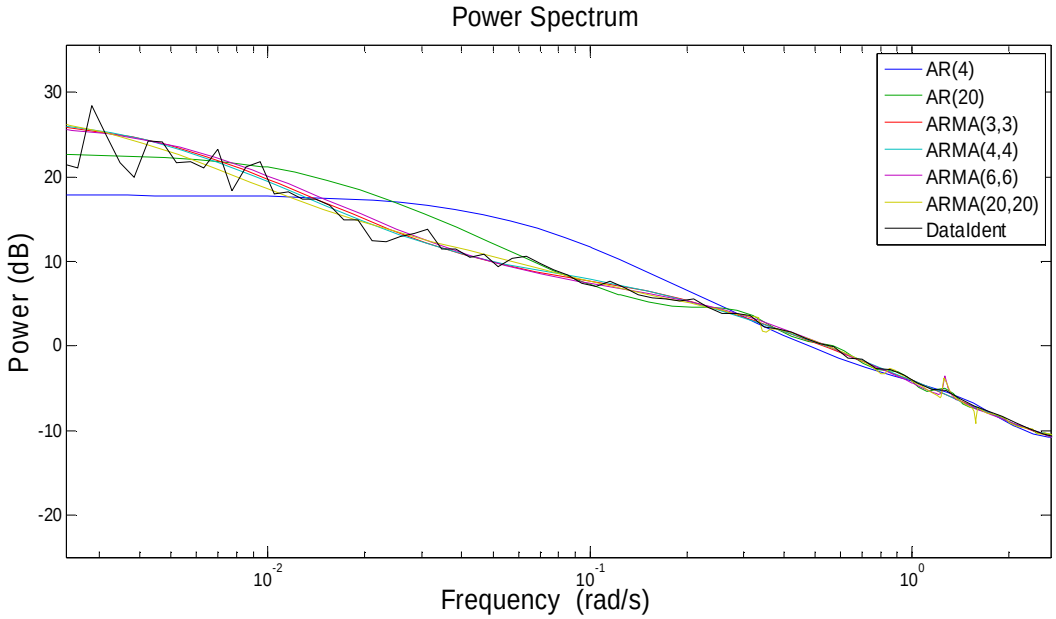


Figura 3.8: confronto tra funzione di trasferimento ottenuta a partire dai dati e le relative ai modelli considerati

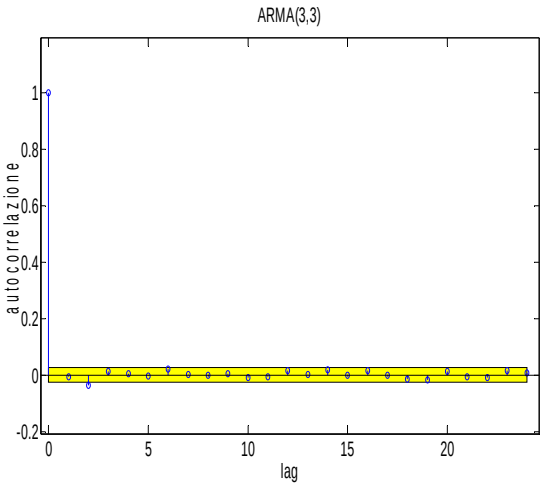
In figura sono stati inserite anche le funzioni di trasferimento relative a due modelli AR, rispettivamente di ordine 4 e 20, per far notare come questa classe di modelli approssimi l'andamento del traffico in maniera peggiore rispetto ai modelli ARMA. Per quanto riguarda quest'ultimi, si può notare come in generale abbiano un andamento pressoché simile, a conferma di quanto concluso dall'osservazione delle mappe di poli e zeri.

3.1.2 Validazione

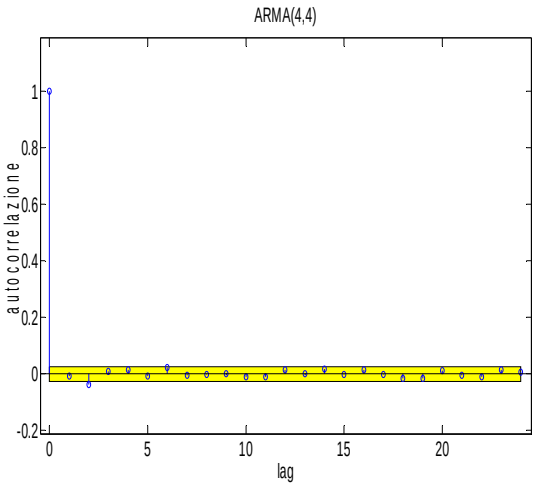
Si passa infine alla validazione dei modelli, in cui si utilizzano i dati relativi all'ultima settimana di traffico a disposizione per determinare quale dei modelli rappresenta al meglio il comportamento del traffico. Per far questo, ricaviamo per ogni modello il predittore ad un passo, (per la trattazione riguardante la parte predittiva si rimanda al paragrafo successivo) e ne confrontiamo l'uscita con i dati reali.

Come verifica del fatto che quelli ricavati siano i predittori ottimi, si propone un'analisi dei residui. Definiamo un residuo come la differenza tra l'uscita del predittore ad un passo e il dato misurato; su di esso si effettua un test di bianchezza: se la correlazione tra due qualsiasi residui relativi ad istanti di tempo diversi è pari a zero allora il test è superato e il predittore ricavato è effettivamente quello ottimale, in caso contrario il test è negativo; ciò significa che è possibile ottenere un predittore migliore.

Definiamo inoltre per ogni test un intervallo di tolleranza entro il quale il residuo è considerato essere circa pari a zero. In figura 3.9 e 3.10 sono mostrati i risultati:

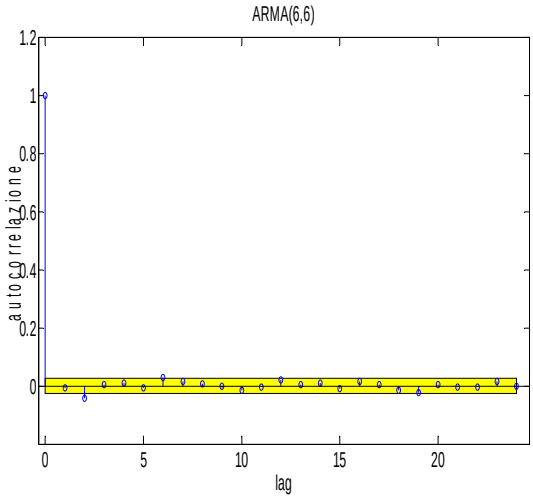


Residui relativi al predittore ricavato dal modello ARMA(3,3)

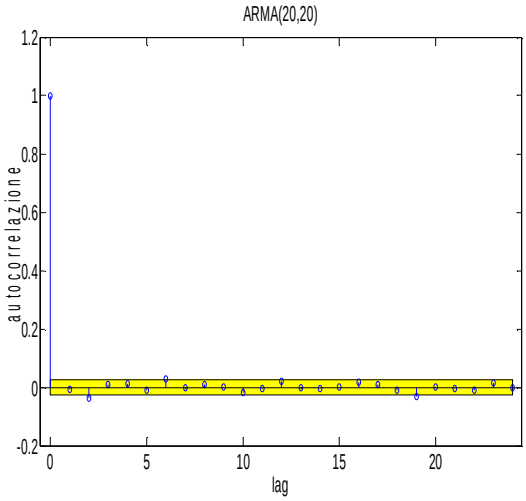


Residui relativi al predittore ricavato dal modello ARMA(4,4)

Figura 3.9



Residui relativi al predittore ricavato dal modello ARMA(6,6)



Residui relativi al predittore ricavato dal modello ARMA(20,20)

Figura 3.10

Si può osservare come, in tutti i predittori considerati, per la quasi totalità dei ritardi (dati dalla differenza dei due istanti temporali considerati di volta in volta) i residui siano all'interno dell'intervallo in cui si considera il residuo circa pari a 0. In base a quanto detto sopra possiamo perciò ritenere i predittori ottimi.

Possiamo quindi passare alla validazione vera e propria: per comodità si considera d'ora in poi solamente la seconda cifra di merito per valutare la percentuale di fitting dei modelli, in cui, ora, il numero di sample N è pari a 10080. In figura 3.11 e 3.12 è mostrato il confronto grafico tra i dati raccolti e il modello ARMA (3,3), mentre in tabella si trovano le percentuali di fitting relativi a tutti gli ordini considerati.

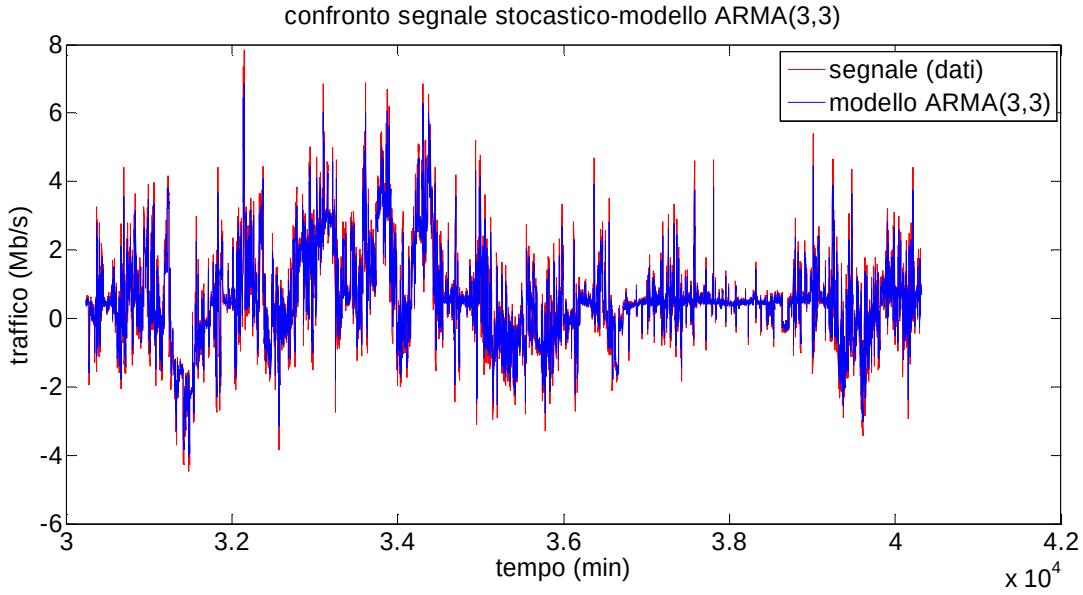


Figura 3.11: confronto tra la componente stocastica di traffico ottenuta a partire dai dati e l'output del modello ARMA(3,3) (settimana di validazione)

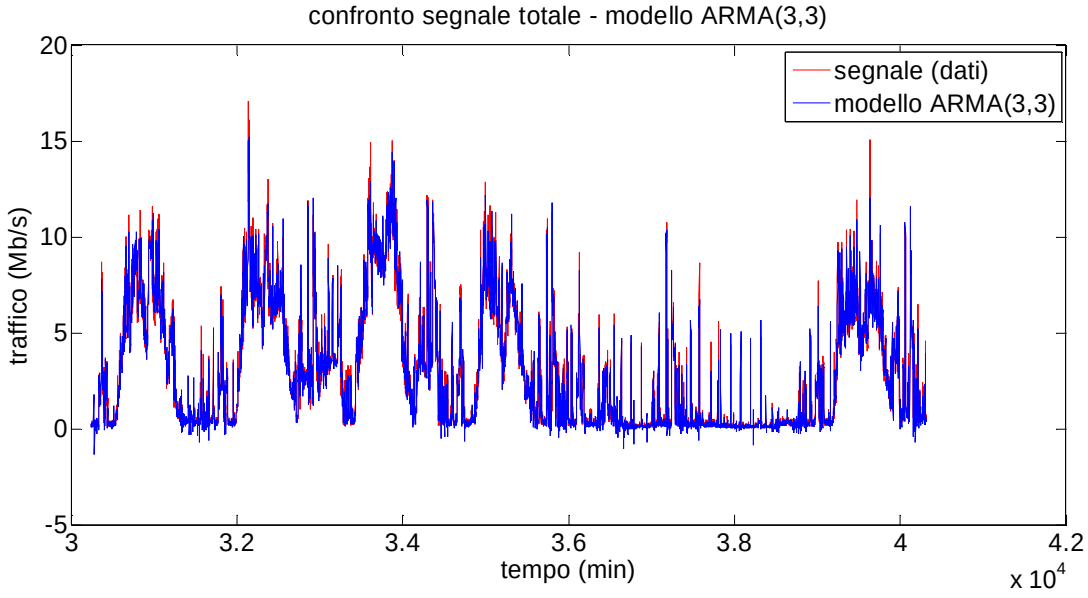


Figura 3.12: confronto tra il segnale totale di traffico e l'output del modello ARMA(3,3) (settimana di validazione)

In tabella sono mostrati i risultati:

Ordine del modello	Fitting su segnale detrendizzato (δu)	Fitting su segnale totale (u)
ARMA(3,3)	84.928%	96.896%
ARMA(4,4)	84.924%	96.895%
ARMA(6,6)	84.892%	96.889%
ARMA(20,20)	84.911%	96.896%

Dai risultati si può concludere che:

-un modello ARMA è in grado di modellizzare in modo più che soddisfacente l'andamento del traffico, come si può concludere dagli alti valori di fitting.

- l'aumento dell'ordine del modello non porta ad un miglioramento sostanziale delle prestazioni; considerando infatti gli ordini 3 e 20, il fitting (considerando la seconda cifra di merito) risulta pressoché identico, con una minima differenza a favore dell'ordine 3 per quanto riguarda il segnale δu. Considerazione analoghe valgono per il segnale totale u.

Per questo motivo, per mantenere basso l'ordine del modello, e ricordando che dal quarto ordine in su aumenta via via il numero di cancellazioni polo-zero, scegliamo di considerare d'ora in poi un modello ARMA (3,3).

I coefficienti dei polinomi φ e ϑ del modello di ordine 3 sono:

$\vartheta_0 = 1$ $\varphi_1 = -2.038$

$\vartheta_1 = -1.202$ $\varphi_2 = 1.229$

$\vartheta_2 = 0.165$ $\varphi_3 = -0.190$

$\vartheta_3 = 0.066$

Risultano in questo modo completamente identificati tutti i parametri del modello.

3.1.3 Predittore del traffico in ingresso ad un router

Identificato il modello ci si focalizza sulla predizione. L'obiettivo di questo capitolo è una predizione di tipo short-range, perciò verranno presentati i risultati di una predizione con un orizzonte temporale limitato. Si sceglie di trattare nello specifico il caso del predittore a 2 passi, generalizzando quanto segue per i predittori a 5 e 10 passi. Ricordando che il traffico viene campionato con sample time pari a 1 minuto, si ottiene una previsione avente un orizzonte temporale di qualche minuto. Vengono inoltre presi in considerazione dei predittori a 50 e 100 passi, che hanno la funzione di mostrare come evolva la predizione di un modello ARMA all'aumentare dell'orizzonte temporale, dando via via risultati meno soddisfacenti.

Ricordando l'equazione di un modello ARMA:

$$\varphi(z^{-1})\delta u(t) = \vartheta(z^{-1})e(t)$$

Che può essere riscritta come:

$$\delta u(t) = -\varphi_1\delta u(t-1) - \varphi_2\delta u(t-2) - \varphi_3\delta u(t-3) + \vartheta_0 e(t) + \vartheta_1 e(t-1) + \vartheta_2 e(t-2) + \vartheta_3 e(t-3)$$

Il predittore ad *h* passi è il seguente [5]:

$$\delta\hat{u}(t+h|t) = \frac{\vartheta_r(z)}{\varphi(z)}\, e(t)$$

In cui $\vartheta_r(z)$ rappresenta il resto della lunga divisione svolta *h* volte. Per cui, una volta fissato l'orizzonte predittivo, è sufficiente iterare la lunga divisione per il corrispondente numero di volte e a partire da $\vartheta_r(z)$ ricavare il predittore.

Un predittore nella forma mostrata non è però utilizzabile, in quanto è presente in ingresso il rumore bianco $e(t)$. Dal momento che si ha bisogno di una relazione che leghi i valori di $\delta u(t)$ a quelli di $\delta\hat{u}(t+h|t)$, c'è bisogno di un ulteriore sviluppo.

Invertendo la formula del predittore ARMA si ottiene:

$$e(t) = \frac{\varphi(z)}{\vartheta(z)}\delta u(t)$$

Andando a sostituire nell'equazione del predittore:

$$\delta\hat{u}(t+h|t) = \frac{\vartheta_r(z)}{\varphi(z)}\, e(t) = \frac{\vartheta_r(z)}{\varphi(z)}\frac{\varphi(z)}{\vartheta(z)}\delta u(t) = \frac{\vartheta_r(z)}{\vartheta(z)}\delta u(t)$$

Va ricordato che l'operazione fatta per ricavare il rumore bianco a partire dall'uscita del modello è lecita solo nel caso in cui i gradi dei polinomi φ e ϑ siano uguali, come nel caso trattato.

Considerando un predittore a 2 passi, la lunga divisione va iterata due volte, per cui il polinomio $\vartheta_r(z)$ sarà pari a :

$$\vartheta_r(z) = 0.639z^{-2} - 0.771z^{-3} + 0.159z^{-4}$$

Quindi il predittore a 2 passi sarà:

$$\hat{u}(t+2|t) = \frac{0.639z^{-2} - 0.771z^{-3} + 0.159z^{-4}}{1 - 1.202z^{-1} + 0.165z^{-2} + 0.066z^{-3}}\, u(t)$$

Per quanto riguarda i predittori a 5 e 10 passi, se ne fornisce direttamente l'equazione finale, in quanto ottenuti dal resto della prima divisione tra polinomi mostrata sopra, iterando rispettivamente altre 4 e 9 volte.

$$\hat{u}(t+5|t) = \frac{0.399z^{-5} - 0.461z^{-6} + 0.087z^{-7}}{1 - 1.202z^{-1} + 0.165z^{-2} + 0.066z^{-3}}\, u(t)$$

$$\hat{u}(t+10|t) = \frac{0.241z^{-10} - 0.267z^{-11} + 0.050z^{-11}}{1 - 1.202z^{-1} + 0.165z^{-2} + 0.066z^{-3}}\, u(t)$$

Per valutare la bontà dei predittori ottenuti si effettua, come per l'identificazione, un'analisi dei residui. In questo caso però, non considerando predittori ad un passo, ci aspettiamo che la funzione di correlazione non sia quella tipica di un white noise, ma di un processo MA(*h*), dove *h* è il numero di passi. Prendendo come riferimento il predittore a 2 passi, nel caso ottimale avremo una funzione di correlazione diversa da zero per un ritardo pari a zero o uno; uguale a zero per tutti gli altri valore di ritardo. Ciò è dovuto al fatto che, non predicendo il passo immediatamente successivo, quando si vuole predire il valore di un determinato istante temporale $\bar{t}$ si sfruttano le informazione dei dati fino all'istante $\bar{t} - 2$. Dal momento che, essendo l'istante $\bar{t} - 1$ non disponibile, non se ne può trarre alcuna informazione. Ciò porta ad un grafico della correlazione diverso da quello proprio di un white noise anche nel caso ottimale.

Da ricordare che, non essendo la funzione di correlazione calcolata a partire dall'equazione del predittore, ma dai residui ottenuti dall'uscita del predittore e dai dati reali, non si avranno necessariamente dei valori pari a zero, ma nel suo intorno; in altre parole, come in precedenza si deve tener conto di una certa tolleranza dal momento che si lavora con dati reali.

In figura è mostrata la funzione di autocorrelazione dei predittore a 2, 5 e 10 passi:

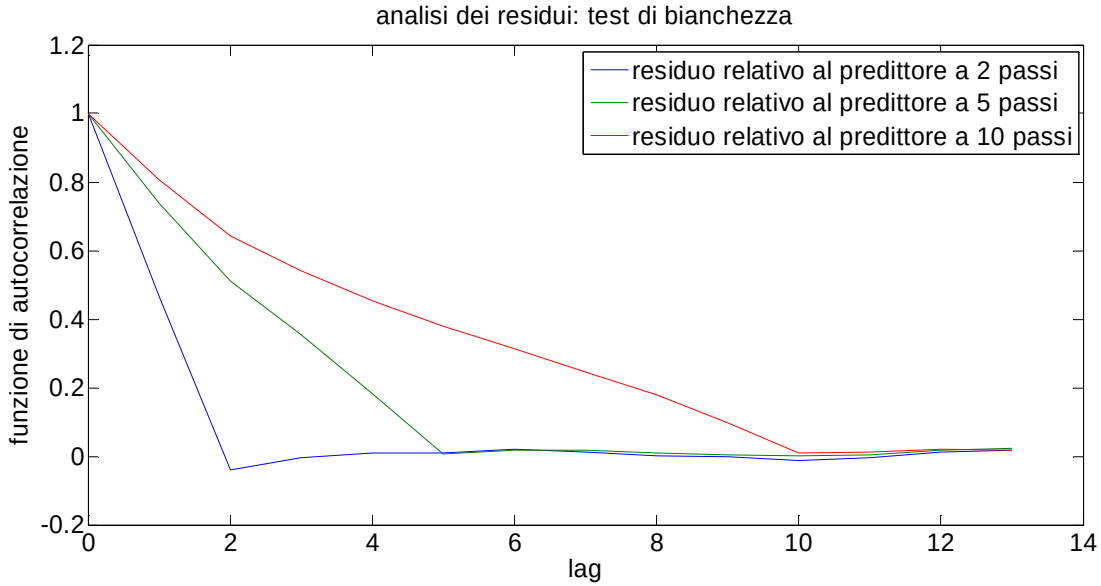


Figura 3.13: andamento delle funzione di correlazione dell'errore di predizione a due, cinque e dieci passi

Come si può osservare, effettivamente i valori di correlazione sono sensibilmente diversi da zero solo in corrispondenza di ritardi maggiori del numero di passi considerati. Ad esempio, la funzione del predittore a 5 passi assume valori nell'intorno dello zero solo a partire da un valore del ritardo pari a 6.

Utilizzando test di questo tipo si può concludere che i predittori ottenuti sono ottimali per un modello ARMA di ordine 3.

Si passa ora a mostrare i risultati ottenuti: per prima cosa si confronta la predizione della parte stocastica del traffico, successivamente si mostra la predizione per il traffico nella sua totalità.

In figura 3.14 si ha il confronto tra il predittore a 2 passi e i dati:

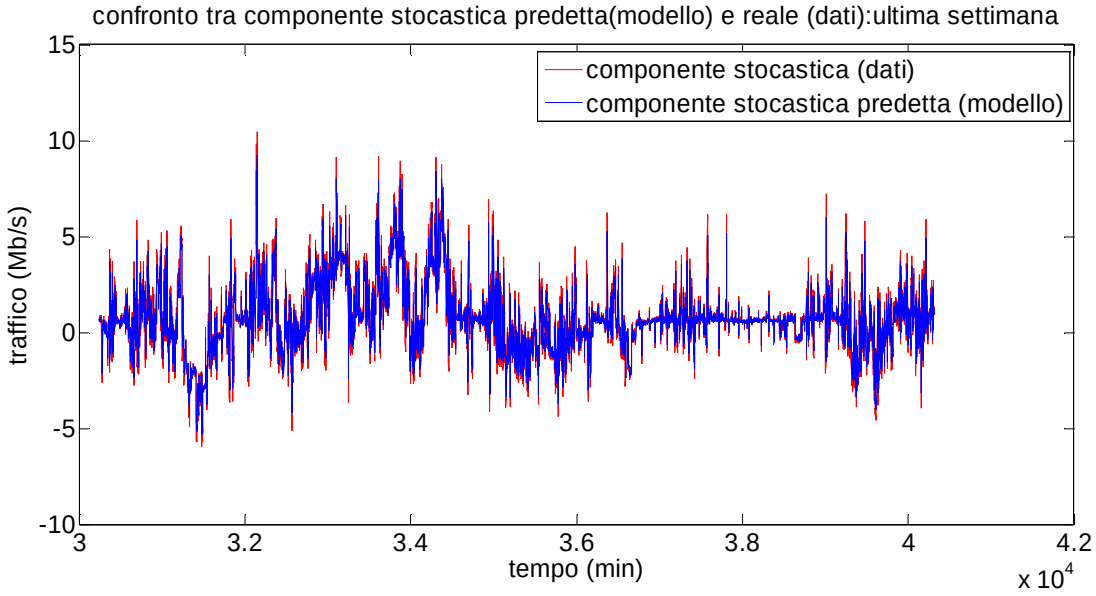


Figura 3.14: confronto tra la componente stocastica di traffico ottenuta a partire dai dati e l'output del predittore a 2 passi

La percentuale di fitting è mostrata in tabella:

	Predittore a 2 passi	Predittore a 5 passi	Predittore a 10 passi
Fitting	74.57%	62.33%	54.08%

Come si può vedere, per quanto riguarda il segnale detrendizzato, i predittori a 2, 5 e 10 passi danno dei risultati soddisfacenti.

Si passa ora a considerare il segnale totale. Da notare che, mentre nel capitolo precedente la stagionalità veniva calcolata considerando tutte le 4 settimane di dati disponibili, ora si considerano solo le prime 3. Questo perché, dovendo fare previsione a partire dalla fine della terza settimana, dobbiamo considerare non noti tutti i dati dell'ultima settimana, per poi usarli in un secondo momento per valutare la bontà del segnale predetto.

In figure 3.15 sono mostrati i risultati per il predittore a 2 passi:

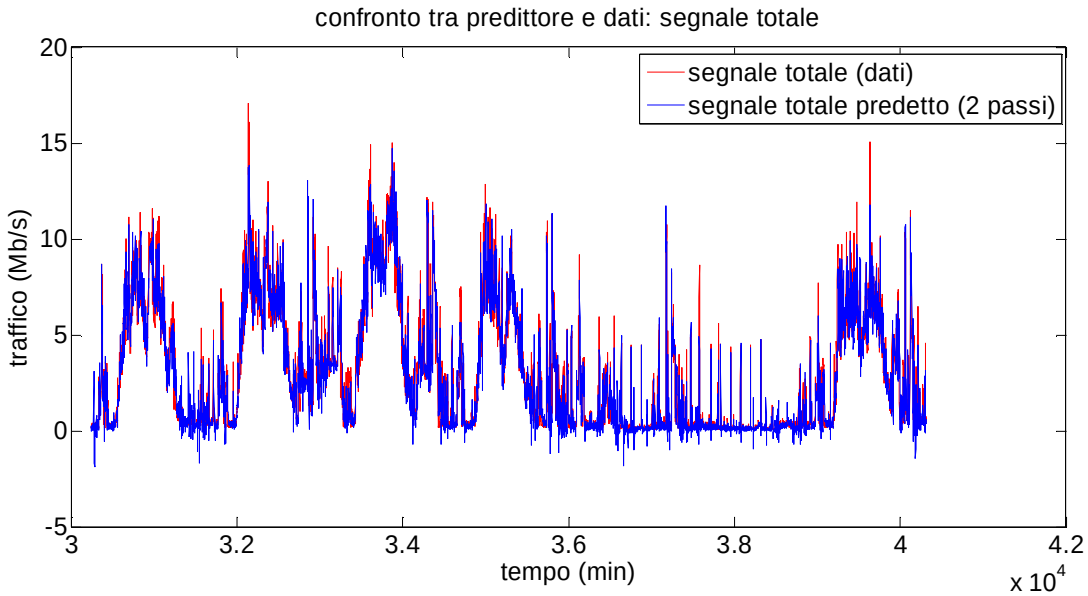


Figura 3.15: confronto tra il segnale totale di traffico e l'output del predittore a 2 passi

Come si può notare il traffico predetto assume valori negativi. Dal momento che ciò nella realtà non è possibile, si aggiunge al modello una saturazione, che pone a zero tutti i valori predetti negativi, mentre non modifica i valori restanti:

$$\hat{u}(t) = \begin{cases} \hat{u}(t) & \text{se } \hat{u}(t) \geq 0 \\ 0 & \text{se } \hat{u}(t) < 0 \end{cases}$$

In questo modo la predizione a 2 passi diventa:

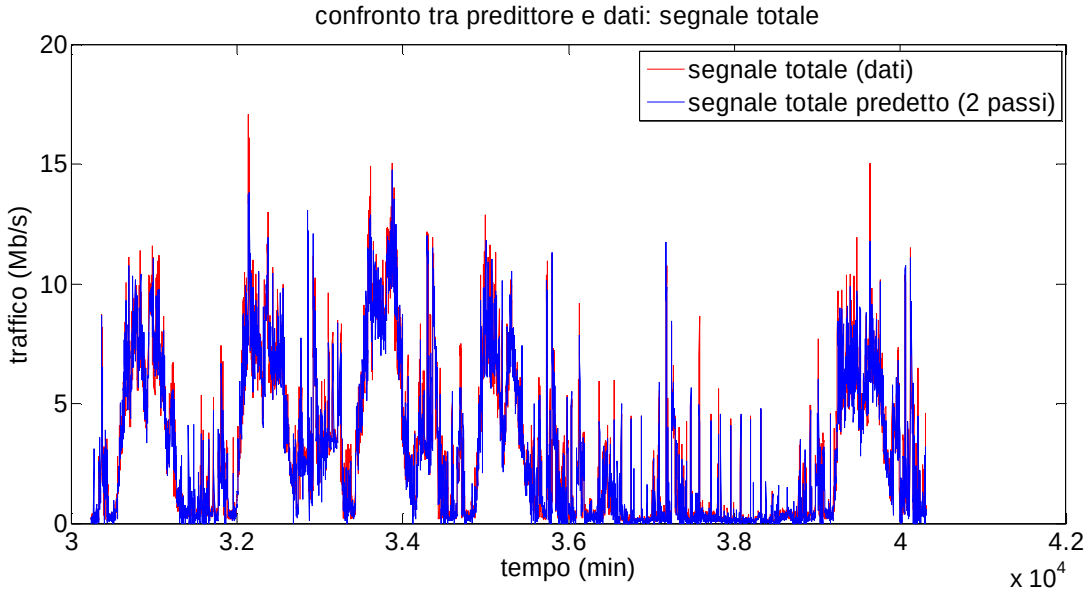


Figura 3.16: confronto tra il segnale totale di traffico e l'output del predittore a 2 passi con saturazione

In cui il segnale predetto non presenta più valori negativi, approssimando più fedelmente la realtà.

In figura 3.17 e 3.18 sono presentati i confronti tra segnali reali e predittori a 5 e 10 passi:

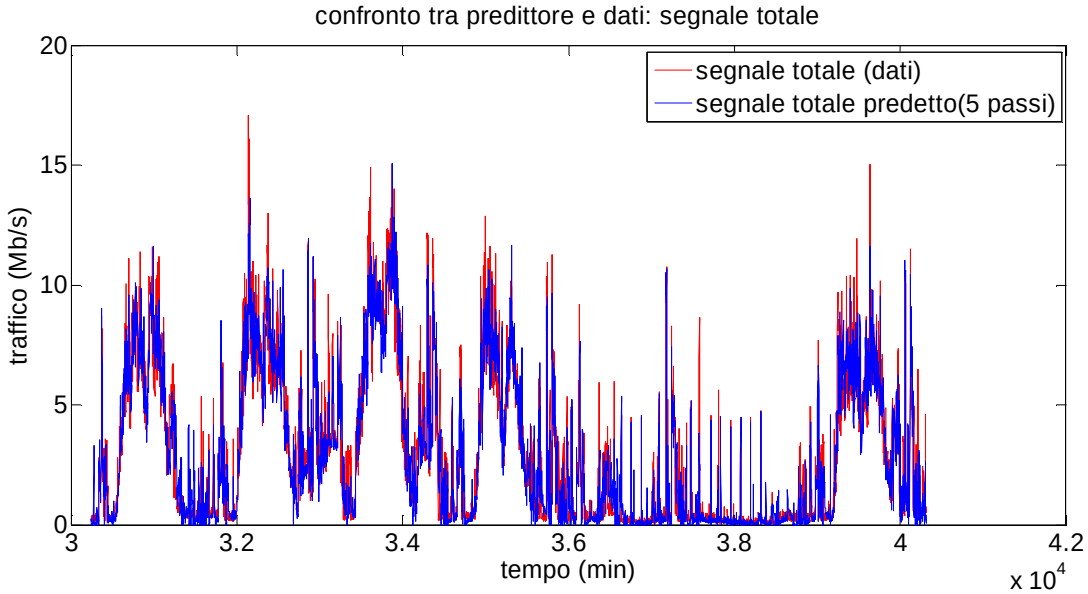


Figura 3.17:confronto tra il segnale totale di traffico e l'output del predittore a 5 passi con saturazione

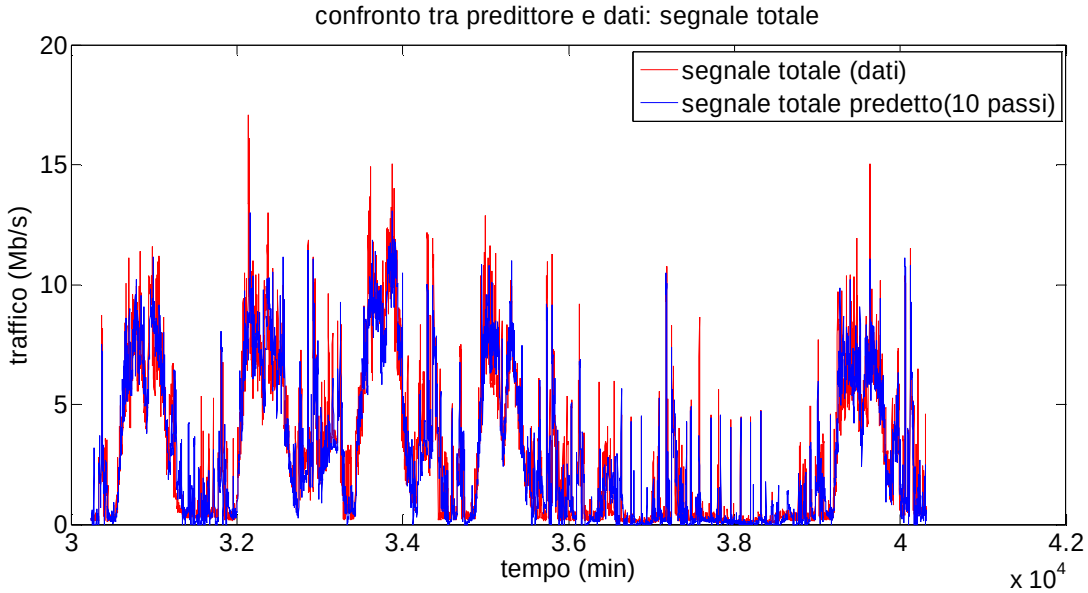


Figura 3.18:confronto tra il segnale totale di traffico e l'output del predittore a 10 passi con saturazione

In tabella sono presenti le percentuali di fitting:

	Predittore a 2 passi	Predittore a 5 passi	Predittore a 10 passi
Fitting	90.82%	86.43%	83.50%

Anche considerando il segnale totale, si può notare come la percentuale di fitting cali all'aumentare dell'orizzonte temporale. L'abbassamento delle prestazioni non è marcato come nel caso precedente in quanto l'apporto del segnale detrendizzato è solo una parte del totale, mentre nel caso precedente era l'unica grandezza in gioco. Il contributo fondamentale risulta essere quello relativo alla stagionalità ed al trend; ciò può essere dimostrato calcolando il fitting che si

avrebbe se si considerasse una predizione formata solamente dalle parti deterministiche del segnale, pari a trend e stagionalità:

$$fitting = \left(1 - \frac{1}{N}\frac{\sum_{i=1}^{N}\left(y_i - \overline{y_i} - s_y\right)^2}{var(y)}\right) * 100 = 65.13\%$$

Si è potuto verificare che la percentuale di fitting in questo caso si discosta del 8% circa rispetto a quella di un predittore a 100 passi, segno che il predittore per la parte detrendizzata del segnale tende alla media (pari a 0) in un numero di passi limitato (circa 300). Essendo questo un problema legato ad una predizione di tipo long-range, si rimanda al capitolo successivo per una trattazione più esaustiva.

Per un ulteriore confronto dei diversi predittori, in figura 3.19 è mostrato l'andamento delle prestazioni al variare dell'orizzonte predittivo (in appendice 3.2 sono mostrate le immagini relative all'andamento dei predittori a 50 e 100 passi):

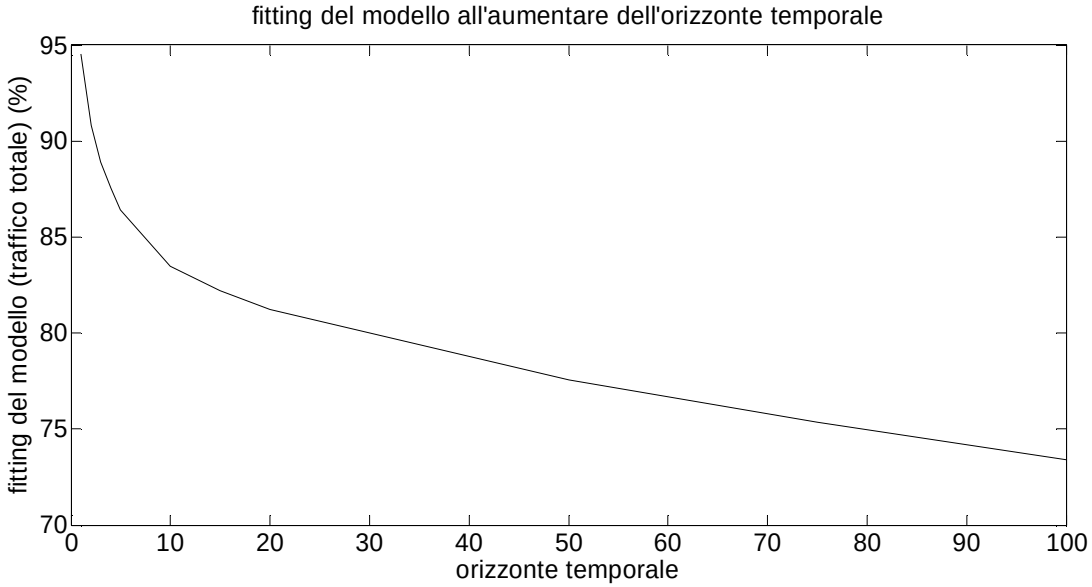


Figura 3.19: andamento dell'indice di fitting all'aumentare dell'orizzonte di predizione

3.1.4 Predizione dell'utilizzo di CPU

Ricavato il predittore del traffico, è possibile ricavare infine il predittore per l'utilizzo di CPU.

Avendo diviso il segnale nelle sue tre componenti si utilizza per ognuna di esse la relazione statica che collega traffico a CPU, moltiplicando ciascuna componente per il coefficiente K_i^O apposito.

In figura 3.20 e 3.21 sono mostrati i risultati relativi alla parte stocastica del segnale e al segnale totale considerando il predittore a 2 passi:

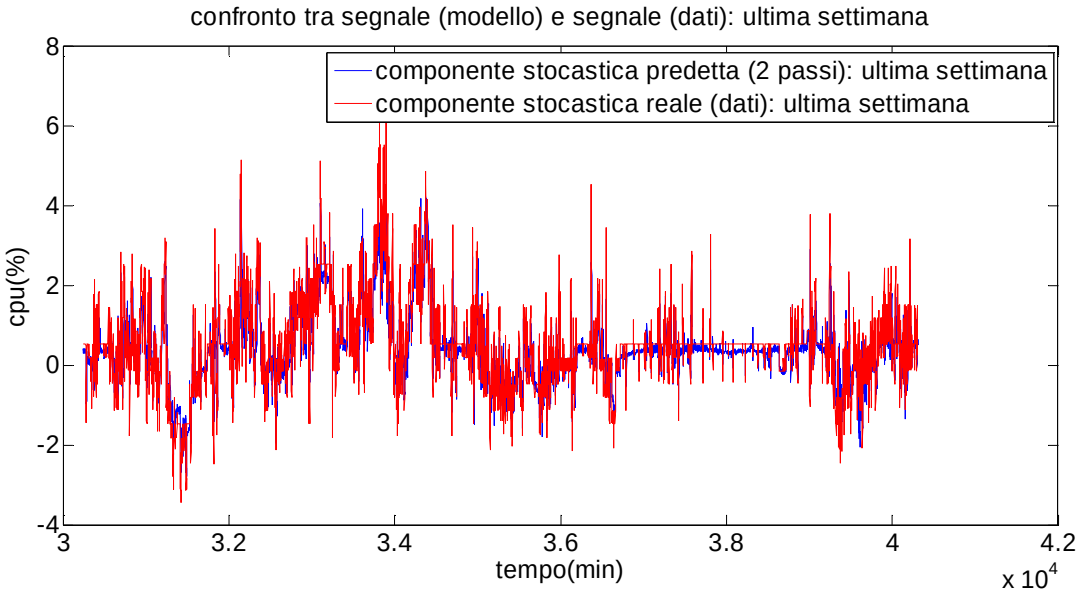


Figura 3.20 confronto tra il segnale totale di CPU e l'output del predittore a 2 passi

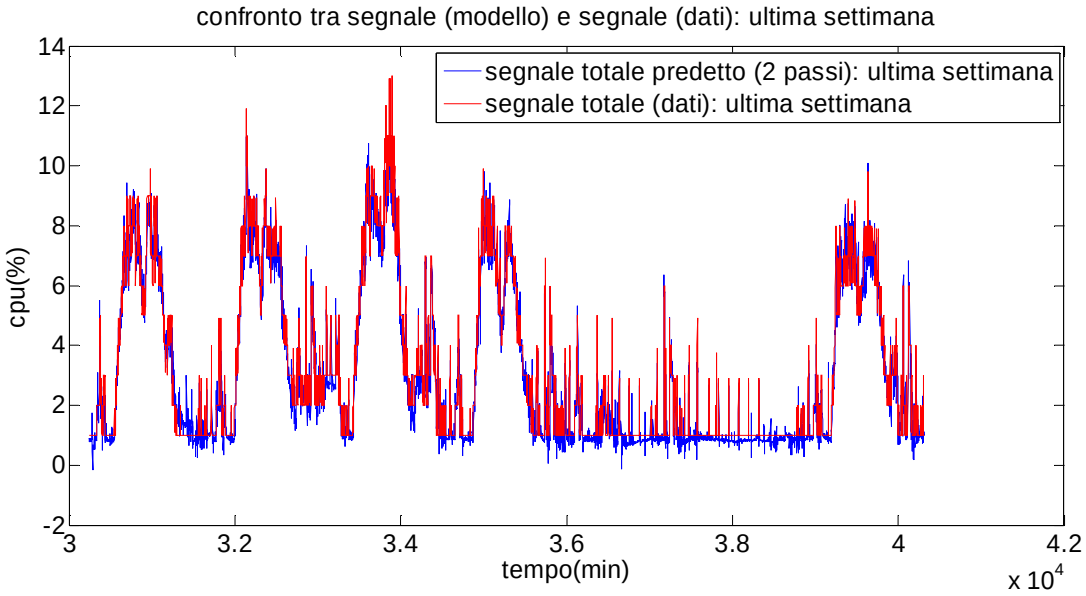


Figura 3.21 confronto tra il segnale totale di CPU e l'output del predittore a 2 passi

Come nel capitolo precedente, è necessario inserire una saturazione sul segnale totale, ponendo il limite inferiore che l'utilizzo che la CPU può assumere pari all'1%:

$$\hat{Y}(t) = \begin{cases} \hat{Y}(t) & se\ \hat{Y}(t) \geq 1 \\ 1 & se\ \hat{Y}(t) < 1 \end{cases}$$

In questo modo la previsione diventa:

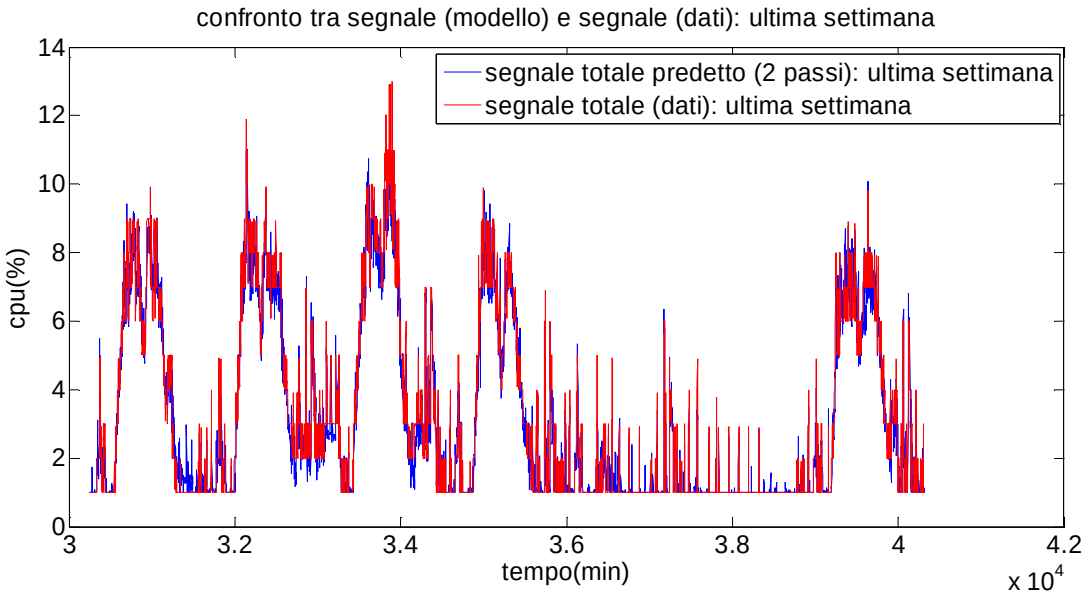


Figura 3.22: confronto tra il segnale totale di CPU e l'output del predittore a 2 passi con saturazione

Di seguito viene mostrato il confronto tra dati e il predittore a 10 passi, si rimanda in appendice per i grafici relativi ai confronti dei dati con i predittori a 5 passi.

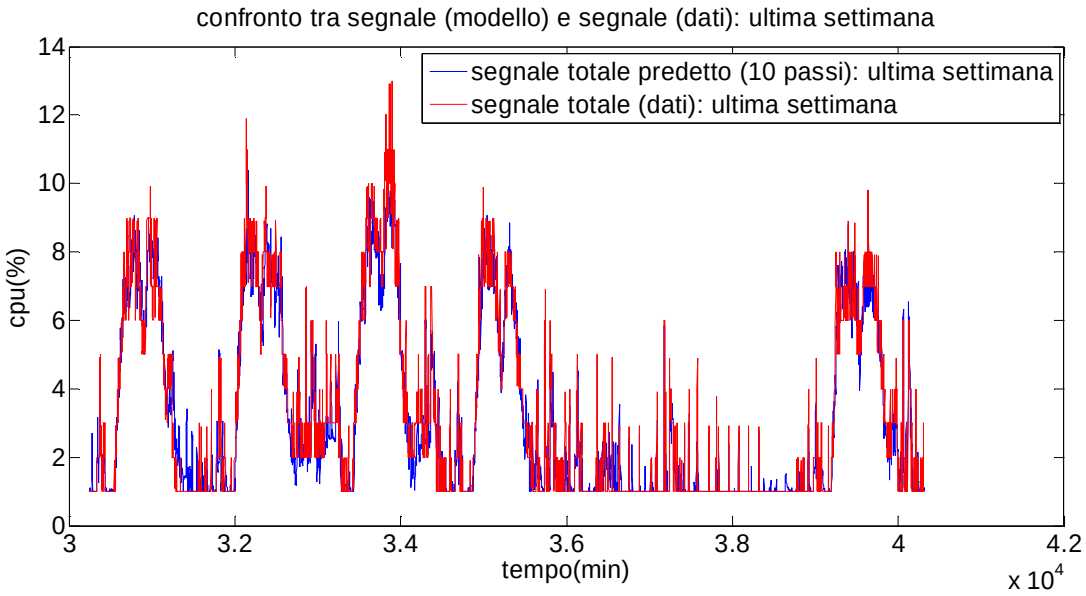


Figura 3.23: confronto tra il segnale totale di CPU e l'output del predittore a 10 passi con saturazione

In tabella sono invece presenti le percentuali di fitting (sempre riguardanti il segnale totale) relativi a tutti i predittori:

	Predittore a 2 passi	Predittore a 5 passi	Predittore a 10 passi
Fitting	96.04%	93.83%	92.45%

Si può osservare come le prestazioni diminuiscano all'aumentare dell'orizzonte predittivo, coerentemente a quanto osservato nel predittore per il traffico. Visto che la relazione utilizzata è una relazione statica, ci si poteva aspettare un andamento delle prestazioni simile a quello visto nel paragrafo precedente, fatto confermato dai dati riportati in tabella e dal grafico mostrato in figura 3.24:

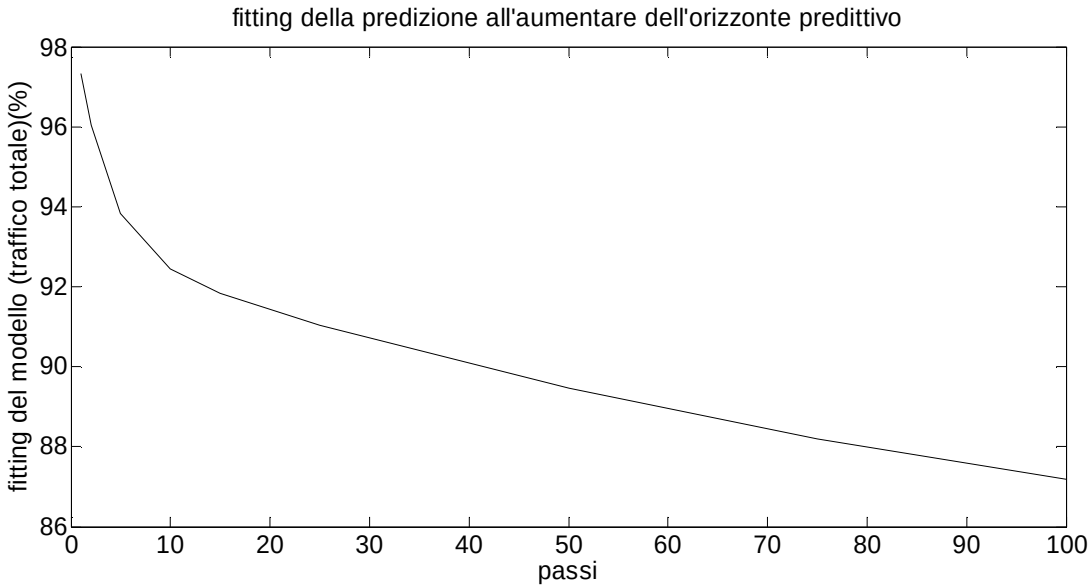


Figura 3.24: andamento dell'indice di fitting all'aumentare dell'orizzonte di predizione

Si può osservare come, per i primi 10-20 passi di predizione, si abbiano buone prestazioni del predittore, dal momento che in molti casi i due segnali risultano sovrapposti, fatto confermato dal basso valore di errore di predizione. Considerando l'applicazione per cui si è sviluppato lo studio di una predizione di tipo short-range (il load balancing illustrato a inizio capitolo) la predizione per un numero di passi pari a 10 porta ad avere un orizzonte predittivo di 10 minuti, sufficiente per tratte le conclusioni necessarie a ripartire in modo corretto le richieste in arrivo ad una server farm.

3.1.5 Conclusioni relative al modello ARMA

Terminando la trattazione per il modello ARMA, si possono trarre le seguenti conclusioni:

- Un modello di tipo ARMA risulta essere una scelta adeguata per la modellizzazione del traffico sulla rete considerata. Considerando prima il set di identificazione e in un secondo momento il set di validazione, si è in grado di osservare, attraverso le percentuali di fitting soddisfacenti, come le dinamiche del segnale (della parte stocastica) siano ben catturate dal modello ARMA.
- L'ordine del modello scelto risulta essere basso: un suo aumento avrebbe portato a miglioramenti delle prestazioni minime, a scapito della semplicità del modello; semplicità mantenuta invece scegliendo un valore pari a 3 sia per la parte AR che per la arte MA del modello.
- Considerando il predittore per il traffico, si può notare come le prestazioni siano particolarmente buone per l'orizzonte temporale considerato (2-10 passi): l'apporto dato dalla componente stocastica predetta dal modello ARMA contribuisce ad ottenere una buona predizione del segnale; l'introduzione di una saturazione consente di eliminare eventuali valor negativi della predizione, facendo sì che tutti i valori predetti possano avere un significato fisico (non avrebbe senso avere un valore di traffico negativo per un determinato istante temporale).
- L'utilizzo di 3 coefficienti consente di avere una buona conversione dei valori predetti da traffico a CPU: come si vedrà nel seguito del capitolo, utilizzare un coefficiente unico porterà ad avere una leggera perdita di prestazione, perdita che per il modello ARMA viene evitata. Anche in questo caso la saturazione introdotta ha come scopo la modellizzazione di un lower bound della predizione dettato dalla realtà.

3.2 **Modello arima**

Terminata la trattazione del modello ARMA, si passa ora a considerare un nuovo approccio per la modellizzazione del traffico attraverso un modello ARIMA.

Le motivazioni fondamentali dell'utilizzo di tale modello rispetto ad un ARMA sono due:

- Un modello ARIMA consente di descrivere segnali non stazionari, permettendo in questo modo di utilizzare il traffico *u(t)* complessivo, senza doverne ricavare le componenti di stagionalità, trend e parte stocastica.
- Non dovendo ricavare una stagionalità, non si ha bisogno di un alto numero di dati: nel corso della trattazione del modello ARMA, si sono utilizzate le 3 settimane di identificazione per ricavare la componente stagionale del segnale; in questo modo ogni punto era ottenuto come la media pesata di 3 campioni. Si può perciò capire come la presenza di un singolo outlier possa portare a valori puntuali che non rappresentano con fedeltà una situazione standard. In questa trattazione, non avendo il traffico misurato valori anomali, ciò non è stato evidenziato. Dal momento che in realtà si hanno occasioni in cui eventi imprevedibili portano a traffici anomali, utilizzare un set di sole 3 settimane potrebbe non essere sufficiente; per questo in un modello ARMA idealmente si dovrebbero considerare il maggior numero di dati possibile per ricavare una rappresentazione della componente deterministica il più fedele possibile. In un modello ARIMA, invece, non avendo bisogno di modellizzare alcuna componente, non si ha bisogno di un numero idealmente alto di dati, ma solo di un numero di dati tale da ricavare con buona precisione i parametri del modello; come si mostrerà nelle prossime pagine, dati per un totale di 3 settimane risultano essere sufficienti.

Questi due motivi sono stati i fattori che hanno portato alla scelta dello sviluppo del modello ARIMA a fianco degli altri precedentemente elencati. Si procede per prima cosa ad un trattazione teorica del modello, seguita poi da un procedimento analogo a quello seguito per lo sviluppo del modello ARMA.

3.2.1 Modello ARIMA: trattazione teorica

L'equazione che descrivere un modello ARIMA è la seguente:

$$\Phi(z^{-1})\nabla^d X(t) = \Theta(z^{-1})e(t)$$

In cui:

- $X(t)$ rappresenta l'uscita del modello, nel nostro caso il traffico *u(t).*

- $e(t)$ è un white noise a media nulla e rappresenta il segnale di ingresso.

- Φ e Θ sono i polinomi, di ordine rispettivamente p e q, che descrivono la parte autoregressiva e la parte a media mobile del modello; corrispondono ai polinomi φ e ϑ di un modello ARMA.

-∇ è l'operatore di differenziazione, applicato al segnale $X(t)$, tale che:

$$\nabla X(t) = X(t) - X(t-1) = (1 - z^{-1})X(t)$$

- d, esponente del termine ∇, determina il numero di volte che la differenziazione deve essere iterata.

L'operazione preliminare di differenziazione è ciò che permette al modello di poter descrivere segnali non stazionari; infatti se esso presenta un trend polinomiale, (o che possa essere ricondotto ad una tale forma, ad esempio attraverso uno sviluppo di Taylor) questo può essere eliminato operando una numero di differenziazioni pari all'ordine del polinomio che descrive il trend. In altre parole, individuato il valore da attribuire al parametri d, ci si può ricondurre ad un nuovo segnale, stazionario, e modellizzarlo attraverso un modello ARMA.

Se ad esempio consideriamo un segnale con trend parabolico:

$$X(t) = \beta_0 + \beta_1 t + \beta_2 t^2 + w(t)$$

In cui w(t) rappresenta un white noise, ponendo d=2 si ottiene:

Prima differenziazione:

$\nabla X(t) = X(t) - X(t-1) =$

$= \beta_0 + \beta_1 t + \beta_2 t^2 + w(t) - \beta_0 - \beta_1(t-1) - \beta_2(t-1)^2 - w(t-1) =$

$= w(t) - w(t-1) + \beta_1 - \beta_2 + 2\beta_2 t$

$\nabla X(t-1) = X(t-1) - X(t-2) =$

$= \beta_0 + \beta_1(t-1) + \beta_2(t-1)^2 + w(t-1) - \beta_0 - \beta_1(t-2) - \beta_2(t-2)^2 - w(t-2) =$

$= w(t-1) - w(t-2) + \beta_1 - 3\beta_2 + 2\beta_2 t$

Seconda differenziazione:

$$\nabla^2 X(t) = \nabla X(t) - \nabla X(t-1) =$$

$$= w(t) - w(t-1) + \beta_1 - \beta_2 + 2\beta_2 t - w(t-1) + w(t-2) - \beta_1 + 3\beta_2 - 2\beta_2 t =$$

$$= w(t) - 2w(t-1) + w(t-2) + 2\beta_2$$

In cui il trend parabolico è stato rimosso.

In figura 3.25 e 3.26 si può vedere la differenza tra segnale $X(t)$ e segnale $\nabla^2 X(t)$: mentre nel segnale originale il trend rende il segnale non stazionario, l'operazione di differenziazione fa sì che esso venga eliminato, come si può vedere dall'andamento mostrato.

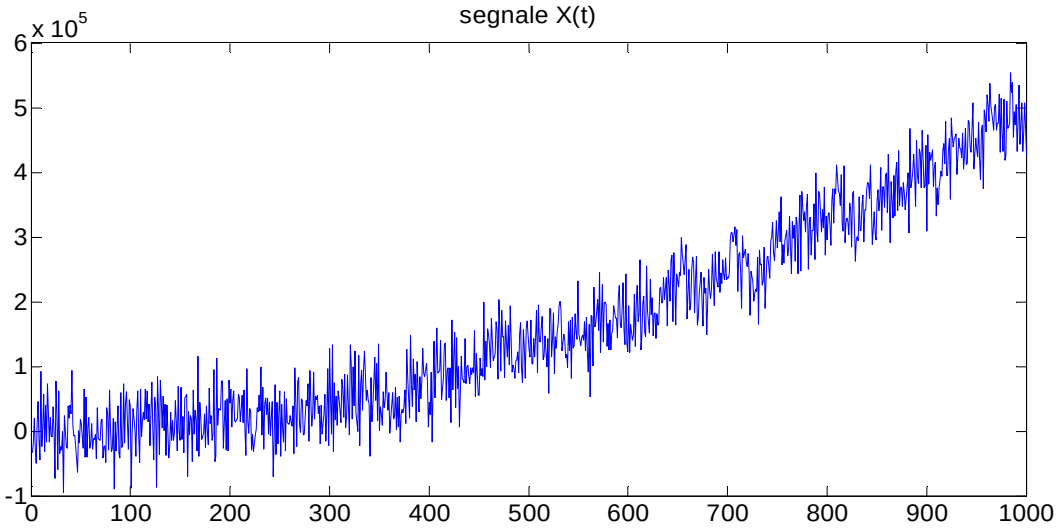


Figura 3.25: esempio di serie temporale con trend parabolico

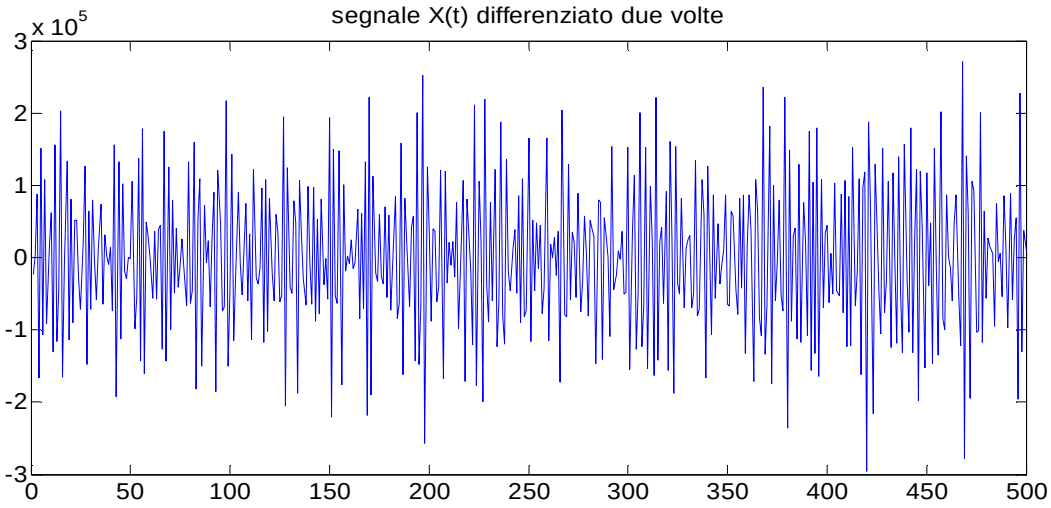


Figura 3.26: serie temporale ottenuta differenziando due volte il segnale originale

Da notare che, a seguito della differenziazione, non ci si può aspettare che i segnali ottenuti corrispondano alla parte stazionaria del segnale originale, essendo stata anche quest'ultima differenziata.

3.2.2 Identificazione del modello

Per quanto riguarda l'identificazione del modello si procede in maniera analoga al modello ARMA, utilizzando le prime tre settimane dei dati a disposizione per l'identificazione e la quarta per la validazione e per il confronto con la predizione, tenendo però conto che si deve identificare un ulteriore parametro, ossia *d*, numero di volte per cui si deve differenziare il segnale.

Per prima cosa si identifica tale parametro, dal momento che al suo variare varierà il segnale che si vuol modellizzare.

In letteratura non sono stati riscontrati metodi formali che portino all'individuazione del valore del parametro; l'approccio generalmente seguito consiste nel differenziare ripetutamente il traffico fino ad ottenere un segnale stazionario; oltre ad un generica indicazione sul fatto che solitamente un valore di *d* pari a 1 o 2 è sufficiente a ottenere la stazionarietà [6]. Osservando i dati a disposizione e ricordandosi che il trend individuato in precedenza era lineare, si potrebbe esser tentati dal porre arbitrariamente il parametro pari ad 1. In realtà si deve ricordare che la non stazionarietà del segnale non è data solo dal trend, ma anche dalla componente stagionale, che fornisce il maggior contributo del segnale totale; questo ci impedisce di poter determinare a prescindere il valore del parametro.

Per questo, procediamo col differenziare il traffico e a verificare se le condizioni di stazionarietà sono verificate.
Un segnale stazionario presenta:
-media costante
-varianza costante
-funzione di autocovarianza $\gamma(t_1,t_2)$ (e quindi anche la funzione di autocorrelazione) che dipende solo da $\tau = t_2 - t_1$ e non dai singoli istanti t_i in cui essa viene calcolata.

In figura 3.27 si possono osservare l'andamento della media (a sinistra) e della funzione di autocorrelazione (a destra) calcolata in settimane diverse relative al traffico differenziato una prima volta.

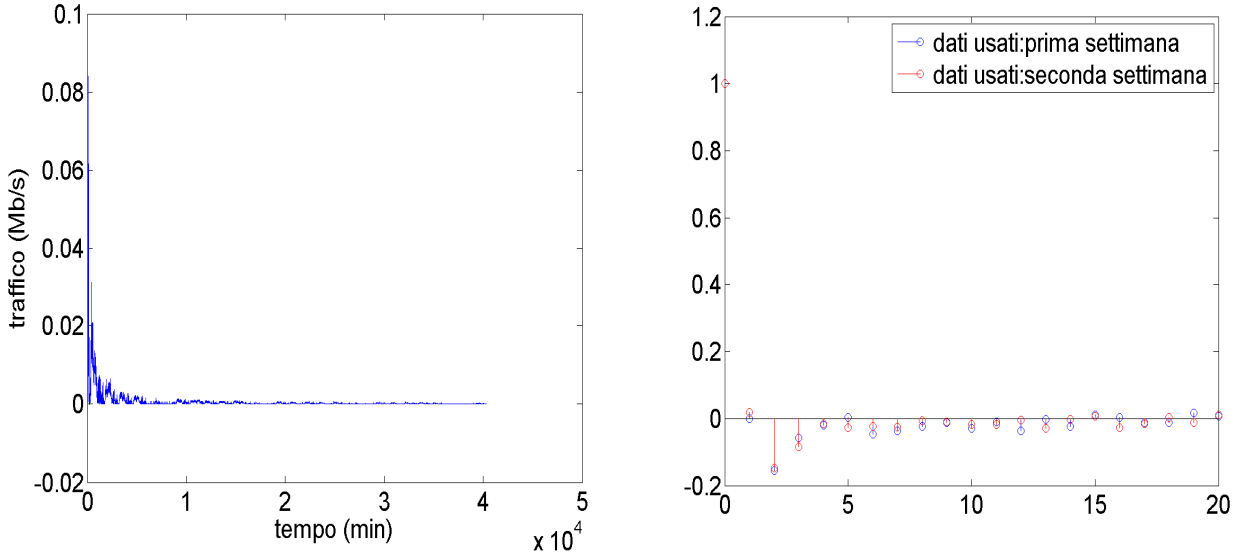


Figura 3.27: media ed autocorrelazione del segnale differenziato una volta

Per quanto riguarda la media, dopo un iniziale transitorio, dovuto al fatto che si sta utilizzando la media campionaria, si osserva un valore costante leggermente diverso da zero. Gli andamenti dell'autocorrelazione nelle due settimane considerate sono all'incirca uguali, in particolare per i primi valori del ritardo τ; si deve inoltre considerare una minima tolleranza per i valori intorno allo zero, così come era considerata nello studio della correlazione dei residui nella predizione usando il modello ARMA.

Per verificare se anche la varianza del segnale sia costante, è sufficiente osservare che i valori di autocovarianza per $\tau = 0$ coincidono al variare della settimana considerata.

Differenziando una seconda volta si ottiene:

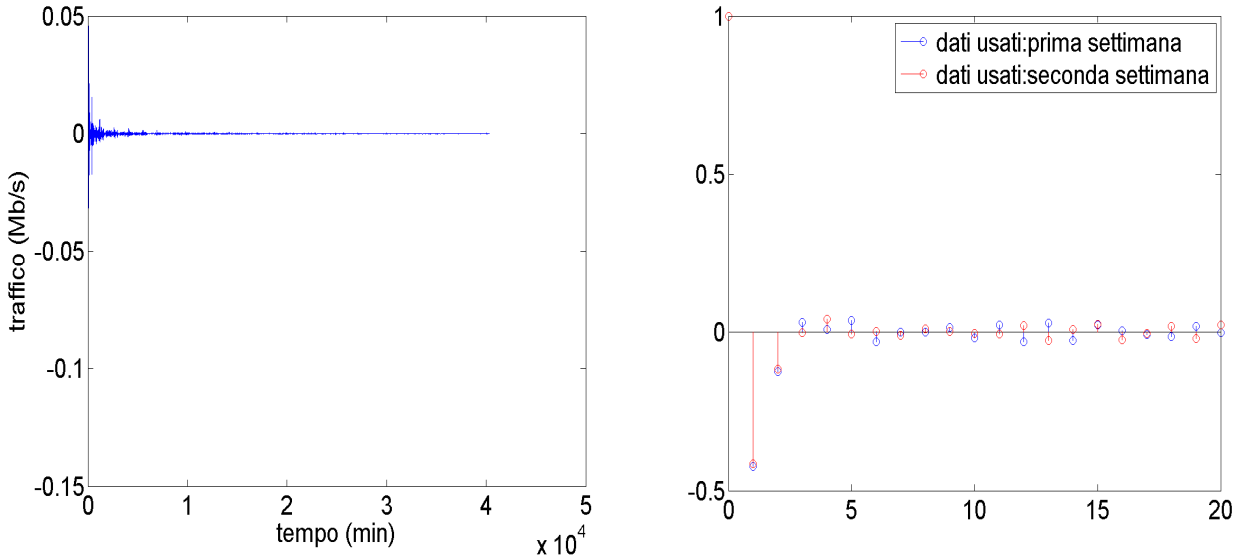


Figura 3.28: media ed autocorrelazione del segnale differenziato due volte

Di nuovo la media è circa pari a zero, mentre la funzione di autocovarianza presenta comportamenti simili nelle 2 settimane considerate, non mostrando un numero di passi di ritardo τ che presentino valori identici maggiore che nel caso precedente. Iterando nuovamente non si

ottengono risultato migliori; consideriamo per cui il segnale stazionario dopo un singolo passo di differenziazione, valutando entro i limiti di tolleranza le discrepanze della funzione di autocovarianza ricavata utilizzando periodi diversi.

Studio dell'andamento delle cifre di merito

Determinato il parametro *d* e differenziato, l'identificazione dei restanti parametri segue lo stesso ordine usato per il modello ARMA; per cui per prima cosa se ne identifica l'ordine.

Considerando le cifre di merito già utilizzate precedentemente, i risultati sono mostrati in figura 3.29 (si mostra per brevità la sola Loss Function *J* normalizzata sulla varianza del segnale, per le restanti si fa riferimento all'appendice numero 3.3):

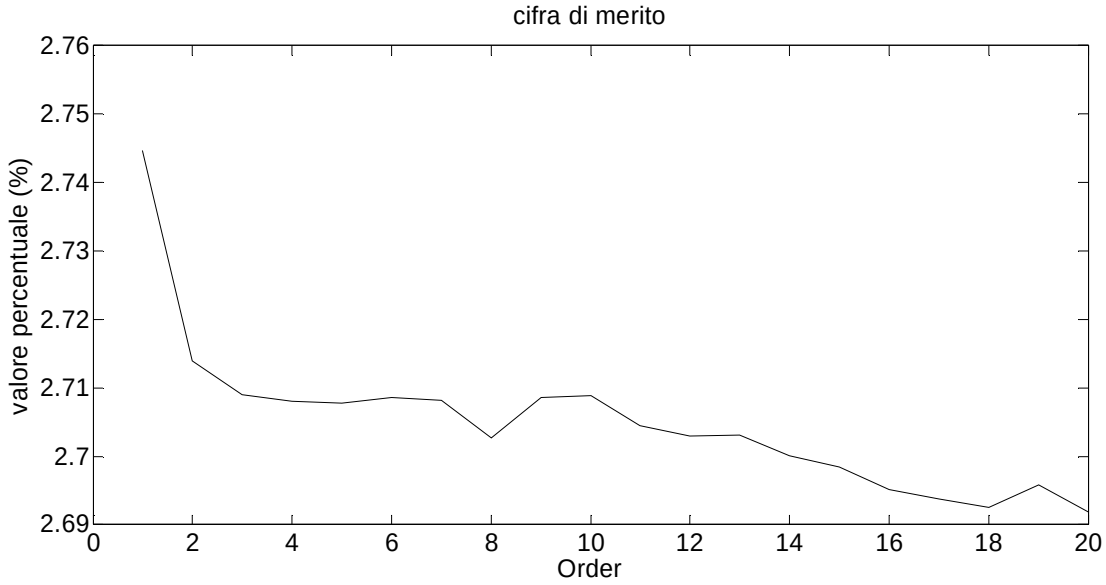


Figura 3.29: andamento della cifra di merito J normalizzata sulla varianza del segnale considerato

Dal grafico si può osservare come, dall'ordine 3 in poi, la cifra di merito tenda a mantenersi all'incirca costante: confrontando i valori relativi all'ordine 3 e all'ordine 20, si ha una differenza minore dello 0.03%, a scapito di un aumento dell'ordine considerevole.

Confrontando il valore su cui si attesta la cifra di merito con la corrispondente del modello ARMA, si può notare come essa sia molto inferiore: mentre in questo caso il valor medio è circa 2.7%, nel modello precedente ci si attestava ad un valore superiore al 16%.

Tale considerazione potrebbe essere fuorviante, in quanto farebbe pensare a prestazioni decisamente migliori da parte del modello ARIMA; in realtà ci si deve ricordare che in questo caso si sta considerando il segnale nella sua interezza, mentre nel caso precedente si considerava la sola componente stocastica; andando poi a sommare le 2 componenti deterministiche l'errore si riduceva a livelli simili a quelli che assume ora la cifra di merito.

Consideriamo ora le mappe di poli e zeri al variare dell'ordine del modello, per verificare se vi siano cancellazioni poli-zero; usando come campioni gli ordine 3 e 4 (come esempi di ordini bassi), l'ordine 8, per il quale si ha un leggero abbassamento di tutte le cifre di merito, e l'ordine 20 (esempio di ordine considerato alto).

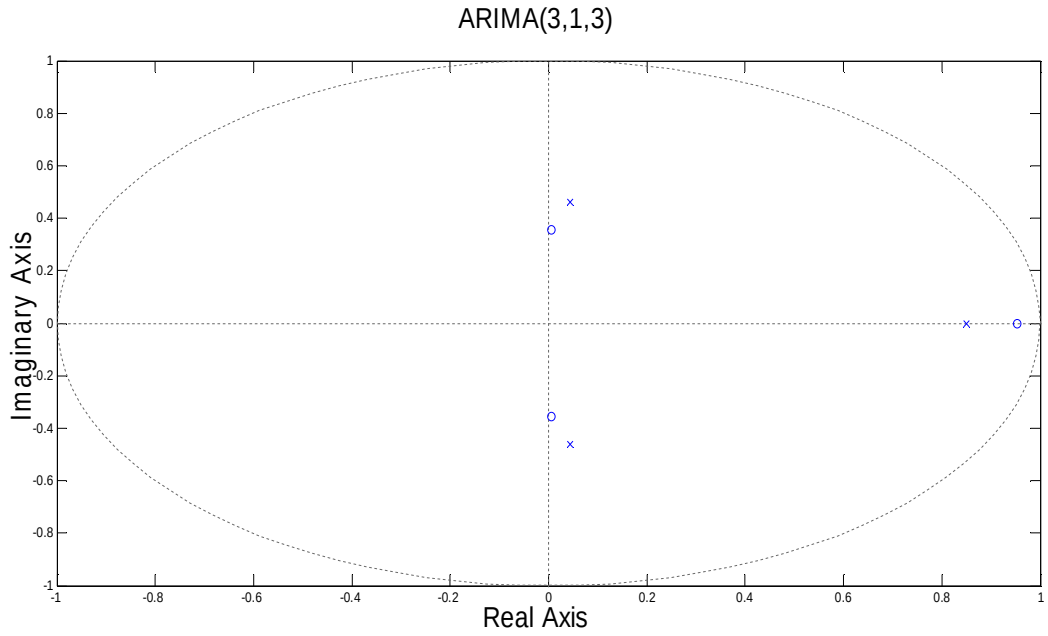


Figura 3.30: Mappa di poli e zeri relativa al modello ARIMA (3,1,3)

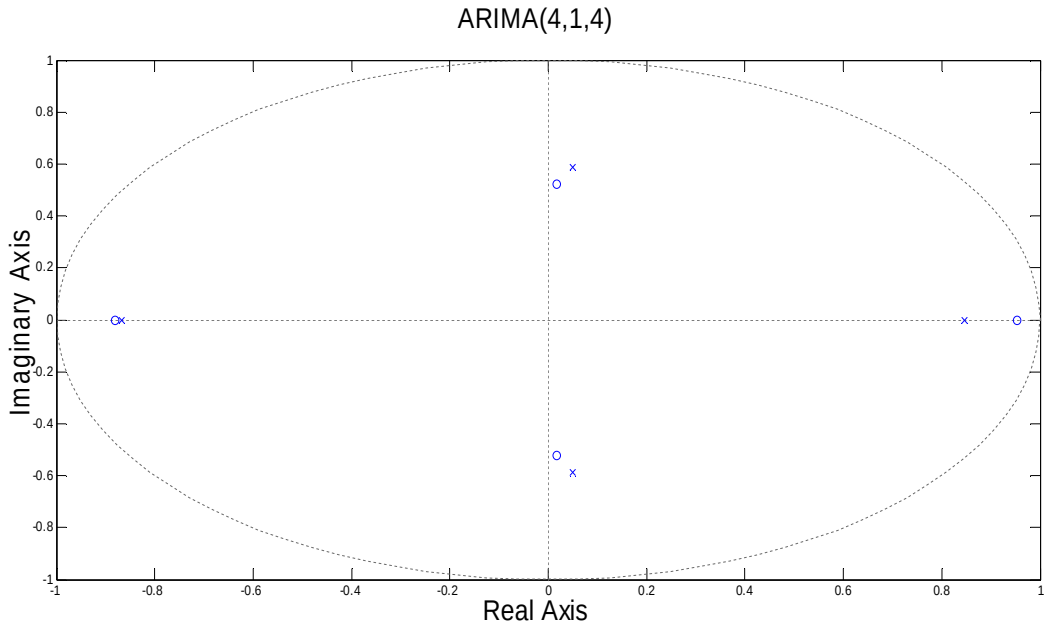


Figura 3.31: Mappa di poli e zeri relativa al modello ARIMA (4,1,4)

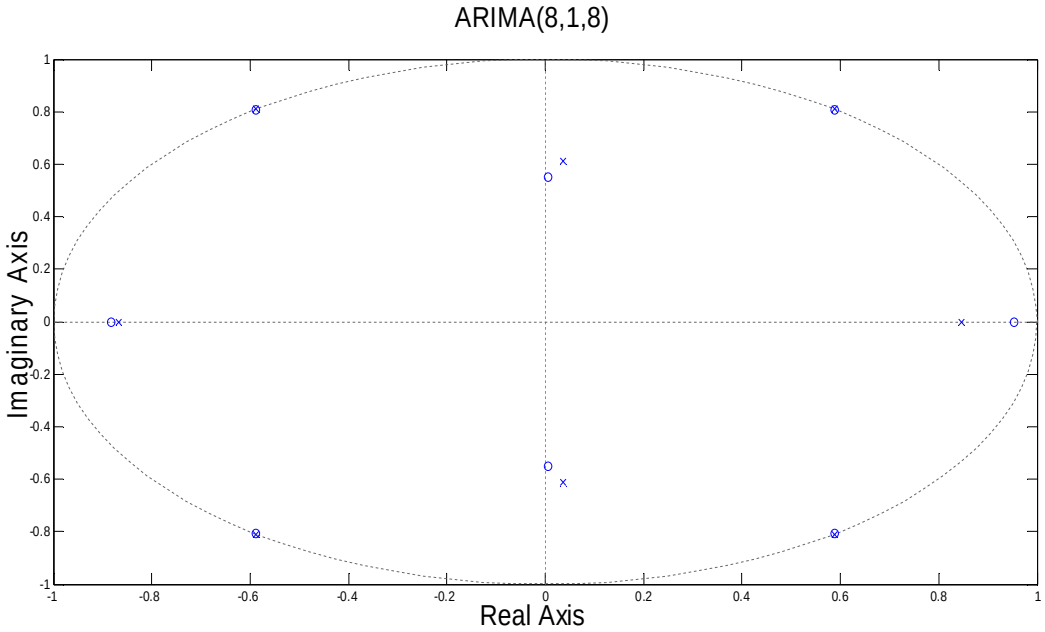


Figura 3.32: Mappa di poli e zeri relativa al modello ARIMA (8,1,8)

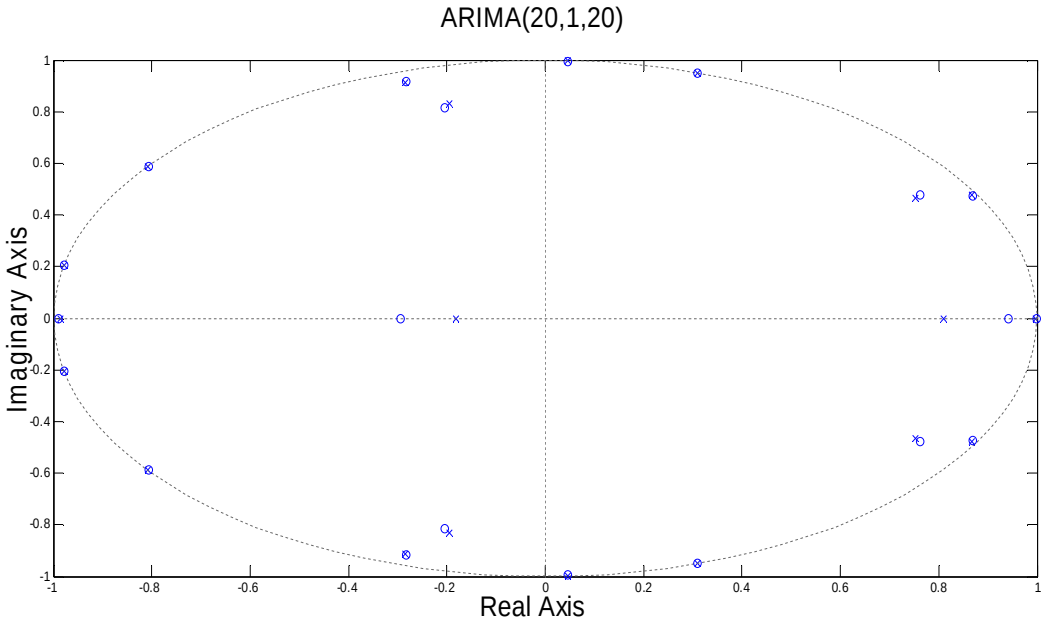


Figura 3.33: Mappa di poli e zeri relativa al modello ARIMA (20,1,20)

Come si può osservare, il numero di cancellazioni polo-zero (o di poli e zeri così vicini da avere scarso impatto sulla funzione di trasferimento) è tale da far sì che i poli non eliminati siano tra 2 e 4, portando alla conclusione che un ordine basso del modello sia preferibile.

Studio della funzione di trasferimento

Si passa ora a ricavare la funzione di trasferimento per i modelli degli ordini considerati e per i dati. In figura 3.34 sono mostrati i risultati:

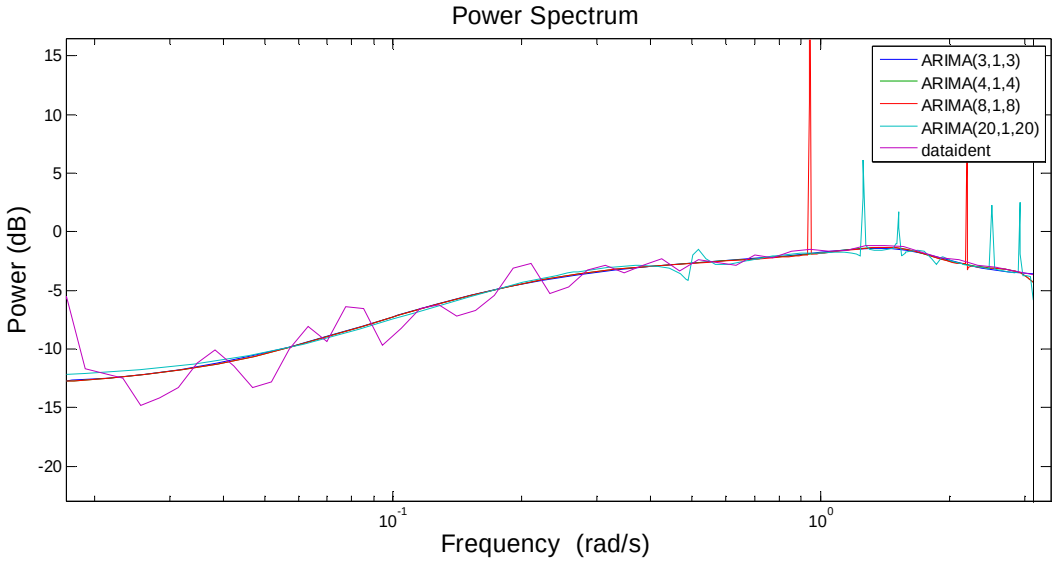


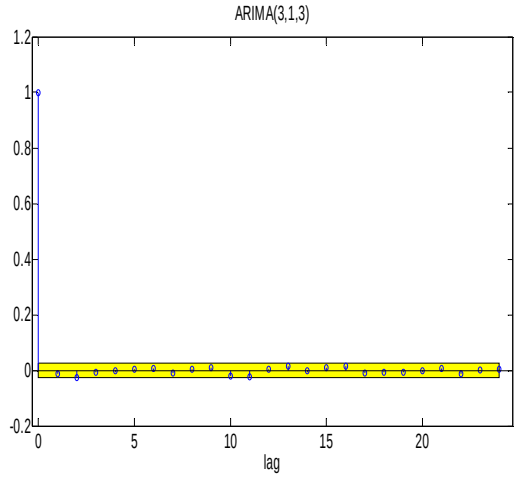
Figura 3.34: confronto tra funzione di trasferimento ottenuta a partire dai dati e le relative ai modelli considerati

Per quanto riguarda la funzione ricavata a partire dai dati, valgono le considerazioni fatte nella trattazione del modello ARMA. Si può notare come i modelli di ordine 8 e 20 presentino nel tratto finale delle frequenze considerate dei picchi, non presenti nei modelli di ordine 3 e 4 né tanto meno nei dati.

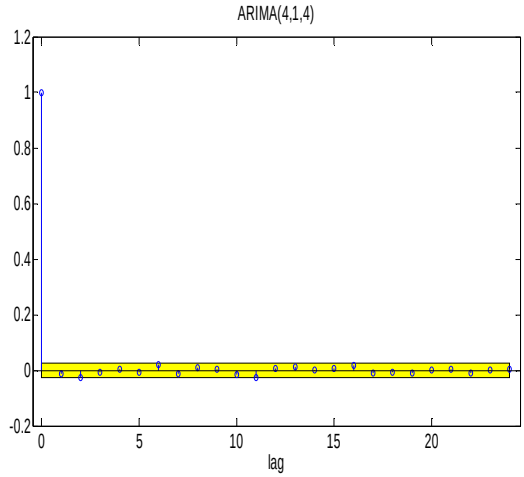
I modelli che meglio approssimano la realtà sono quindi quelli di ordine 3 e 4, avendo gli ordini successivi risonanze non presenti nella funzione di trasferimento ricavata dai dati.

3.2.3 Validazione

Considerando ora i dati relativi alla quarta settimana, si procede con la validazione dei modelli, ricavando i residui dell'errore di predizione, ottenuti come differenza tra i dati e l'uscita del predittore ad un passo. Per ciascun residuo si effettua un test di bianchezza andando a calcolarne l'autocorrelazione; come in precedenza ci si aspetta che tale funzione presenti valori sensibilmente non nulli solo per un valore del ritardo τ pari a zero.

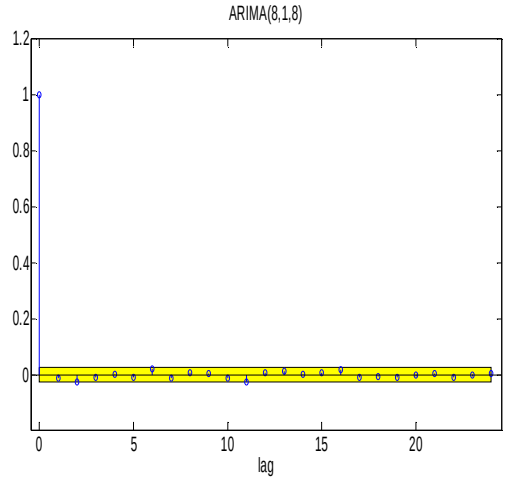


Residui relativi al predittore ricavato dal modello ARIMA(3,1,3)

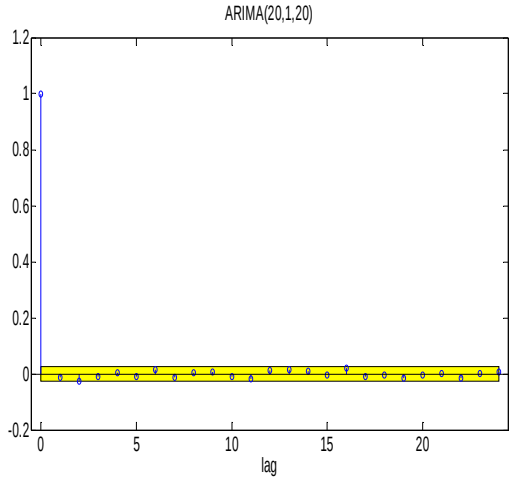


Residui relativi al predittore ricavato dal modello ARIMA(4,1,4)

Figura 3.35



Residui relativi al predittore ricavato dal modello ARIMA(8,1,8)



Residui relativi al predittore ricavato dal modello ARIMA(20,1,20)

Figura 3.36

Dalle figure 3.34 e 3.35 si può osservare come i risultati coincidano con ciò che ci si aspettava, essendo l'unico valore di autocorrelazione nettamente diverso da zero quello corrispondente a $\tau = 0$.

Potendo concludere che i predittori a un passo calcolati sono ottimi, si ricavano ora le percentuali di fitting per i vari ordini considerati confrontandoli con la settimana di validazione. In tabella sono presenti i risultati; si sottolinea ancora una volta come, a differenza del modello ARMA, si stia considerando il traffico totale *u* e non la sola componente stocastica δu.

	ARIMA(3,1,3)	ARIMA(4,1,4)	ARIMA(8,1,8)	ARIMA(20,1,20)
Fitting (%)	94.2797%	94.2770%	94.2889%	94.3028%

Dai risultati si può osservare come il fitting di tutti i modelli sia molto soddisfacente, e che sia circa costante: in altre parole, come si era già visto osservando le cifre di merito (ricavate però a partire dai dati di identificazione) un incremento dell'ordine non porta a un aumento sostanziale delle prestazioni del modello. Per questi motivi scegliamo di considerare un modello ARIMA(3,1,3).

In tabella sono mostrati i valori dei parametri del modello:

Ordine parte AR:	3
Polinomio $\Phi(z^{-1})$:	$1 - 0.936z^{-1} + 0.289z^{-2} - 0.183z^{-3}$
Ordine parte MA:	3
Polinomio $\Theta(z^{-1})$:	$1 - 0.965z^{-1} + 0.137z^{-2} - 0.120z^{-3}$
Numero differenziazioni:	1

Per finire, in figura 3.37 viene mostrato il confronto grafico tra i dati (quarta settimana) e il modello di ordine 3.

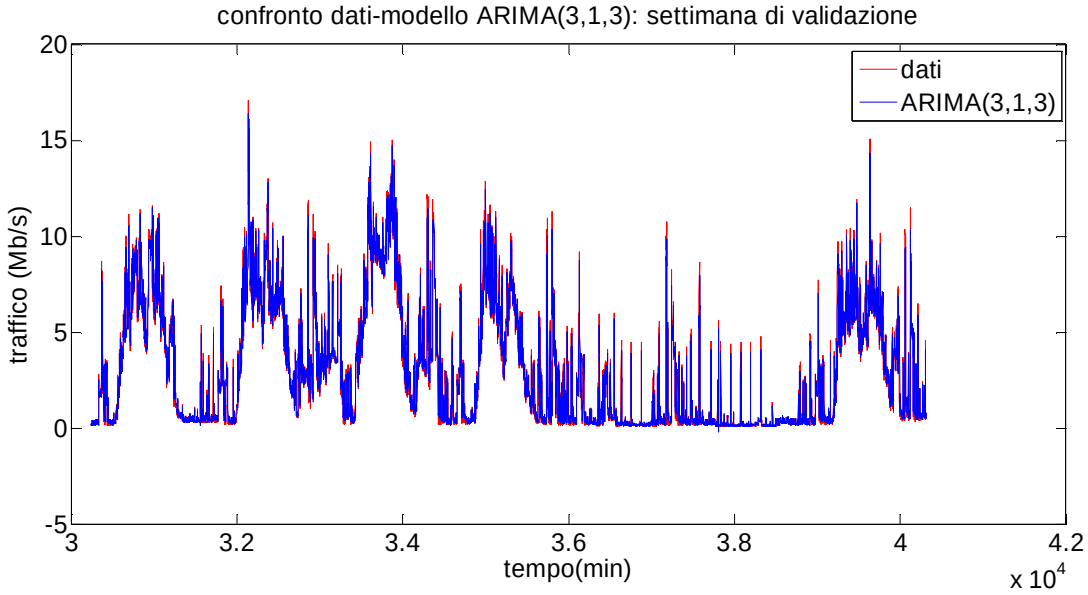


Figura 3.37: confronto tra il segnale totale di traffico e l'output del modello ARIMA(3,1,3)

3.2.4 Predittore del traffico in ingresso ad un router

Stabiliti i parametri del modello ARIMA, è ora possibile derivare a partire da quest'ultimo la predizione short-range obiettivo di questo capitolo. Si sceglie di considerare gli stessi orizzonti predittivi utilizzati nella trattazione del modello ARMA, in modo da poterne poi confrontare le predizioni: si mostreranno perciò i risultati relativi a predittore a due, cinque e dieci passi.

Il procedimento per ottenere il predittore si divide in 2 passi:

-il primo, identico a quello utilizzato per il modello ARMA, consiste nell'iterare la lunga divisione tra i polinomi Φ e Θ un numero di volte pari all'orizzonte predittivo, ricavare il resto della divisione Θ_r e a partire da esso ricavare il predittore per il segnale differenziato:

$$\nabla\hat{u}(t+h|t) = \frac{\Theta_r(z)}{\Phi(z)}\, e(t) = \frac{\Theta_r(z)}{\Phi(z)}\frac{\Phi(z)}{\Theta(z)}\, \nabla u(t) = \frac{\Theta_r(z)}{\Theta(z)}\, \nabla u(t)$$

-il secondo, aggiuntivo rispetto al procedimento per il modello ARMA, formato da un operazione di integrazione per ricondursi al segnale di traffico non differenziato $u(t)$.

Di seguito si riportano in formulazione estesa i predittori a 2, 5 e 10 passi, ottenuti iterando l'operazione di lunga divisione il corrispondente numero di volte:

$$\nabla\hat{u}(t+2|t) = \frac{-0.179z^{-2} - 0.0076z^{-3} - 0.0053z^{-4}}{1 - 0.965z^{-1} + 0.1372z^{-2} + 0.120z^{-3}}\nabla u(t)$$

$$\nabla\hat{u}(t+5|t) = \frac{-0.09212z^{-5} - 0.00190z^{-6} - 0.00215z^{-7}}{1 - 0.965z^{-1} + 0.1372z^{-2} + 0.120z^{-3}}\nabla u(t)$$

$$\nabla\hat{u}(t+10|t) = \frac{-0.044z^{-10} - 0.0037z^{-11} - 0.0095z^{-12}}{1 - 0.965z^{-1} + 0.1372z^{-2} + 0.120z^{-3}}\nabla u(t)$$

Prima di mostrar i risultati ottenuti dai predittori, si procede ad un ulteriore test di bianchezza sui residui dei predittori, simile a quello effettuato in fase di validazione.

Ciò che ci si aspetta all'aumentare dell'orizzonte predittivo considerato è un grafico dell'autocorrelazione in cui aumentano i valori di ritardo τ per cui la correlazione è sensibilmente diversa da zero; infatti come è noto dalla teoria, il residuo ottenuto a partire da un predittore a più passi non sarà un white noise, ma un segnale colorato. Ci aspettiamo perciò funzioni di autocorrelazione simili a quelle di un processo MA.

In figura 3.38 sono mostrati i risultati:

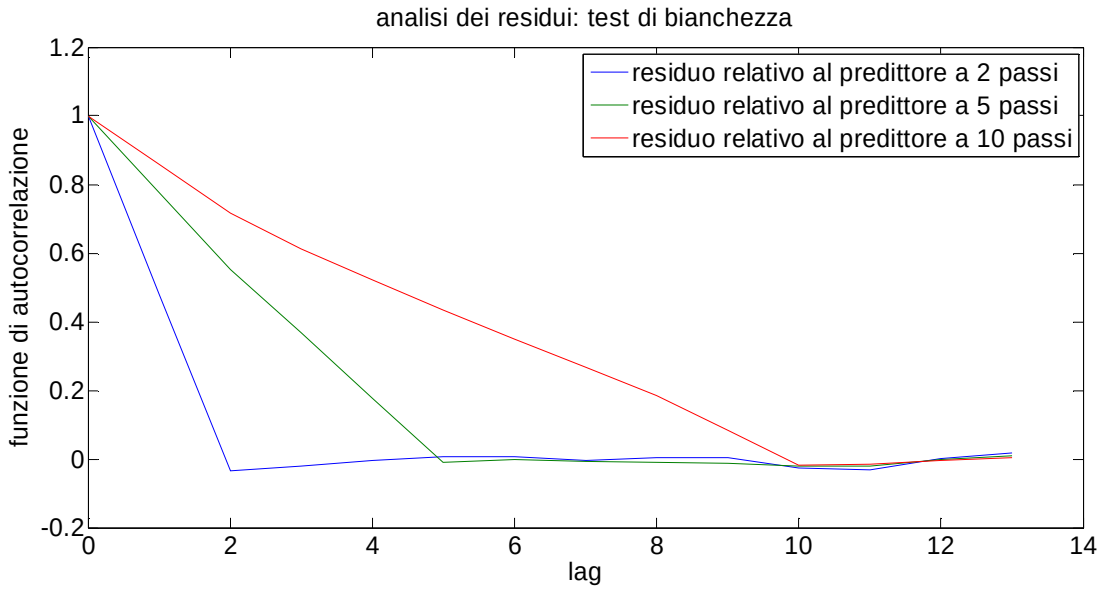


Figura 3.38: andamento delle funzione di correlazione dell'errore di predizione a due, cinque e dieci passi

Si osservi come dal grafico si può concludere che il residuo relativo al predittore a due passi sia un processo MA(1), il residuo relativo al predittore a 5 passi sia un MA(4) e così via, dimostrando che anche in questo caso i predittori sono ottimi.

Inoltre, anche in questo caso inseriamo una saturazione, che ha come obiettivo l'impedire al traffico di assumere valori negativi:

$$\hat{u}(t) = \begin{cases} \hat{u}(t) & \text{se } \hat{u}(t) \geq 0 \\ 0 & \text{se } \hat{u}(t) < 0 \end{cases}$$

Si passa ora ai risultati numerici; in tabella sono mostrate le percentuali di fitting relative ai vari orizzonti temporali:

	Predittore a 2 passi	Predittore a 5 passi	Predittore a 10 passi
Fitting (MSE)	89.01%	81.16%	73.81%

I grafici nelle figure 3.39 e 3.40 mostrano i risultati per i predittori a 2 e 5 passi, si rimanda all'appendice numero 3.4 per i restanti grafici per brevità.

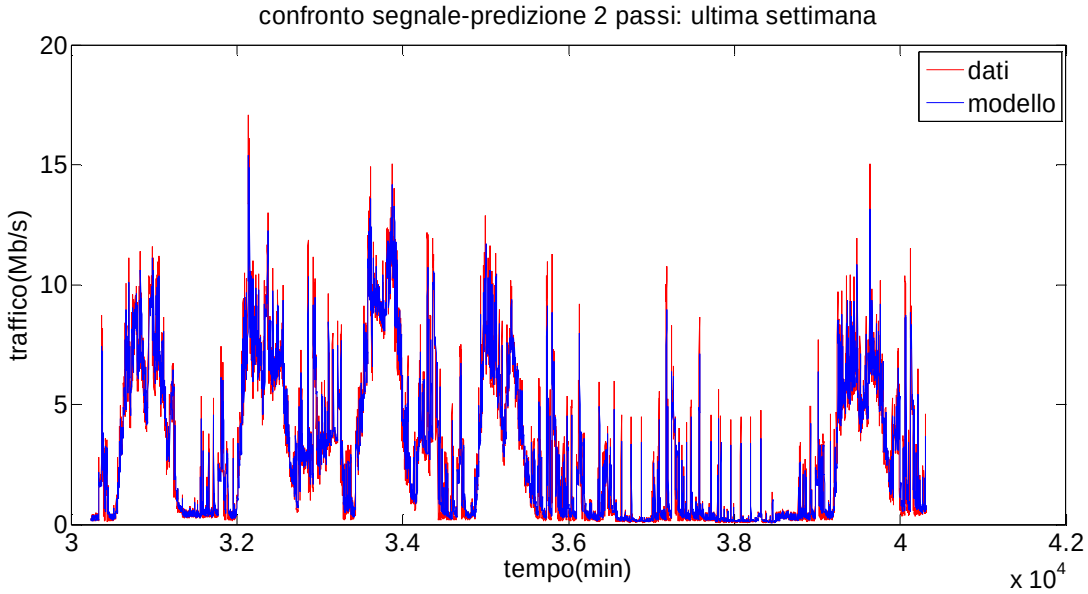


Figura 3.39: confronto tra il segnale totale di traffico e l'output del predittore a 2 passi con saturazione

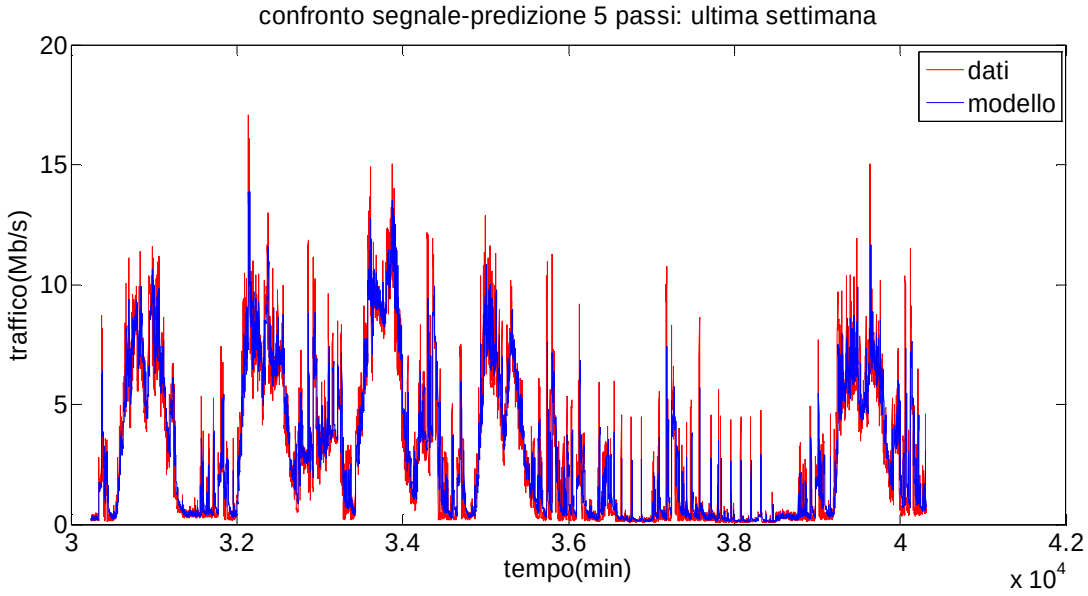


Figura 3.40: confronto tra il segnale totale di traffico e l'output del predittore a 5 passi con saturazione

Dai valori riportati in tabella si può osservare come il modello ARIMA presenti buone prestazioni solo per un orizzonte predittivo limitato: già per un orizzonte temporale pari a 10 si ha un fitting pari all'73%, risultato raggiunto dall'ARMA solo per un orizzonte temporale molto superiore.

In figura 3.41 si confronta la resa del modello ARIMA con il modello ARMA al fine di rendere chiara la differenza tra le due prestazioni.

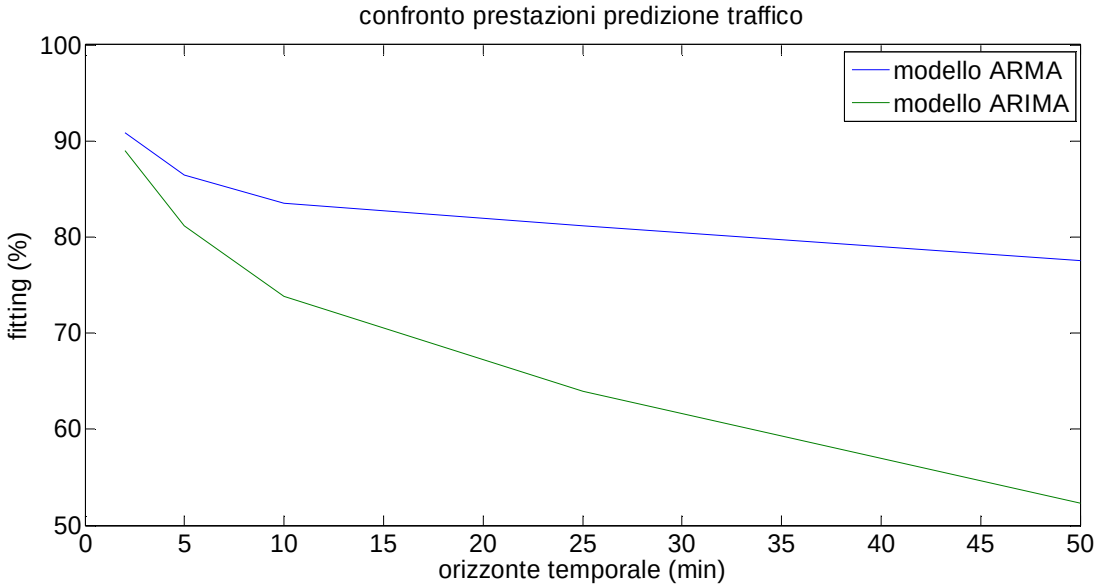


Figura 3.41: confronto dell'andamento del fitting all'aumentare dell'orizzonte predittivo per i modelli ARMA e ARIMA

Ci si chiede ora quale sia il motivo per cui una modello ARIMA, evoluzione di un modello ARMA, non abbia prestazioni superiori o per lo meno simili. La causa di ciò deve essere ricercata nel differente segnale che i due modelli trattano: mentre il segnale modellizzato dal modello ARIMA è il segnale totale u, il modello ARMA descrive solamente la componente stocastica δu. Se si osservano le percentuali di fitting relativi alla sola componente δu, ci si accorgerà come esse siano nettamente inferiori a quelle mostrate qui sopra. Il risultato finale è dato dal fatto che la maggior parte del segnale è formato dalla componente stagionale s_u, che essendo individuata con buona approssimazione nelle prime 3 settimane di identificazione, porta a buoni risultati in predizione. Dal momento che tale componente non viene impiegata nel modello ARIMA, il suo impatto positivo non interviene nelle prestazioni del modello. Tuttavia non bisogna dimenticare che aver bisogno di una stagionalità ben descritta comporti anche lati negativi, in quanto ciò richiede un lasso di tempo minimo per avere un numero sufficiente di campioni per poter determinare con buona approssimazione l'andamento della componente. Nel nostro caso, essendo i dati raccolti in un periodo relativamente stabile dell'anno, non è stato riscontrato tale problema; tuttavia, se si volesse prevedere l'andamento del traffico a partire dall'avviamento di una rete appena approntata, con un conseguente andamento non ciclo-costante ma con un iniziale transitorio di qualche settimana, o si raccogliessero dati in un periodo di traffico diverso dalla norma (un periodo estivo in cui il traffico registrato è minore del normale ad esempio), si potrebbero riscontrare problematiche nell'identificazione della componente stagionale del segnale. Non è quindi da scartare a priori l'utilizzo di un modello ARIMA, in particolare nelle situazioni sopra elencate.

3.2.5 Predizione di utilizzo di CPU

Come ultimo passo nella trattazione del modello ARIMA, si ricava la predizione dell'utilizzo di CPU.

Per far ciò si considera la relazione statica individuata nel capitolo 2 tra traffico e relativo impatto sulla risorsa CPU. Per far questo si deve considerare, a differenza del modello precedentemente studiato, una relazione che leghi le grandezze totali in gioco e attraverso un unico coefficiente. Per la determinazione di tale coefficiente si rimanda al capitolo 2.

Conformemente al modello ARMA, invece, si utilizza una seconda saturazione per limitare i valori di CPU ad un minimo pari all'1%, percentuale di utilizzo dovuta a funzioni proprie del router indipendenti dalla quantità di traffico in arrivo dalle varie interfacce:

$$\hat{Y}(t) = \begin{cases} \hat{Y}(t) & \text{se } \hat{Y}(t) \geq 1 \\ 1 & \text{se } \hat{Y}(t) < 1 \end{cases}$$

In tabella sono mostrati i valori di fitting per tutti i predittori considerati:

	Predittore a 2 passi	Predittore a 5 passi	Predittore a 10 passi
Fitting	90.20%	86.32%	81.94%

Come per la predizione di traffico, vengono mostrati qui i risultati grafici per i predittori a 2 e 5 passi, rimandando all'appendice 3.4 per il predittore a 10 passi.

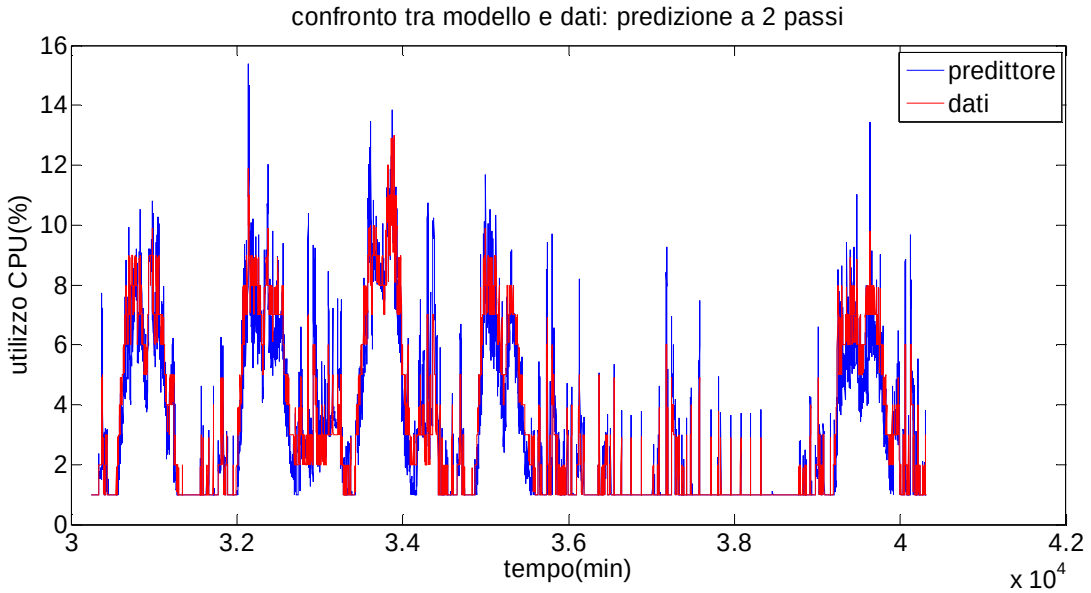


Figura 3.42: confronto tra il segnale totale di CPU e l'output del predittore a 2 passi con saturazione

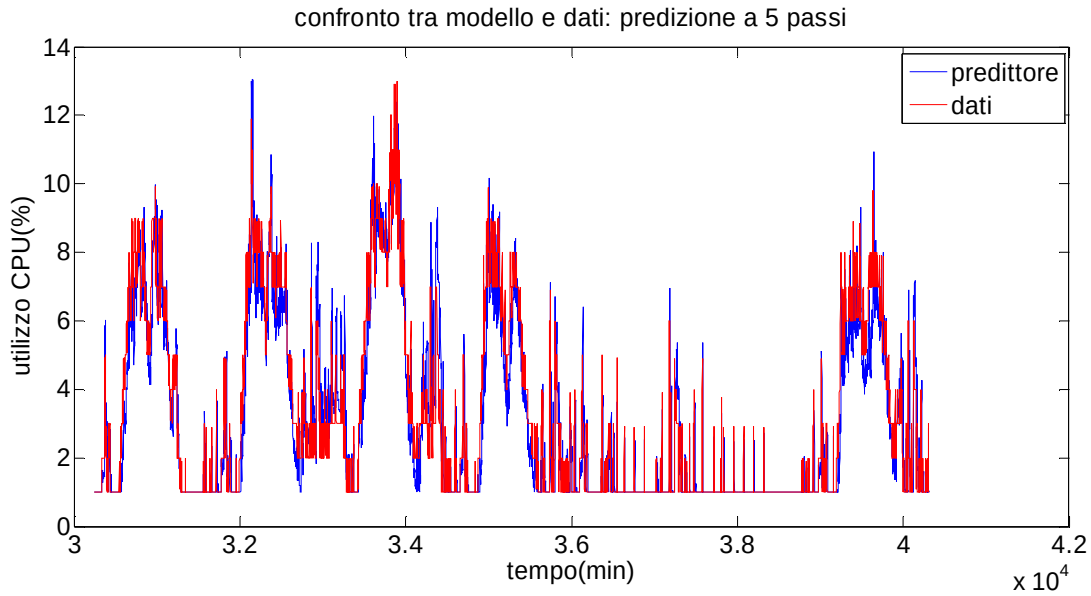


Figura 3.43: confronto tra il segnale totale di CPU e l'output del predittore a 5 passi con saturazione

Come si può vedere dai risultati in tabella e dai confronti grafici, l'andamento della CPU predetta approssima in buona misura i dati reali; per orizzonti temporali entro i 10 passi la predizione si mantiene entro un errore inferiore al 20%. Ciò ci porta a considerare il modello ARIMA un buon metodo per descrivere le dinamiche di utilizzo della CPU, posto che la predizione rimanga sotto un orizzonte temporale inferiore ai 10-15 passi.

Come nel caso della predizione del traffico, riportiamo qui il confronto con la predizione del modello ARMA all'aumentare dell'orizzonte temporale.

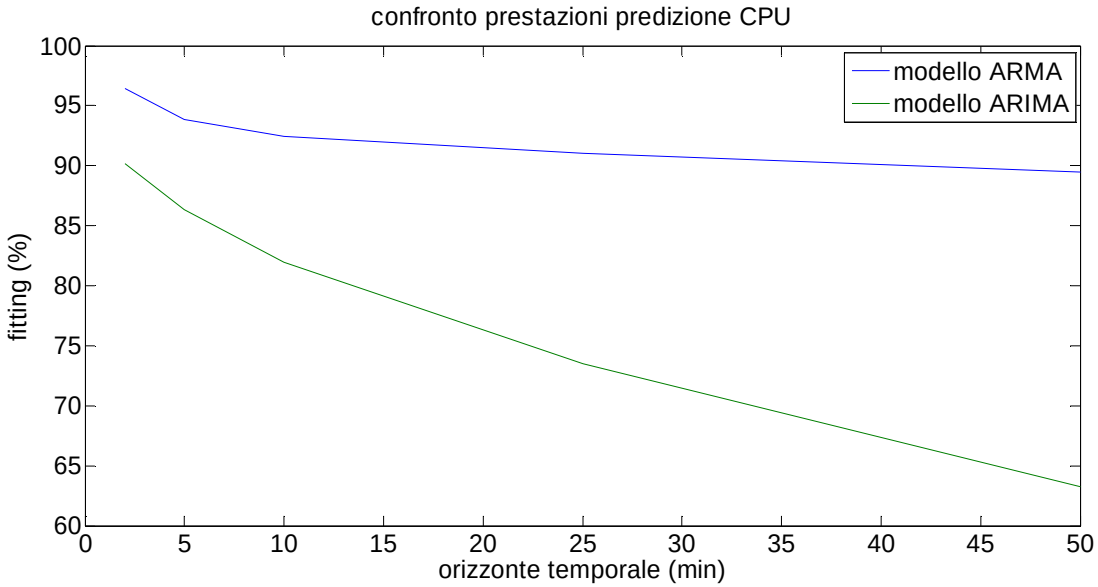


Figura 3.44: confronto dell'andamento del fitting (CPU) all'aumentare dell'orizzonte predittivo per i modelli ARMA e ARIMA

Coerentemente con il grafico 3.41, il modello ARIMA degrada all'aumentare dell'orizzonte predittivo in modo molto più rapido rispetto al modello ARMA. Un ulteriore motivo, seppur

secondario, che porta ad avere una predizione meno performante, è il fatto che l'utilizzo di un solo coefficiente per legare traffico ad utilizzo di CPU denota prestazioni leggermente inferiori rispetto all'uso di 3 coefficienti distinti.

3.2.6 Conclusioni relative al modello ARIMA

A termine della trattazione per questo modello, si può concludere che:

- un modello ARIMA è in grado di modellizzare efficacemente l'andamento del segnale di traffico, riuscendo ad approssimarne in maniera soddisfacente le dinamiche.
- L'ordine del modello identificato risulta basso, permettendo di avere un modello relativamente semplice, pur avendo prestazioni simili a modello con ordine superiore.
- La predizione del traffico mostra valori soddisfacenti per i primi passi di predizione, degradando successivamente all'aumentare dell'orizzonte predittivo; tuttavia, limitando la predizione ad i primi 10 passi (considerando un orizzonte di tipo short-range) si può comunque considerare accettabile la bontà della predizione.
- La conversione da traffico a CPU è effettuata attraverso un unico coefficiente che mette in relazione i segnali totali (come mostrato nel capitolo 2): l'andamento delle prestazioni risulta simile a quelle relative alla predizione del traffico, anche se si attesta su percentuali superiori, portando a prestazioni migliori.

Confrontando le prestazioni ottenute con il modello ARIMA con quelle relative al modello ARMA, si può concludere che in generale le performance del secondo risultano migliori all'aumentare dell'orizzonte predittivo, anche grazie al fatto di considerare 3 coefficienti per il passaggio da traffico e CPU. Per quanto riguarda invece i primissimi passi di predizione le prestazioni risultano circa simili. Ulteriore svantaggio del modello ARIMA sta nel fatto che richiede la taratura di un parametro aggiuntivo (il numero di differenziazioni da effettuare) e che in letteratura non sono stati rilevati metodi per ricavare in modo rigoroso tale parametro.

Vantaggio fondamentale del modello risulta invece essere il fatto che non si ha bisogno di identificare una stagionalità del segnale; ciò comporta una riduzione del tempo di attesa prima di poter adoperare il modello: in questa trattazione si sono utilizzate per tutti i modelli proposti un numero di settimane uguale sia per l'identificazione per che per la validazione; idealmente si richiederebbe invece un numero di settimane superiore per poter identificare con buona approssimazione la stagionalità. In situazioni anomale in cui il traffico presenta picchi in momenti inaspettati (notti o week end per esempio) dovuti a fattori esterni (come interventi di aggiornamento della rete) usare solamente 3 settimane per determinare la stagionalità potrebbe portare all'identificazione di una componente stagionale poco fedele. Nel router utilizzato tale anomalie non si verificano, portando all'identificazione di una stagionalità soddisfacente ma, in generale, ciò non può essere garantito.

Per questo, dovendo scegliere tra i 2 modelli, la scelta dipenderebbe dal numero di dati a disposizione per l'identificazione: avendo un numero di settimane limitato conviene utilizzare un modello ARIMA, all'aumentare dei dati disponibili potrebbe essere più vantaggioso identificare un modello ARMA, che vada dapprima ad affiancare e poi a sostituire il modello ARIMA.

3.3 **Metodo di Holt-Winters**

Si propone ora un terzo modello, che utilizza un approccio differente da quelli precedentemente illustrati, e che viene definito modello di Holt-Winters. [21][22]

In letteratura è possibile trovarne degli impieghi in campi quale quello economico, per la previsione di andamenti di rendite finanziarie, titoli e simili; ma non è stato rilevato un suo utilizzo nel campo delle telecomunicazioni.

I segnali utilizzati dal metodo sono quelli complessivi (quindi il traffico u e l'utilizzo di CPU y), che vengono suddivisi automaticamente dal modello in 3 componenti, da non confondere con quelle utilizzate precedentemente nel modello ARMA, in quanto ciascuna di quest'ultime rappresenta lo stesso contributo delle corrispondente grandezza nel modello di Holt-Winters, ma è definita in modo diverso.

Il modello di Holt-Winters è in grado di prevedere l'andamento di serie temporali caratterizzate da trend e stagionalità, ed è un'evoluzione ricavata a partire da un modello più semplice, detto lisciamento esponenziale, che tratta serie temporali sprovviste di queste due componenti. Per una trattazione teorica completa, viene descritto in primo luogo il metodo del lisciamento esponenziale, procedendo poi a generalizzare tale modello nel caso in cui siano presenti trend (modello di Holt) e trend e stagionalità (modello di Holt-Winters).

Ottenuto il modello, viene ricavata in un primo momento la predizione per il traffico, procedendo poi a ricavare la predizione per l'utilizzo di CPU attraverso due diversi approcci che identificano rispettivamente uno e tre coefficienti per la conversione traffico-CPU (considerando le componenti del modello di Holt-Winters), dal momento che il secondo porta ad avere prestazioni superiori, ma non risulta applicabile per tutti i set di dati dei diversi router considerati. All'interno della trattazione verrà approfondito il motivo di tale affermazione.

3.3.1 Lisciamento esponenziale

Il lisciamento esponenziale è un semplice metodo impiegato quando l'obiettivo è predire i valori futuri di una serie temporale caratterizzata dall'assenza di trend e stagionalità.

Esso può essere diviso in 2 step consecutivi: nel primo, a partire dalla serie temporale presa in esame, si ricava una serie "lisciata", che viene aggiornata istante per istante, per ogni nuovo campione del segnale originale. Nel secondo, a partire dalla serie lisciata, si ricava la predizione per il segnale originale per il numero di passi desiderato.

La serie lisciata è interpretabile come un filtraggio della serie originale dei dati; uno dei motivi per cui si ricorre ad essa sta nel fatto che, in molti casi applicativi, risulta più utile lavorare con un valore filtrato piuttosto che con i dati grezzi, in modo da eliminare eventuali outliers e anomalie in generale.

Il filtraggio (o lisciatura, per usare il linguaggio dettato dal metodo) viene ottenuto calcolando ad ogni istante la media pesata di tutte le osservazioni, a partire dalla prima a disposizione fino all'ultimo istante a disposizione (quindi fino all'istante t, che rappresenta di volta in volta il presente): la serie lisciata sarà tanto più simile ai dati (e quindi meno filtrata) quanto più peso viene dato all'osservazione presente e meno a quelle passate; perciò per lisciare considerevolmente la serie l'osservazione al tempo t dovrà avere minor peso.

Il valore al tempo *t* della serie lisciata è calcolato come:

$$\tilde{u}(t) = c_0 u(t) + c_1 u(t-1) + c_2 u(t-2) + \cdots + c_{t-1} u(1)$$

In cui ognuno dei pesi c_j dipende da un unico parametro λ, detto costante di lisciamento, nella forma:

$$c_j = \lambda(1-\lambda)^j$$

In cui il parametro λ è compreso tra 0 ed 1.

La somma pesata può essere quindi riscritta in una forma più compatta come:

$$\tilde{u}(t) = \lambda \sum_{j=0}^{t-1} (1-\lambda)^j \, u(t-j)$$

La serie lisciata viene detta esponenziale poiché tale è il peso che si dà a ciascuna osservazione: più ci si allontana dal presente più è alto il valore dell'esponente j, rendendo quindi meno influente il contributo del valore all'istante $t-j$ dal momento che, essendo il parametro λ compreso tra zero e uno, la base dell'esponenziale risulta minore di uno.

A partire da questa formulazione se ne può ricavare una variante in forma ricorsiva che esemplifica il rapporto tra la serie lisciata e quella originale e che risulta più vantaggiosa dal punto di vista computazionale:

$$\tilde{u}(t) = \lambda \sum_{j=0}^{t-1} (1-\lambda)^j \, u(t-j)$$

$$= \lambda u(t) + \lambda \sum_{j=1}^{t-1} (1-\lambda)^j \, u(t-j)$$

$$= \lambda u(t) + (1-\lambda)\lambda \sum_{j=1}^{t-1} (1-\lambda)^{j-1} \, u(t-j)$$

$$= \lambda u(t) + (1-\lambda)\lambda \sum_{j=0}^{t-2} (1-\lambda)^j \, u(t-j-1)$$

$$= \lambda u(t) + (1-\lambda)\tilde{u}(t-1)$$

$$\tilde{u}(t) = \lambda u(t) + (1-\lambda)\tilde{u}(t-1).$$

Attraverso tale formulazione si può notare come per valori di λ vicini a 1 la serie lisciata risulti simile alla serie originale di partenza, mentre per valori prossimi a 0 la serie tenda a rimanere vicino ai valori precedentemente assunti con basse variazioni dovute ai dati, per questo la serie viene detta lisciata.

Considerando i due possibili casi estremi quindi, se $\lambda = 1$ ogni valore $\tilde{u}(j)$ è identico al corrispondente valore $u(j)$ (non lisciando perciò il segnale), mentre nel caso in cui $\lambda = 0$ la serie lisciata rimane costante e coincide in tutti gli istanti di tempo al valore iniziale.

In figura 3.45 e 3.46 viene mostrata la differenza tra due segnali generici lisciati con diversi valori di λ.

Si può notare come il lisciamento causato da un basso valore di λ equivalga a filtrare il segnale.

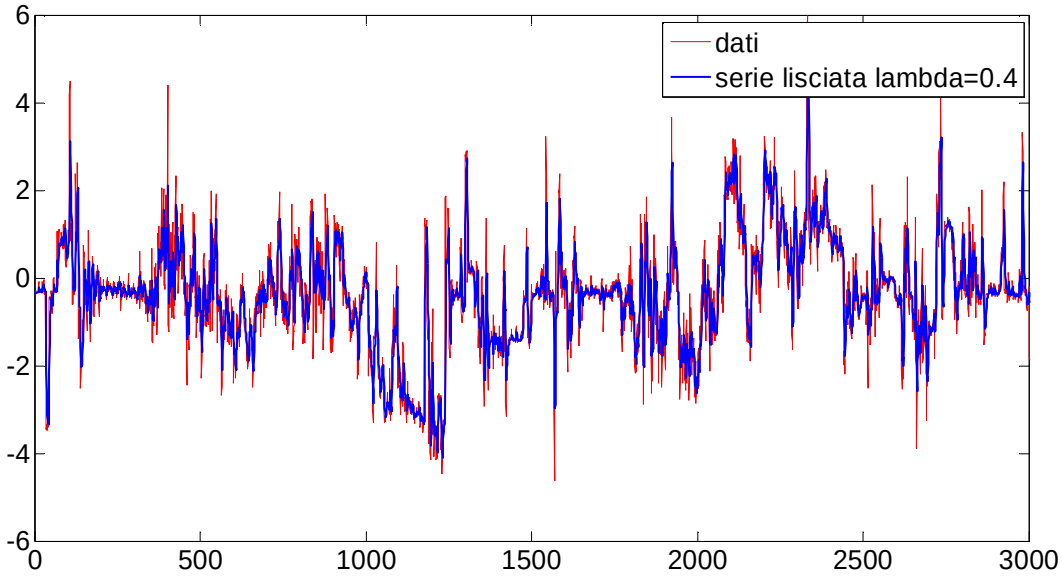


Figura 3.45: confronto tra una serie temporale e il corrispondente lisciamento per λ_1

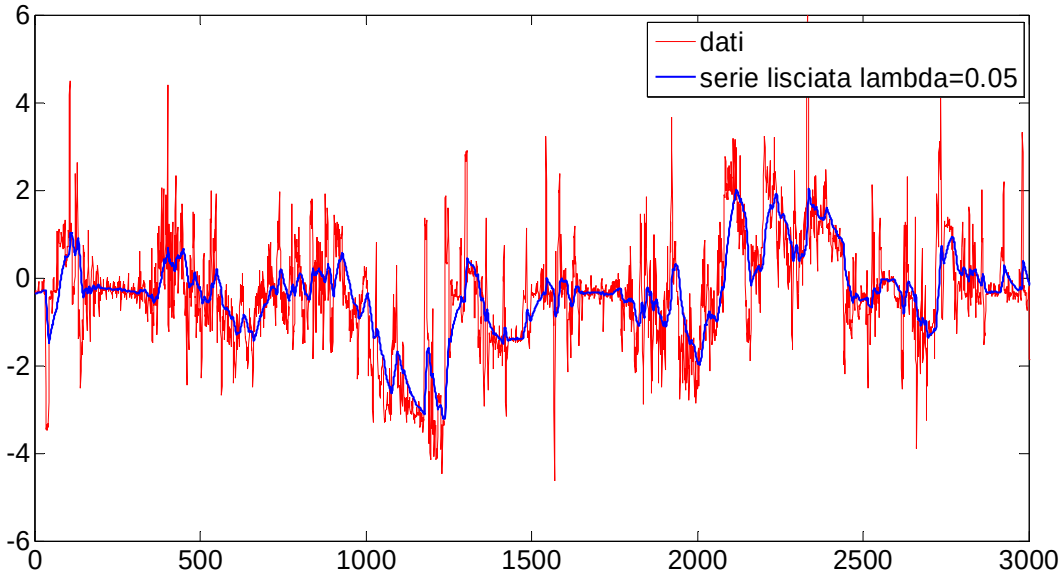


Figura 3.46: confronto tra una serie temporale e il corrispondente lisciamento per λ_1

A partire dalla serie lisciata si ricava successivamente il predittore della serie originale. Il metodo del lisciamento esponenziale assume che la miglior predizione ad un passo corrisponda all'ultimo valore della serie lisciata [21]:

$$\hat{u}(t+1|t) = \tilde{u}(t)$$

è quindi possibile, come per la serie lisciata, calcolare il valore del predittore in modo ricorsivo, andando a sostituire nel predittore il valore della serie lisciata al tempo t:

$$\hat{u}(t+1|t) = \lambda u(t) + (1-\lambda)\hat{u}(t|t-1)$$

Volendo ricavare i predittori per più passi, ad esempio per due, si procede nel seguente modo: a partire dall'equazione precedente, si trasla in avanti di un passo l'istante temporale della predizione:

$$\hat{u}(t+2|t+1) = \lambda u(t+1) + (1-\lambda)\hat{u}(t+1|t)$$

Nella nuova equazione il primo termine a destra dell'uguale indica un dato ad un istante futuro, che quindi non è possibile avere all'istante t; tale valore viene quindi sostituito dal predittore ad un passo, perciò ora il predittore dell'istante $t+2$ dipende solamente da valori delle serie u non successivi all'istante t.

L'equazione diventa:

$$\hat{u}(t+2|t) = \lambda\hat{u}(t+1|t) + (1-\lambda)\hat{u}(t+1|t)$$

Espandendo l'ultimo termine e semplificando si ottiene:

$$\hat{u}(t+2|t) = \hat{u}(t+1|t)$$

Tale risultato mostra come la predizione per un qualsiasi orizzonte temporale sia sempre pari al valore dato dal predittore ad un passo, che a sua volta è uguale all'ultimo valore della serie lisciata $\tilde{u}(t)$. Perciò:

$$\hat{u}(t+h|t) = \tilde{u}(t)$$

Grazie a questo risultato il predittore, indipendentemente dal numero dei passi scelto, è sempre disponibile non appena si è calcolato l'ultimo valore della serie lisciata. Da ciò si capisce come sia fondamentale scegliere con cura il parametro λ, poiché in base ad esso varierà la bontà della predizione. In generale, all'aumentare dell'orizzonte predittivo voluto occorre diminuire il valore di λ utilizzato, mentre se si ricerca una predizione di pochi passi conviene considerare un valore di λ più vicino a 1. Si ricorre ad un esempio per rendere più chiara l'affermazione.

Consideriamo di avere il segnale $u(t)$ mostrato in figura 3.47 e di voler prevederne i valori futuri a partire dall'istante 225. Se l'ultimo istante disponibile fosse ad esempio quello a t=225 e si volesse prevedere pochi passi in avanti (ad esempio 2) converrebbe utilizzare un valore di λ vicino all'unità: questo poiché in tal modo la serie lisciata risulterebbe poco filtrata e perciò vicino al valore assunto dalla serie originale di partenza; se, invece, si filtrasse molto (λ vicino allo zero) il valore della serie lisciata risulterebbe lontano rispetto a quello della serie $u(t)$.

Al contrario, se si volesse prevedere su un lungo orizzonte temporale (partendo sempre dall'istante 225) avere un valore di λ circa pari a 1 porterebbe ad una previsione quasi sempre maggiore in modulo rispetto al segnale $u(t)$, mentre usare un valore di λ basso porterebbe a predire valori nell'intorno della media, generando un errore di predizione minore. Da ciò si può vedere come si possa avere, almeno qualitativamente, un'indicazione sul valore da assegnare al parametro λ.

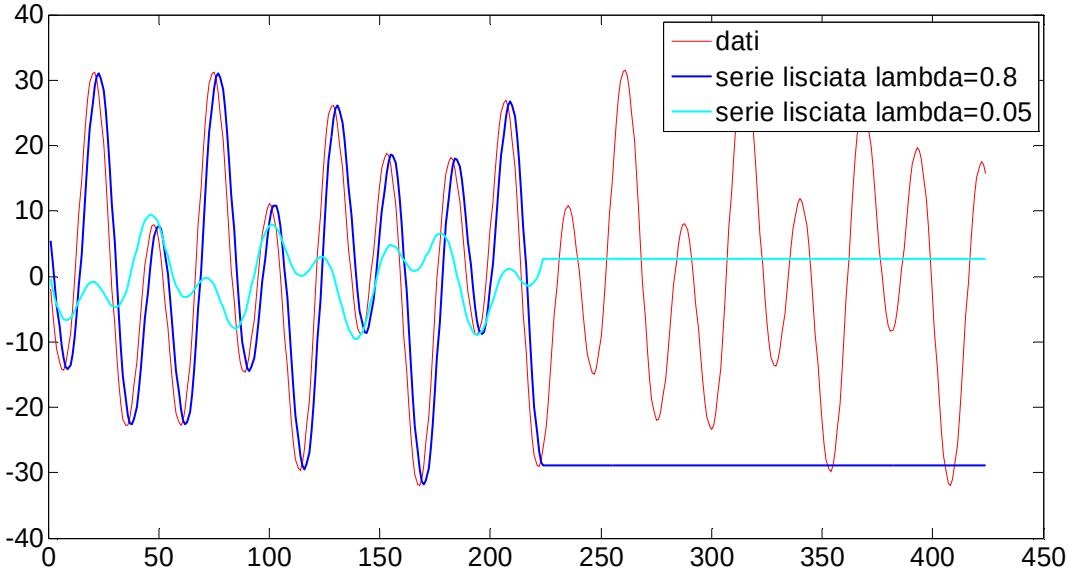


Figura 3.47: confronto tra previsioni ottenute utilizzando diversi valori di λ

Infine, poiché la serie lisciata è ottenibile ricorsivamente, è necessario conoscerne il valore iniziale; esso può essere scelto in due modi:

$$\begin{cases} \tilde{u}(1) = u(1) & \text{se } \lambda \text{ è vicino ad 1} \\ \tilde{u}(m) = \frac{1}{m}\sum_{t=1}^{m} u(t) & \text{se } \lambda \text{ è vicino a 0} \end{cases}$$

Nel caso in cui λ sia circa pari ad 1, conviene inizializzare la serie lisciata al primo valore della serie temporale, dal momento che le due serie presenteranno valori abbastanza simili per tutti gli istanti t.

Viceversa, se λ è nell'intorno dello 0, l'inizializzazione non può essere posta uguale ad un valore della serie $u(t)$, dal momento che, essendo preponderanti in questo caso i valori passati della serie nella determinazione del valore $\tilde{u}(t)$, se $u(1)$ fosse un valore molto lontano dalla media, andrebbe a inficiare tutti i valori via via assunti dalla serie lisciata. Per questo motivo si inizializza la serie lisciata solo dopo m istanti, imponendo che essa non abbia un valore per i primi m istanti in cui si raccolgono i valori della serie $u(1)$, e inizializzando $\tilde{u}(m)$ con la media dei valori raccolti fino a quell'istante.

3.3.2 Metodo di Holt

Considerando serie temporali che presentano trend, il lisciamento esponenziale risulta essere un metodo di predizione inadeguato.

Seguendo infatti quanto mostrato precedentemente, la predizione del modello coincide con l'ultimo valore della serie lisciata indipendentemente dall'orizzonte temporale considerato. Ciò porta a predizioni errate, in quanto, a fronte di un aumento (o diminuzione) dei valori della serie temporale presa in esame dovuto al contributo del trend, la predizione si mantiene costante, ignorando di fatto tale contributo, essendo stato sviluppato il modello ipotizzando tale componente nulla.

Per poter utilizzare l'idea alla base del modello è quindi necessario apportare una modifica che tenga conto della componente aggiuntiva del modello. Il nuovo metodo viene denominato metodo di Holt: esso tiene conto del trend della serie aggiungendo una seconda equazione ricorsiva che permette di mantenere aggiornato il valore della pendenza del trend e che sfrutta per far ciò un secondo parametro λ_2.

La nuova formulazione risulta essere:

$$\begin{cases} \tilde{u}(t) = \lambda_1 u(t) + (1-\lambda_1)\big(\tilde{u}(t-1) + F(t-1)\big) \\ F(t) = \lambda_2\big(\tilde{u}(t) - \tilde{u}(t-1)\big) + (1-\lambda_2)F(t-1) \end{cases}$$

La prima equazione è ottenuta a partire dal lisciamento esponenziale: la differenza sta nel fatto che, nel secondo termine a destra dell'uguale, non si considera più solamente la serie lisciata al passo precedente ($\tilde{u}(t-1)$) ma ad esso viene sommato il contributo dato dal trend. Questo perché si tiene conto del fatto che rispetto all'istante $t-1$ ci deve essere stata una variazione dovuta alla componente di trend.

La seconda equazione aggiunta dal modello serve invece a modellizzare l'andamento del valore di trend: per far ciò, si tiene conto del valore di trend all'ultimo istante (ricavato come differenza tra i valori della serie lisciata al tempo t e $t-1$) e del precedente valore che si era ricavato al tempo $t-1$. Per quanto riguarda i 2 parametri del modello, λ_1 ha la stessa funzione che svolgeva nel lisciamento esponenziale, andando a determinare la lisciatura della serie $\tilde{u}(t)$; λ_2 svolge un compito analogo per la componente di trend: essendo compreso anch'esso tra zero e uno, per valori prossimi allo zero si dà maggior rilevanza ai valori passati della serie $F(t)$, viceversa per valori nell'intorno di uno si dà più peso all'ultimo valore ottenuto. In questo modo, nel primo caso si sarà più soggetti a eventuali picchi presenti nella serie $u(t)$, nel secondo, essendo il segnale filtrato, se ne risentirà meno.

Una precisazione da fare riguarda il significato associato alla componente di trend: essa è vista come il coefficiente angolare del trend che influenza il segnale. Ciò potrebbe portare a fraintendimenti se si osservassero le equazioni del modello: ad esempio, nella prima equazione, $F(t-1)$ va sommarsi a $\tilde{u}(t-1)$, portando quindi a credere che debba avere la stessa unità di misura della serie lisciata $\tilde{u}(t)$. In realtà, considerando un $F(t-h)$ ad un generico passo h, si vedrebbe come alla serie lisciata andrebbe a sommarsi $F(t-h)$ moltiplicato per il numero di passi h: moltiplicando un coefficiente angolare per un intervallo di tempo, otteniamo un valore

sommabile alla serie lisciata $\tilde{u}(t)$. Nell'equazione, essendo h pari a uno, ci si deve ricordare che la moltiplicazione per il numero di passi è implicita, evitando possibili ambiguità.

Ottenute le 2 serie lisciate $\tilde{u}(t)$ e $F(t)$, si può ora passare a predire l'andamento futuro della serie $u(t)$: in questo caso si assume come migliore predizione possibile l'ultimo valore della serie lisciata più l'ultimo valore calcolato di trend moltiplicato per l'orizzonte predittivo:

$$\hat{u}(t + h|t) = \tilde{u}(t) + h \cdot F(t)$$

Infine, anche in questo caso le formule per calcolare $\tilde{u}(t)$ e $F(t)$ sono ricorsive, perciò se ne devono ricavare i valori iniziali.

Per calcolare $\tilde{u}(1)$ e $F(1)$ viene sfruttato un periodo iniziale di raccolta dati di m campioni; attraverso il metodo dei minimi quadrati si ricava la retta che meglio approssima il trend della serie $u(t)$ ottenendo due valori $\hat{\alpha}_0$ e $\hat{\beta}_0$: il primo rappresenta l'intercetta della retta, il secondo il coefficiente angolare.

Fatto ciò, si pone:

$$\begin{cases} \tilde{u}(1) = \hat{\alpha}_0 + \hat{\beta}_0 \cdot m \\ F(1) = \hat{\beta}_0 \end{cases}$$

3.3.3 Metodo di Holt-Winters

In presenza di stagionalità non è possibile utilizzare i metodi fin qui mostrati, poiché la previsione data dal modello di Holt, composta solamente da trend e serie lisciata, non tiene conto dell'andamento periodico dovuto alla stagionalità del segnale che considereremo d'ora in avanti.

Per risolvere tale problema, il modello di Holt può essere ulteriormente generalizzato, introducendo una terza equazione che descriva la componente stagionale delle serie temporale $u(t)$. Tale modifica è stata introdotta da Winters, perciò il nuovo modello considerato prende il nome di metodo di Holt-Winters.

In esso la serie lisciata viene vista come la somma di due componenti:

$$\tilde{u}(t) = L(t) + S(t)$$

In cui $L(t)$ corrisponde alla serie lisciata che si avrebbe nel modello di Holt, cioè senza stagionalità, mentre $S(t)$ rappresenta l'andamento periodico.

Le equazioni che compongono il modello salgono a tre:

$$L(t) = \lambda_1\big(u(t) - S(t-s)\big) + (1-\lambda_1)\big(L(t-1) + F(t-1)\big)$$

$$F(t) = \lambda_2\big(L(t) - L(t-1)\big) + (1-\lambda_2)F(t-1)$$

$$S(t) = \lambda_3\big(u(t) - L(t)\big) + (1-\lambda_3)S(t-s)$$

In cui s è pari alla periodicità della componente stagionale.

Rispetto al modello di Holt, nelle prime due equazioni non compare più la serie lisciata $\tilde{u}(t)$ ma la sua componente $L(t)$; a partire da essa, avendo anche la componente $S(t)$, si può ottenere immediatamente la serie lisciata. Inoltre, dal momento che la prima equazione serve per ottenere di volta in volta i valori di $L(t)$, il primo termine a destra dell'uguale non coincide più con la serie temporale che si vuol predire $u(t)$, ma alla sua differenza con la componente stagionale.

L'aggiunta della terza equazione permette di modellizzare istante per istante la stagionalità: essa viene ottenuta attraverso la somma pesata della differenza tra le serie temporali $u(t)$ e $L(t)$ e il valore di $S(t-s)$, pari cioè al valore di stagionalità ricavato al periodo precedente.

I pesi che vengono assegnati ai 2 contributi sono ottenuti grazie all'utilizzo del terzo parametro λ_3: essendo anch'esso compreso tra zero e uno, nel caso in cui esso sia circa nullo, la serie $S(t)$ tende a rimare circa uguale ai valori con cui è stata inizializzata, viceversa, per valori prossimi all'unità, si considera preponderante il valore ricavato all'ultima iterazione, aggiornando costantemente i suoi valori. Da ciò si può capire come, per segnali che presentano stagionalità ben definite nel tempo, convenga porre il valore di λ_3 a valori bassi; al contrario, nel caso in cui si ritenga la stagionalità assegnata inizialmente non sufficientemente fedele all'originale (avendo avuto pochi campioni a disposizione per ricavarla, ad esempio) conviene porre il parametro λ_3 uguale a valori nell'intorno di uno.

Il predittore ad $h$passi viene ottenuto nel seguente modo:

$$\hat{u}(t+h|t) = L(t) + h \cdot F(t) + S(t+h-qs), \quad \text{con } q = \left[\frac{h}{s}\right]$$

In cui il valore predetto è formato da tre contributi: i primi due coincidono con quelli del modello di Holt, mentre il terzo serve per tener in considerazione la stagionalità. L'operatore $[Z]$ indica il più piccolo numero intero maggiore o uguale a Z.

Nel caso in cui $h \leq s$ il termine q vale sempre 0; nel caso in cui invece risulta $h > s$, il termine q serve a far in modo che non si richiedano valori di $S(t)$ precedenti più di s istanti passati nel tempo, che non sarebbero disponibili.

Infine, si deve considerare l'inizializzazione delle 3 serie temporali $L(t)$, $F(t)$ e $S(t)$. Per le prime 2 si utilizza la stessa procedura mostrata nel modello di Holt; per quanto riguarda la componente stagionale, invece, non è sufficiente inizializzare un solo valore, dal momento che per le prime s iterazioni si avrà bisogno, all'interno delle equazioni, di valori precedentemente assegnati. Per questo si ricava una stagionalità iniziale per un numero di valori pari ad s, utilizzando m (multiplo di s) campioni e ricavando $S(t)$ come la media di quest'ultimi. Compatibilmente con il numero di dati che si può dedicare all'inizializzazione, maggiore è la quantità dedicata, maggiore sarà la sua precisione.

$$S(t)_{1 \leq t \leq s} = \frac{u(t) + u(t-s) + u(t-2s) + \cdots + u(t-m)}{m/s}$$

3.3.4 Taratura dei parametri $\lambda_1, \lambda_2, \lambda_3$

Terminata la trattazione teorica sul modello di Holt-Winters, si passa ora illustrare l'utilizzo che se ne fa all'interno di questa tesi.

Per prima cosa si procede alla taratura dei parametri del modello. Come già detto nel paragrafo 3.3.1, all'aumentare dell'orizzonte predittivo che si considera, i valori di λ_i varieranno, in particolare quelli del coefficiente λ_1, portando ad un serie sempre più lisciata. Si capisce quindi come non sia possibile effettuare, come nei modelli ARMA e ARIMA, un'unica identificazione e successivamente procedere a ricavare i predittori per i vari orizzonti temporali considerati, ma come risulti più utile ritarare i parametri ogni volta che si voglia cambiare l'orizzonte predittivo h.

Per ricavare i valori esatti di $\lambda_1, \lambda_2, \lambda_3$ ci si affida anche in questo caso alla minimizzazione di una cifra di merito. A differenza dei modelli ARMA e ARIMA, in cui si confrontava l'uscita del modello al tempo t con il corrispondente valore della serie temporale $u(t)$, qui si confrontano i valori del predittore ad h passi con il corrispondente valore di $u(t+h)$: se così non si procedesse, minimizzando l'errore tra la serie lisciata e i dati, si otterrebbe un valore di λ_1 pari a 1: ciò porterebbe la serie lisciata a coincidere perfettamente con i valori della serie $u(t)$, di fatto non filtrando, e renderebbe quindi la predizione non ottimale. Si considerano quindi il predittore per diversi orizzonti temporali, e la cifra di merito diventa quindi:

$$errore = \frac{1}{N} \frac{\sum_{t=1}^{N} \left(u(t+h) - \hat{u}(t+h|t)\right)^2}{var(u)} \cdot 100$$

In cui N equivale al numero di dati del set di identificazione.

Ci si aspetta che, minimizzando l'errore di predizione per h crescenti, la lisciatura aumenti progressivamente, poiché la predizione è tanto migliore quanto più l'ultimo valore della serie lisciata si allontana dall'ultimo valore dei dati, per avvicinarsi via via al valor medio.

I dati a disposizione sono gli stessi utilizzati per i modelli ARMA e ARIMA, in modo da poter poi confrontare, a fine capitolo, le prestazioni del modello. Per questo consideriamo, delle 4 settimane disponibili, le prime 3 per poter identificare i valori dei parametri λ_i e la quarta per confrontare i risultati ottenuti dalla predizione con i dati reali. Dal momento che i dati sono campionati con periodo pari a 1 minuto, nella formula dell'errore qui riportata N è pari a 30240 (numero dei minuti totali di 3 settimane).

Per quanto riguarda l'inizializzazione delle serie temporale del modello, dal momento che hanno a disposizione solamente 3 settimane, da usare sia per l'inizializzazione che per l'identificazione, si è ricorso a un artificio: per quanto riguarda la stagionalità, si è considerata la prima settimana e la si è filtrata, in modo da ottenere un segnale meno oscillante di quello originale, e si è assunto che questo segnale fosse circa pari alla stagionalità iniziale, in modo da avere un segnale che, per quanto non coincidente con la vera stagionalità, ne avesse comunque un andamento simile. Procedendo poi ad iterare le equazioni nelle 3 settimane, il modello dovrebbe essere in grado di restituire il valore ottimo di λ_3 tale per cui, a partire dal segnale filtrato con cui è stato inizializzato,

si ottiene l'aggiornamento della stagionalità $S(t)$. Analogamente, anche per il trend si è proceduto con lo stesso metodo, considerando dati per una settimana. Usando tutte le prime 3 settimane, si sarebbero avute inizializzazioni sicuramente più fedeli, ma identiche ai valori di trend e stagionalità ottenute dopo la taratura, per questo i coefficienti λ_2 e λ_3 sarebbero risultati pari a 0, in quanto i valori iniziali sarebbero stati quelli ottimi per la serie considerata, dal momento che porre a 0 tali parametri equivale a non riaggiornare le serie . Considerando invece la sola prima settimana, evitiamo questo problema. Per quanto riguarda il valore di $L(1)$, lo si inizializza come $u(1) - S(1-s)$, entrambi ora noti; infine come detto $F(1)$ viene inizializzato calcolando il trend della prima settimana, utilizzando un numero di dati analogo a quello utilizzato per S.

Per ricavare i valori ottimali dei λ_i, si utilizza uno script Matlab in cui i possibili valori dei parametri variano con un passo di 0.001. Considerata una generica tripla di valori per i 3 λ_i, a partire dalle condizioni iniziali si ricavano, iterando le equazioni, le corrispondenti serie $L(t), F(t)$ e $S(t)$; a partire da queste si ricava il predittore per l'orizzonte temporale via via considerato e si confrontano con i dati del set di identificazione; quindi per ogni orizzonte temporale i valori di λ_i varieranno.

Considerando poi il fatto che il traffico in un router non può mai assumere valori negativi, viene inserita anche in questo modello una saturazione, tale per cui, se l'uscita del modello è negativa, essa viene posta pari a zero, in caso contrario il valore rimane invariato:

$$\hat{u}(t) = \begin{cases} \hat{u}(t) & \text{se } \hat{u}(t) \geq 0 \\ 0 & \text{se } \hat{u}(t) < 0 \end{cases}$$

Si riportano ora i valori dei λ_i ottenuti considerando gli orizzonti temporali pari a $h = 1, h = 2$, $h = 5$ e $h = 10$. Viene considerato inoltre anche un orizzonte temporale molto lungo, pari ad $h = 100$: questo non perché sia di interesse in questo momento predire fino a 100 passi in avanti (essendo noi interessati in questo capitolo a fare previsione short-range, quindi per pochi passi) ma per vedere come e quanto degradano le prestazioni del predittore all'aumentare dell'orizzonte predittivo.

	$h = 1$	$h = 2$	$h = 5$	$h = 10$	$h = 100$
$\boldsymbol{\lambda_1}$	0.882	0.425	0.124	0.063	0.013
$\boldsymbol{\lambda_2}$	0	0	0	0	0
$\boldsymbol{\lambda_3}$	1	0.424	0.312	0.330	0.346

In figura 3.48 l'andamento del predittore a un passo rispetto ai dati per le settimane su cui sono tarate le costanti di lisciamento.

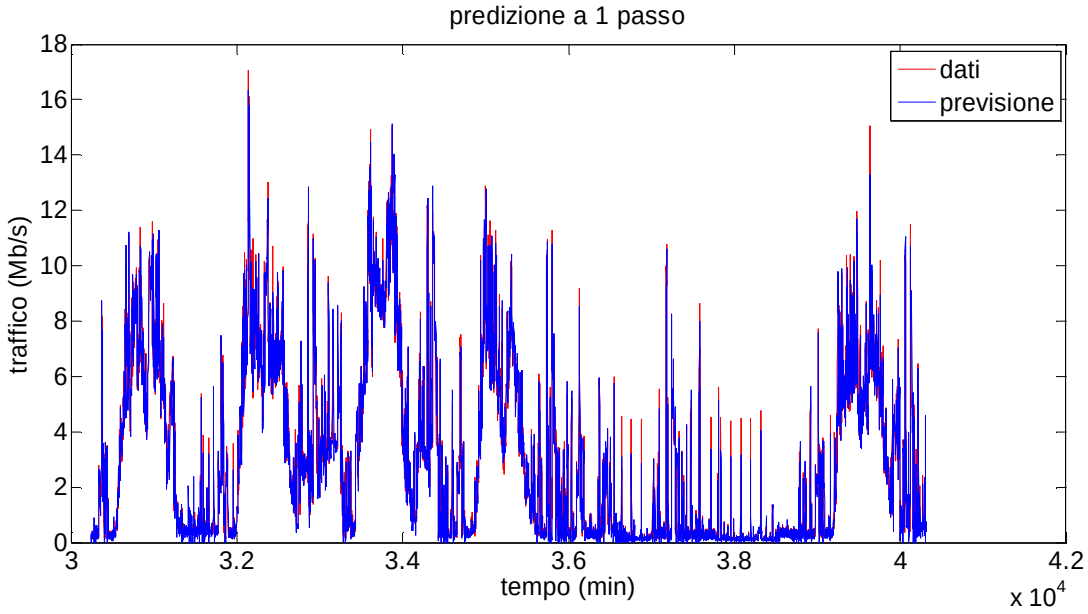


Figura 3.48: confronto tra il segnale totale di traffico e l'output del modello identificato

Si può osservare come i valori di λ_1 diminuiscano all'aumentare dell'orizzonte predittivo: come ci si può aspettare, infatti, la serie viene progressivamente lisciata maggiormente, generando una componente $L(t)$ via via più filtrata.

Per quanto riguarda λ_2 si può osservare come tale parametro sia sempre pari a 0. Ciò è spiegabile considerando l'inizializzazione che si è assegnata ad $F(1)$: questa viene calcolata su un'intera settimana iniziale, dando un valore iniziale di trend pari circa a zero (si ricorda che il trend del segnale è infatti circa nullo). Il modello di Holt-Winters compara questa inizializzazione, istante per istante, alla differenza tra i due valori consecutivi della serie lisciata $\tilde{u}(t)$ e $\tilde{u}(t-1)$. Tale differenza corrisponderà quasi sempre ad un valore che, se preso come nuovo valore di $F(t)$, darebbe valori peggiori rispetto a quello ottenuti mantenendo il valore iniziale.

Ciò risulta chiaro se si pensa al numero di punti considerati per ricavare il valore di inizializzazione (10080), comparato con i 2 punti che si considerano per ricavare il valore con cui aggiornare la componente F. Per avere valori di λ_2 diversi da 0 si dovrebbe assegnare un valore iniziale $F(1)$ errato, ma ciò non porterebbe a un miglioramento delle prestazioni, ma solo a una variazione del parametro.

Infine, considerando λ_3, si può notare come all'aumentare della predizione corrisponda un abbassamento del valore che via via il parametro assume, pur rimanendo sempre sensibilmente diverso da zero. Ciò è corretto dal momento che, andando a 0, si manterrebbe la stagionalità iniziale, ricavata a partire utilizzando unicamente la prima settimana e perciò non precisa.

In più, come il primo parametro λ_1, a mano a mano che si aumenta il numero di passi utilizzato nella cifra di merito minimizzata, il fatto che λ_3 diminuisca comporta che il segnale totale $\tilde{u}(t) =$

$L(t) + S(t)$ sia più lisciato. Ciò è coerente con quanto affermato prima, cioè che la lisciatura aumenta all'aumentare dell'orizzonte predittivo.

Non vengono riportate percentuali relative al fitting del modello in quanto in questo caso il segnale ricavato $\tilde{u}(t)$ non ha come obiettivo l'essere il più possibile uguale al segnale $u(t)$, ma essere lisciato in maniera conveniente per la predizione che ci si propone di fare.

3.3.5 Predizione del traffico in ingresso ad un router

Si passa ora a considerare la quarta settimana, per giudicare la bontà della predizione su un set di dati non usati nel ricavare i parametri λ_i. I parametri del modello rimangono ovviamente quelli appena ricavati. In tabella sono mostrati i risultati:

	$h = 1$	$h = 2$	$h = 5$	$h = 10$	$h = 50$
Fitting (%)	94.33%	89.85%	84.81%	82.36%	76.86%

In figura 3.49 e 3.50 sono visibili gli andamenti dei traffici previsti per $h = 2$ e $h = 10$.

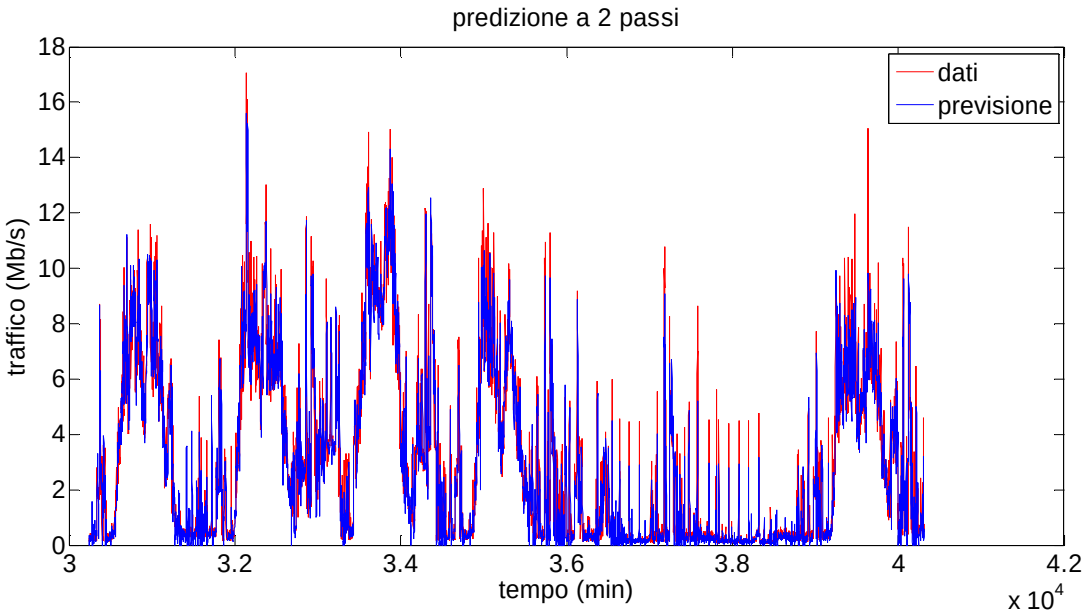


Figura 3.49:confronto tra il segnale totale di traffico e l'output del predittore a 2 passi

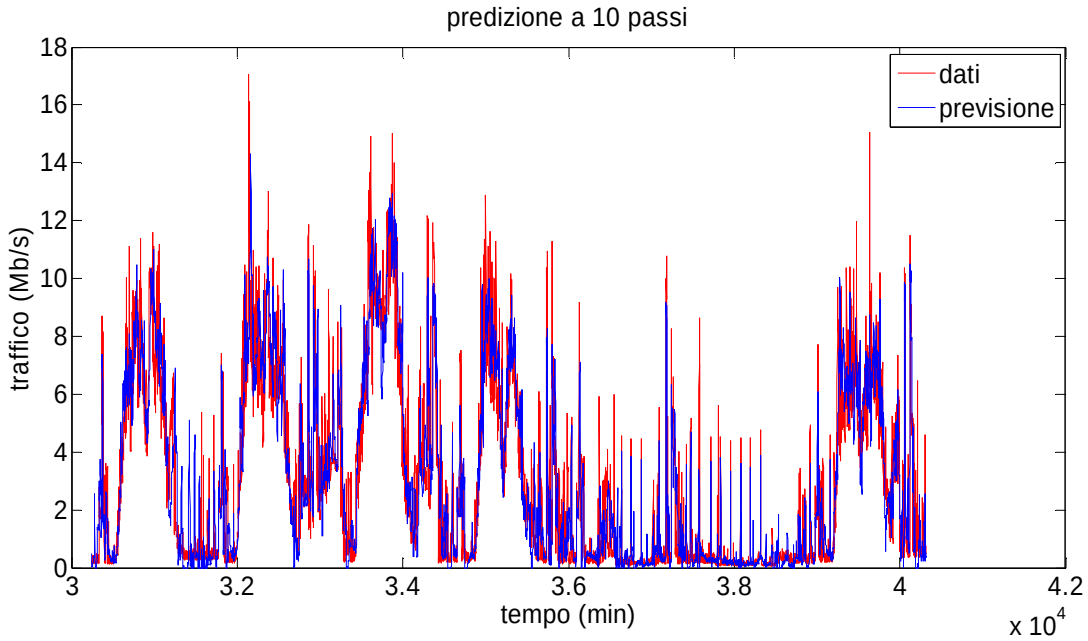


Figura 3.50:confronto tra il segnale totale di traffico e l'output del predittore a 10 passi

Si può notare come la predizione short-range dia risultati soddisfacenti, rimanendo l'errore entro valori inferiori al 20%. Dal confronto grafico tra predittore e dati si nota come gli andamenti siano seguiti con buona approssimazione, andando in molti casi a sovrapporsi.

In figura 3.51 si ha il confronto tra i fitting dei 3 modelli considerati, sempre riferendosi al traffico:

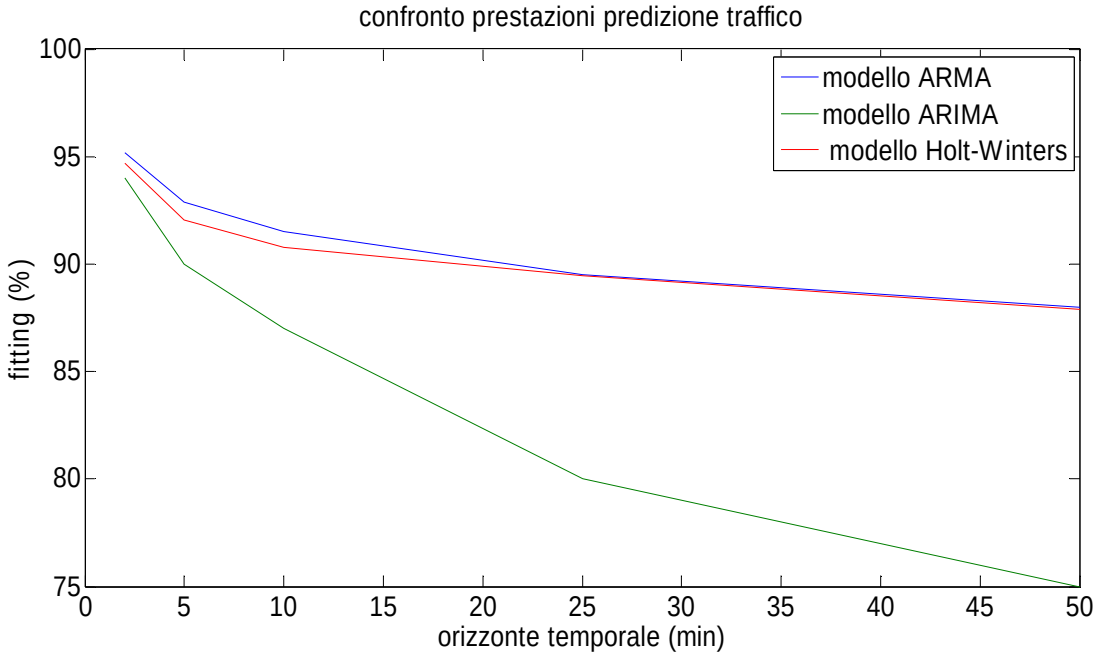


Figura 3.51: confronto dell'andamento del fitting all'aumentare dell'orizzonte predittivo

Si può osservare come, per i primi istanti (da 1 a 6-7 passi) di predizione, il modello ARMA presenti performance leggermente migliori del modello di Holt-Winters, mentre il modello ARIMA risulti inferiore. All'aumentare della orizzonte temporale, invece, la previsione del modello di Holt-Winters va assumere valori simili a quelli ottenuti dal predittore ARMA. Ciò può essere spiegato osservando che nel predittore ARMA le prestazioni degradano all'aumentare dell'orizzonte predittivo in quanto il contributo della componente stocastica tende ad avvicinarsi via via alla media (che è pari a zero), viceversa, il modello di Holt-Winters pone la componente $L(t)$ pari a una costante qualunque sia il valore di h.

Va però ricordato che il modello di Holt è stato inizializzato con una stagionalità ricavata unicamente dalla prima settimana, e quindi sicuramente non ottimale, viceversa, la stagionalità considerata nel modello ARMA è sempre quella ricavata a partire dalle 3 settimane.

3.3.5 Predizione di utilizzo di CPU

Terminata la trattazione per la previsione del traffico, si passa ora alla previsione dell'utilizzo di CPU. Per ottenere quest'ultima, parrebbe sufficiente utilizzare la relazione statica individuata nel capitolo 2. Il modello di Holt-Winters prevede di scomporre il segnale in 3 componenti, ma esse risultano diverse da quelle considerate nel capitolo 2: in questo caso il trend risulta essere un coefficiente angolare; inoltre la stagionalità è ricavata a partire dal segnale $u(t)$, mentre nel capitolo 2 la si otteneva considerando il segnale già depurato da trend.

Si aprono quindi due alternative: la prima, considerata anche nel modello ARIMA, consiste nell'utilizzare il coefficiente che mette in relazione traffico e CPU totali: ricavata la predizione per il traffico mostrata nel paragrafo precedente, è sufficiente moltiplicare per il coefficiente statico ricavato a partire dalle 3 settimane di validazione per ricondursi alla predizione di CPU.

La seconda, più vantaggiosa in termini di prestazioni, prevede di utilizzare, in parallelo al modello di Holt-Winters dedicato al traffico, un secondo modello di Holt-Winters per l'utilizzo di CPU: in questo modo vengono ricavate tre componenti $L(t), S(t)$ e $F(t)$ per entrambi i segnali. Ricavate tali componenti, si può procedere ad identificare i 3 coefficienti che legano le componenti $L(t)$, $F(t)$ e $S(t)$ dei due rispettivi segnali.

Ci si potrebbe domandare perché, una volta determinato un modello di Holt-Winters per il segnale di CPU, non si utilizzi direttamente questo per poter far predizione e non dover più considerare quindi il traffico in ingresso. In primo luogo ciò porterebbe a non descrivere più il legame esistente tra traffico e utilizzo della risorsa; inoltre impiegando dapprima un predittore per il traffico e un ulteriore passaggio per la conversione traffico-CPU si introduce un grado di libertà aggiuntivo, dato dalla relazione tra i due segnali. In questo modo, qualora tale relazione subisse variazioni, ad esempio a causa di un sovraccarico o per aver considerato device diversi da un router, il modello per la predizione di traffico risulterebbe comunque valido, essendo il segnale di traffico in entrata indipendente dal device verso cui è diretto, e si dovrebbe intervenire solo sulla conversione traffico-CPU, non dovendo perciò modificare tutta la trattazione sviluppata.

Da sottolineare come tale metodo sia utilizzabile unicamente solo se i valori dei coefficienti λ_i dei due modelli risultano essere molto simili. In caso contrario non è possibile utilizzare tale metodo, dal momento che le componenti $L(t)$ dei due segnali differiscono: la componente corrispondente a λ_1 minore risulta essere più filtrata, mentre quella corrispondente a $\lambda_1$maggiore presenta andamento meno filtrato.

Nel caso del router Adderley finora considerato i coefficienti risultavano pressoché uguali, perciò si è scelto di utilizzare il secondo approccio descritto. Negli altri router considerati, usati per generalizzare i risultati di questo capitolo, di cui si riportano i risultati in appendice 3.5, tale uguaglianza era assente, comportando l'utilizzo del coefficiente unico.

Il valore del coefficiente K_L^0 che lega le due serie lisciate $L(t)$ è:

$$K_L^0 = \sum_{i=1}^{N} L_{y,i} L_{u,i} \Big/ \sum_{i=1}^{N} {L_{u,i}}^2 = 0{,}589 \text{ [\%/(Mb/s)]}$$

Per quanto riguarda invece il valore di K_F^0:

$$K_F^0 = \sum_{i=1}^{N} F_{y,i} F_{u,i} \Big/ \sum_{i=1}^{N} {F_{u,i}}^2 = 1.066 \text{ [\%/(Mb/s)]}$$

Infine per la componente stagionale del segnale:

$$K_S^0 = \sum_{i=1}^{N} s_{y,i} s_{u,i} \Big/ \sum_{i=1}^{N} {s_{u,i}}^2 = 0{,}955 \text{ [\%/(Mb/s)]}$$

Inoltre anche in questo caso si introduce una saturazione per caratterizzare il fatto che, a causa dei processi interni ad un router, una parte delle risorse rimane sempre allocata:

$$\hat{Y}(t) = \begin{cases} \hat{Y}(t) & \text{se } \hat{Y}(t) \geq 1 \\ 1 & \text{se } \hat{Y}(t) < 1 \end{cases}$$

Si può quindi passare a mostrare i risultati per la predizione, considerando degli orizzonti predittivi pari a quelli usati per la predizione del traffico; in tabella sono mostrate le percentuali di fitting, considerando sempre la medesima cifra di merito.

	h=1	h=2	h=5	h=10	h=50
Fitting (%)	98.07%	96.28%	93.41%	92.24%	89.43%

In figura 3.52 e 3.53 sono presenti i confronti tra i dati e le predizione per 2 e 10 passi:

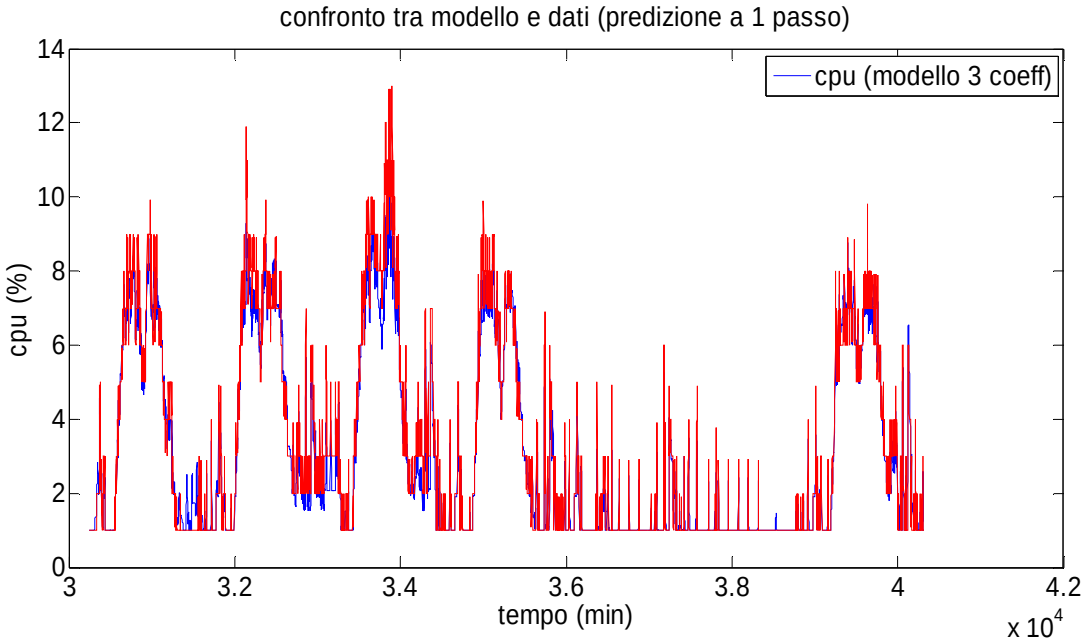


Figura 3.52: confronto tra il segnale totale di CPU e l'output del predittore ad un passo

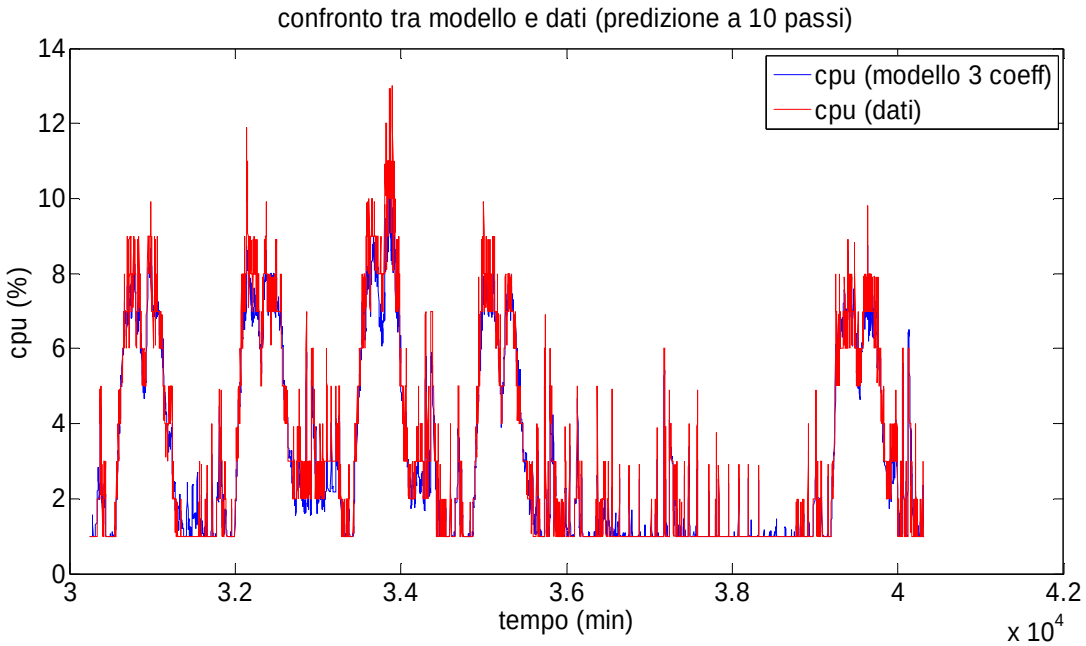


Figura 3.53: confronto tra il segnale totale di CPU e l'output del predittore ad un passo

Dai risultati mostrati in figura così come dai valori riportati in tabella si può concludere che il modello di Holt-Winters è in grado di predire in modo convincente l'andamento di CPU per un orizzonte temporale di tipo short-range. In particolare la decisione di utilizzare un coefficiente dedicato per ogni componente del segnale permette di ottenere risultati migliori rispetto al caso in cui si considerasse unicamente un coefficiente (per eventuali confronti, si rimanda all'appendice 3.5 in cui si considerano altri router interni alla rete Acme).

Da ultimo si confrontano le prestazioni del modello con i risultati ottenuti dai modelli ARMA e ARIMA. In figura 3.54 è visibile l'andamento dei valori di fitting per tutti i 3 modelli considerati:

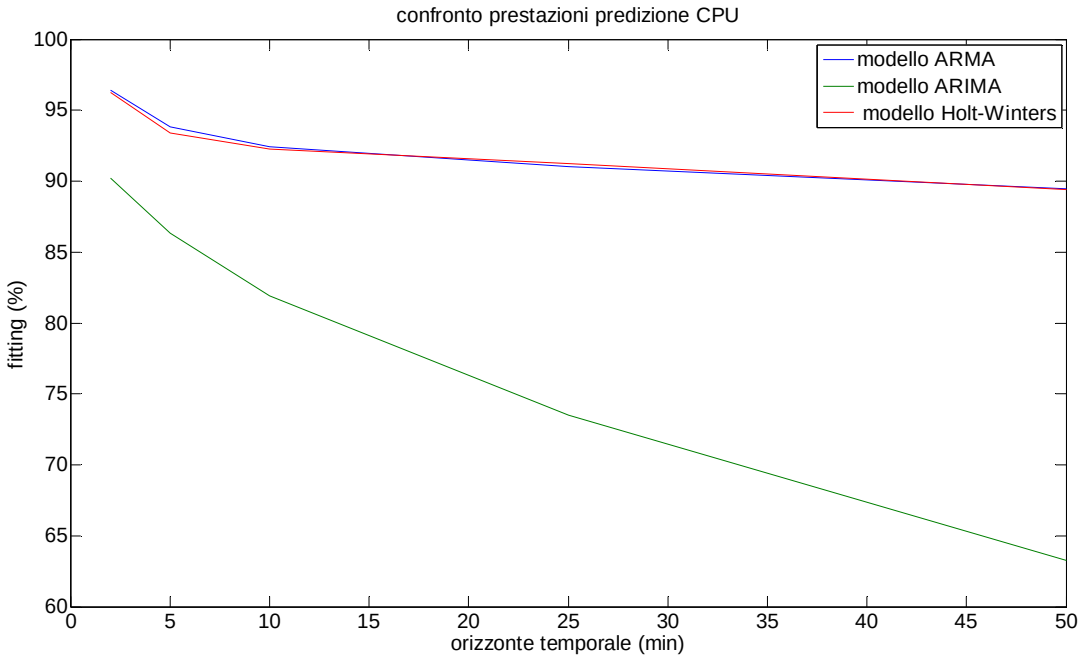


Figura 3.54: confronto dell'andamento del fitting all'aumentare dell'orizzonte predittivo

Nuovamente, il modello di Holt-Winters risulta avere prestazioni praticamente identiche a quelle del modello ARMA, mentre il modello che ne esce perdente è il modello ARIMA.

Ciò può essere spiegato considerando che, oltre ad avere prestazioni inferiori già nella predizione di traffico, il modello ARIMA utilizza un unico coefficiente nella conversione d traffico a CPU. Questo, sommato alle prestazioni già inferiori, accentua ulteriormente la differenza di performance.

Al contrario, avendo identificato 3 coefficienti separati per le componenti proprie del modello, Holt-Winters dà risultati alla pari del modello ARMA. Nel caso in cui non fosse possibile utilizzare tali coefficienti (nel caso in cui i coefficienti λ_i non risultassero uguali, come per i router considerati nell'appendice 3.6) l'utilizzo del coefficiente unico porterebbe a prestazioni simili a quelle del modello ARIMA per i primi passi e migliori di quest'ultimo successivamente (per orizzonti oltre il decimo passo e quindi non di tipo short-range) ma leggermente peggiori del modello ARMA.

Conclusioni relative al modello di Holt-Winters

A temine della trattazione per il modello considerato, si possono trarre le seguenti conclusioni:

- Il metodo di Holt-Winters risulta adatto a prevedere gli andamenti futuri di traffico e CPU a partire dai dati passati.
- A differenza dei modelli precedenti, non viene identificato un modello per caratterizzare il segnale considerato, ma si crea un nuovo segnale (la serie lisciata $\tilde{u}(t)$) che varia a seconda di quale orizzonte temporale si voglia considerare: più grande è l'orizzonte predittivo, maggiore è il filtraggio che il segnale iniziale subisce per poter ottenere la serie $\tilde{u}(t)$.
- La previsione del traffico dà buoni risultati, riuscendo a mantenere ridotto lo scarto tra dati e predizione.
- Anche la predizione per quanto riguarda l'utilizzo di CPU dà risultati soddisfacenti; in particolare nel caso in cui, identificato un modello di Holt-Winters anche per il segnale di CPU, si riscontri un'uguaglianza tra i valori dei coefficienti λ_i dei due segnali; in tal caso è possibile ricavare una relazione tra le componenti del segnale che porti a buona percentuali di fitting del predizione di CPU totale. Qualora ciò non accada, attraverso l'uso di un unico coefficiente è comunque possibile ricondursi ad una predizione di utilizzo di CPU soddisfacente.

CONCLUSIONI

In questo capitolo si è preso in considerazione la predizione dell'utilizzo di CPU a partire dai dati passati del traffico in ingresso ad un router, lungo un orizzonte temporale breve.

Il processo è diviso in diversi passaggi: per prima cosa, a partire dal traffico passato si cerca di individuare un modello in grado di descrivere l'andamento del traffico presente; in secondo luogo si ricava un predittore di traffico per poterne prevedere i valori futuri lungo vari orizzonti temporali; da ultimo, ottenuta la predizione di traffico, si ricava il corrispondente utilizzo di CPU attraverso la relazione statica ricavata all'interno del capitolo 2.

Si sono considerati tre possibili modellazioni per il problema: un modello ARMA, un modello ARIMA ed il metodo di Holt-Winters.

Tutti i tre approcci presentano prestazioni soddisfacenti, mostrando però delle differenze tra loro:

- Il modello ARMA ottiene in generale prestazioni buone, ma va considerato che, per poter essere usato, si devono ricavare la stagionalità ed il trend per tutte le settimane che si vanno a considerare per l'identificazione, obbligando a svolgere un'operazione preliminare sui dati a disposizione. Per la conversione da traffico ad utilizzo di CPU si utilizzano tre coefficienti distinti, uno per ogni componente del segnale.
- Il modello ARIMA, al contrario, presenta prestazioni inferiori, pur rimanendo sufficienti; punto a favore del modello è il fatto che non necessita di una scomposizione del segnale nelle sue componenti e quindi di un'operazione volta a stabilire la stagionalità, ma considera il segnale nella sua totalità, senza aver bisogno di ulteriori informazioni. Uno dei suoi parametri (d, corrispondente al numero di differenziazioni del segnale) deve essere tarato empiricamente ma, nel corso di questa tesi, ciò non ha rappresentato un problema, sapendo che generalmente porre tale parametro ad un valore pari all'unità o a due risulta adeguato, fatto confermato sperimentalmente. Per la conversione del traffico nel corrispondente utilizzo di CPU si utilizza un unico coefficiente.
- Il modello di Holt-Winters, infine, considera come il modello ARIMA il segnale complessivo. Come quest'ultimo non ha bisogno di operazioni preliminari per poter essere utilizzato (il filtraggio della prima settimana è stato utilizzato a causa della scarsità del numero di dati a disposizione, normalmente basterebbe utilizzare una settimana non usata nel processo di identificazione). Le prestazioni di quest'ultimo variano in base a quale coefficiente si considera per la conversione da traffico ad utilizzo di CPU: nel caso sia possibile utilizzare un coefficiente per ciascuna componente dei segnali esse risultano simili a quelle del modello ARMA, al contrario, se ciò non è possibile, si ottengono percentuali di fitting simili a quelle del modello ARIMA (per un orizzonte temporale breve).

Come si può vedere dai risultati nel corso del capitolo, tutti i modelli sono in grado di prevedere in modo soddisfacente l'utilizzo di CPU; a seconda di quali siano le esigenze considerate è preferibile scegliere un modello piuttosto che un altro: se ad esempio non si vogliono fare operazioni preliminari sul segnale, conviene usare un modello ARIMA o di Holt-Winters, se invece si ricerca la precisione massima possibile, conviene utilizzare un modello ARMA, tenendo presente che va ricavata per prima cosa la stagionalità, oppure un modello di Holt-Winters, qualora fosse possibile utilizzare più coefficienti per il passaggio da traffico ad utilizzo di CPU.

4

Predizione long-range di traffico e utilizzo di CPU di un router

Nel corso del precedente capitolo si è discussa la possibilità di prevedere gli andamenti di traffico e CPU per un orizzonte temporale breve, in modo da poter avere uno strumento utile per il problema del load balancing in una server farm.

All'interno di questo capitolo ci si occupa del problema complementare: non si considera infatti un lasso temporale breve, ma una predizione su un orizzonte predittivo più esteso. L'obiettivo è quindi predire con un'accuratezza adeguata il modo in cui evolveranno traffico e CPU di un router nel lungo periodo.

Un'applicazione che necessita particolarmente di tale predizione è il capacity planning: esso consiste nel poter prevedere, con ragionevole anticipo, quali saranno le necessità di una rete telematica in termini di componenti hardware per poter far fronte all'eventuale aumento del traffico circolante. In questo modo si evitano congestionamenti della rete, andando a potenziare i nodi in cui vengono individuati i colli di bottiglia e rimuovendo quindi le cause di possibili prestazioni non soddisfacenti.

In mancanza di un capacity plan, per evitare possibili saturazioni di router e switch della rete, la pratica adottata normalmente consiste nell'aumentare periodicamente le risorse dei device senza tener conto di quale sarà l'andamento futuro del traffico: in questo modo esiste la possibilità che parte delle risorse spese siano male utilizzate, andando a incrementare le prestazioni di router che potrebbero non essere interessati da aumenti del carico di lavoro da svolgere, o non potenziando a sufficienza i router maggiormente stressati della rete.

Si capisce quindi quanto sia importante disporre di previsioni protratte nel tempo, per poter avere uno strumento con cui poter fare capacity planning e poter disporre di tempo a sufficienza per poter effettuare le modifiche necessarie.

Lo studio della predizione long-range, paragonato a quello della predizione di tipo short-range si differenza per:

- Il set di dati è relativo a 16 settimane di misurazioni.
- La granularità dei dati: i segnali vengono campionati con un periodo di 24 minuti poiché si è interessati al loro andamento medio e non al valore puntuale minuto per minuto, ciò porta ad

avere un numero di campioni in un giorno pari a 60, che risulta sufficiente per caratterizzare l'andamento giornaliero dei dati. A differenza del capitolo precedente i dati vengono sottoposti ad un preprocessing in modo da avere la granularità desiderata.

- L'applicazione per la quale viene sviluppata: mentre nel capitolo 3 l'obiettivo era fornire uno strumento per il load balancing e la prevenzione di attacchi informatici, in questo caso l'applicazione possibile è il capacity planning.

All'interno del capitolo vengono utilizzati nuovamente i modelli ARMA, ARIMA e Holt-Winters, procedendo in modo analogo al capitolo precedente:

- Identificazione del modello
- Validazione (inizializzazione e taratura per il metodo di Holt-Winters)
- Predizione per il segnale di traffico a partire dal modello identificato
- Predizione per utilizzo di CPU utilizzando le relazioni statiche e lineari del capitolo 2.

Per concludere si procede con una comparativa dei vari modelli, per poter stabilire quale sia il più adatto allo scopo del capitolo.

4.1 **Dati**

4.1.1 Dati utilizzati

Per prima cosa si considera il set di dati utilizzato in questo capitolo.

In tutti i capitoli precedenti si sono considerati i dati relativi alle 4 settimane di maggio, campionate con una frequenza pari ad un minuto. Tale scelta era giustificata dal fatto che, volendo ottenere una predizione short-range, era necessario avere un campionamento abbastanza fitto del segnale; non avrebbe avuto infatti senso, dal punto di vista del load balancing all'interno di una server farm, campionare con un tempo di campionamento, per esempio, di 10 minuti, dal momento che si era interessati ad una previsione per i minuti immediatamente successivi in modo da poter capire quali, tra le macchine a disposizione, fossero le più sollecitate nell'immediato. Utilizzare un set di dati campionati poco frequentemente nel tempo avrebbe quindi portato a non avere una granularità sufficiente per lo scopo che ci si era prefissati.

In questo capitolo invece ci si pone l'obiettivo di prevedere con ragionevole affidabilità l'andamento futuro di traffico e utilizzo di CPU su un orizzonte predittivo esteso: si vuole cioè prevedere il comportamento dei segnali per più settimane a partire dall'ultimo dato disponibile; da ciò si capisce come non si sia interessati all'andamento puntuale che si può assumere minuto per minuto, ma all'andamento generale assunto dai segnali presi in esame. Non importa quindi se ad un determinato istante ci sia stata un'impennata di traffico della durata di qualche decina di secondi, ma quale sia il comportamento di traffico e CPU per lassi prolungati di tempo.

Lo scopo del capacity planning è capire in quali punti della rete si ha bisogno di potenziare le componenti hardware maggiormente stressate, dal momento che tali device possono diventare dei colli di bottiglia e portare a un rallentamento del traffico lungo tutta una parte o addirittura la totalità della rete. E' noto che per poter mostrare rallentamenti sensibili, un router deve avere valori alti di utilizzo di CPU per un numero prolungato di minuti: se tale circostanza si verifica, il router satura, diminuendo la quantità di traffico che è in grado di processare e spedire sulle proprie interfacce; ciò porta ad un "effetto domino" per cui il router immagazzina sempre più un maggior numero di megabyte da smaltire, innalzando ulteriormente l'utilizzo delle proprie risorse fino a quando il router cessa del tutto di rispondere, ed è considerato essere in uno stato di "down". E' chiaro a questo punto come, in funzione del capacity planning, gli spikes improvvisi di traffico non siano influenti, in quanto portano ad un incremento di utilizzo di risorse solamente puntuale; e si sia invece interessati a trend e stagionalità di traffico e utilizzo di CPU.

Alla luce di quanto detto, si è scelto quindi di utilizzare un set di dati con una granularità differente rispetto a quella dei dati usati fino ad ora. A nostra disposizione per questa tesi vi erano due set di dati: un primo, utilizzato nei capitoli precedenti, composta da 4 settimane campionate con una frequenza pari ad un minuto, ed un secondo, composto da 16 settimane, comprese tra il 7 maggio ed il 29 agosto. Ricordando il codice RRDTool utilizzato per creare i database in cui salvare i dati (presente in appendice 2.1), si può ricavare la granularità del secondo set di dati a disposizione, variabile a seconda di quanto i dati siano recenti: per le 4 settimane dal 2 agosto al 29, la

frequenza di campionamento risulta essere nuovamente pari a 1 sample al minuto; i 3 giorni precedenti si ha un campione ogni 6 minuti; per i dati dal 17 al 31 luglio si ha un campione ogni 24 minuti e per quelli precedenti (a partire dal 7 maggio, giorno in cui si è cominciato a popolare il database) la frequenza è pari ad un sample ogni 144 minuti.

I fattori che influiscono nella scelta del campionamento da utilizzare sono 2: l'avere un numero di punti in grado di caratterizzare fedelmente l'andamento giornaliero del segnale unito alla necessità di disporre di un numero di settimane di dati sufficiente.

Il primo requisito deriva dal fatto che, se si considerassero troppo pochi campioni per giorno, si potrebbero avere dei dati non in grado di descrivere i valori massimi di traffico e CPU raggiunti nelle ore di punta: se ad esempio si considerassero solo 3-4 punti per tutto il giorno (ore diurne), considerando che ogni dato viene ottenuto come la media dell'andamento giornaliero, si avrebbe un andamento con valori inferiori a quelli reali, dal momento che mediare valori dell'ora di punta con valori registrati nelle prime ore della mattina (bassi in modulo) porterebbe ad avere un andamento più basso dei primi e più alto dei secondi, andando di fatto a distorcere il segnale.

Il secondo deriva dal fatto che, per poter predire per un buon numero di settimane, occorre poter disporre di un buon set di dati, usati per ricavare con precisione la stagionalità (nel caso del modello ARMA) e per poter avere due set di dati distinti per l'inizializzazione e l'identificazione dei parametri (modello di Holt-Winters).

Tenendo a mente questo, si può capire come i dati, così come sono disponibili, risultano essere inadeguati:

- Per prima cosa non si possono considerare le settimane di agosto, in quanto presentano valori medi più bassi che nel resto dell'anno: se si scegliesse di usarle per valutare la bontà della predizione, si avrebbe una performance della predizione pessima, in quanto la predizione si attesterebbe a valori più alti rispetto ai dati. Ciò non sarebbe dovuto ad un errore, ma al fatto che i dati considerati per l'identificazione mostrano valori uniformemente maggiori rispetto a quelli di agosto; conseguentemente anche la predizione mostrerebbe valori simili che, confrontati con i dati di agosto, porterebbe a ritenere ingiustamente pessimo il predittore ricavato. Per tale motivo le ultime 4 settimane di dati non possono essere considerate.
- Considerando i dati campionati ogni 6 e 24 minuti, essi soddisfano il primo requisito, ma non il secondo: tali dati coprono infatti solamente 2 settimane in totale, essendo quindi insufficienti per poter essere utilizzati senza far ricorso al resto del set di dati a disposizione.
- Prendendo in esame i dati campionati ogni 144 minuti, ci si ritrova nella situazione opposta: combinati con i dati precedenti si ottengono un totale di 12 settimane di dati utili; purtroppo però avere un sample ogni 144 minuti porta ad avere un totale di 10 campioni ogni 24 ore, insufficiente a descrivere in maniera soddisfacente i segnali considerati.

Si è quindi deciso di ricorrere ad un artificio: a partire dai dati campionati ogni 144 minuti, ci si riconduce ad un segnale campionato ogni 24 minuti, ripopolando dapprima il set di dati andando ad interpolare una retta per ciascuna coppia di sample consecutivi e individuando su questa retta dei nuovi campioni equispaziati, e aggiungendo in un secondo momento un rumore per rappresentare l'andamento fortemente oscillatorio dei dati.

In particolare si è scelto di considerare un campionamento pari a 24 minuti poiché tale periodo è stato considerato in grado di soddisfare entrambi i requisiti qui sopra illustrati, ottenendo una buona approssimazione degli andamenti giornalieri del segnale e un numero di dati sufficiente su cui poter lavorare.

In questo modo si genera un segnale fittizio non direttamente riconducibile ai dati, ma che ci consente di poter considerare un caso di studio con dati sufficienti per poter identificare dei modelli che descrivano traffico e CPU e in un secondo momento poter fare predizione long-range. In caso contrario, non sarebbe stato possibile effettuare tale lavoro. Inoltre, scegliendo con attenzione il modo in cui si è ripopolato il set di dati e l'ampiezza del rumore, si è potuto ottenere un segnale ragionevolmente simile ad un traffico in ingresso ad un router con il medesimo tempo di campionamento. Si è infine ritenuto tale metodo migliore rispetto ad una simulazione in cui il segnale fosse generato partendo da zero, in quanto l'andamento alla base del segnale proviene da un caso reale.

In appendice 4.1 sono riportate più approfonditamente le operazioni effettuate sul segnale che sono state qui descritte.

4.2 **Modello arma**

4.2.1 Identificazione delle componenti dei segnali

Dopo aver ottenuto un segnale sufficientemente verosimile alla realtà, si passa ora alla fase di identificazione dei modelli che ci si è prefisso di usare. Il primo considerato è, come per la predizione di tipo short-range, un modello ARMA.

Consideriamo i segnali formati da tre componenti, pari a trend, stagionalità e parte stocastica del segnale: occorre quindi identificare le componenti deterministiche di trend e stagionalità e, a partire da queste, ottenere la terza componente, cioè la parte stocastica.

Per prima cosa si deve decidere in che modo dividere il set di dati, in modo da definire un set di identificazione e uno su cui fare validazione e confrontare i risultati della predizione. Dal momento che ci pone come obiettivo la predizione di un mese di dati, si sceglie di considerare un set di identificazione costituito da 8 settimane, lasciando le rimanenti per la validazione e il confronto con i risultati che si otterranno dal predittore.

Determinato il set a disposizione si ricavano le componenti di trend e stagionalità:

-per quanto riguarda il trend, si procede in modo analogo a quanto fatto nel capitolo 2, utilizzando le funzioni Matlab Polyfit e Polyval e supponendo anche in questo caso la presenza di un trend lineare, dal momento che tale ipotesi era verificati in precedenza e che, facendo riferimento allo stesso router, ci si aspetta che valga ancora.

I coefficienti (da intendere nella forma $y=at+b$) ottenuti per il traffico in ingresso sono:

$$a = -4 \cdot 10^{-6}\ [Mb/s^2] \quad b = 3.14\ [Mb]$$

Mentre quelli ottenuti per l'utilizzo di CPU:

$$a = -4 \cdot 10^{-6}\ [\%/s] \quad b = 3.53\ [\%]$$

Anche in questo caso il trend risulta essere quasi nullo sia per il segnale di traffico sia per quel che riguarda l'utilizzo di CPU; se ne terrà comunque conto per completezza.

-per quanto riguarda la stagionalità si procede in un modo leggermente diverso: nel capitolo 3 si era considerata una stagionalità costante per tutte le settimane del set di identificazione, identificata utilizzando le 3 settimane a disposizione. In questo caso, invece, avendo un numero di settimane maggiore, si può utilizzare un approccio differente, in cui la stagionalità viene riaggiornata ogni settimana: la stagionalità delle terza settimana, ad esempio, dipende solo dai valori dei giorni precedenti ad essa, quindi ogni valore risulta essere la media di 3 campioni; nell'ottava settimana, invece, ogni punto corrisponde alla media di 8 campioni diversi, rendendo la stagionalità ricavata molto più affidabile.

Questo metodo verrà utilizzato sia per il modello ARMA che per il metodo di Holt-Winters descritto più avanti nel capitolo. Per ottenere la stagionalità della prima settimana si decide di utilizzare il segnale originale (quindi senza l'aggiunta di rumore) depurato dal trend.

La stagionalità ottenuta alla fine delle 8 settimane è mostrata in figura 4.4 (per il traffico) e 4.5 (per la CPU), in cui viene confrontata con il segnale depurato da trend:

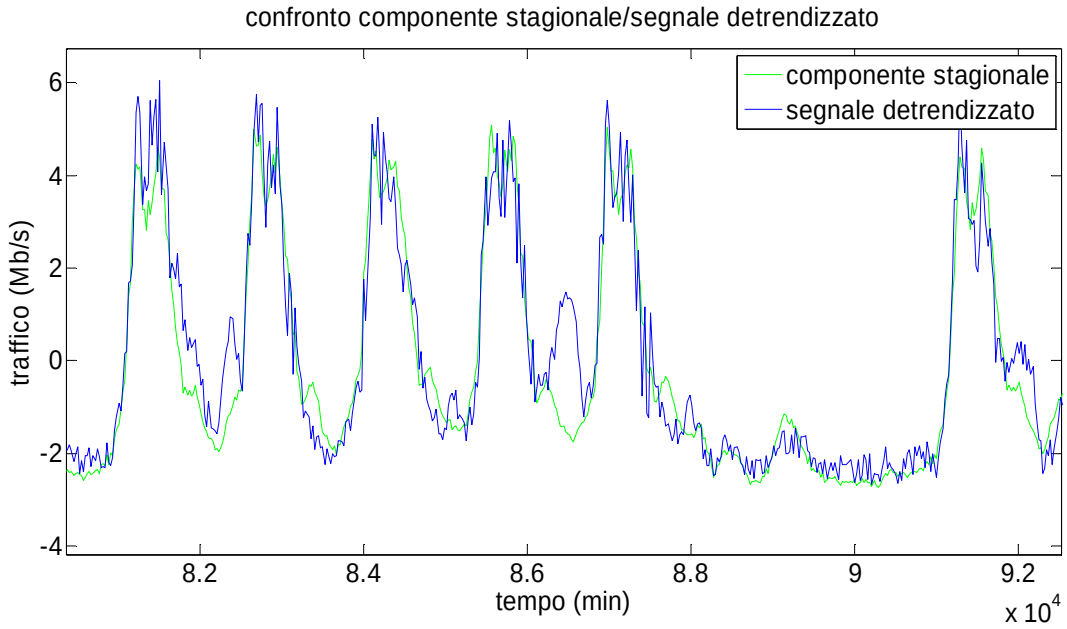


Figura 4.4: confronto tra il segnale depurato da trend (nona settimana) e la componente stagionale (traffico)

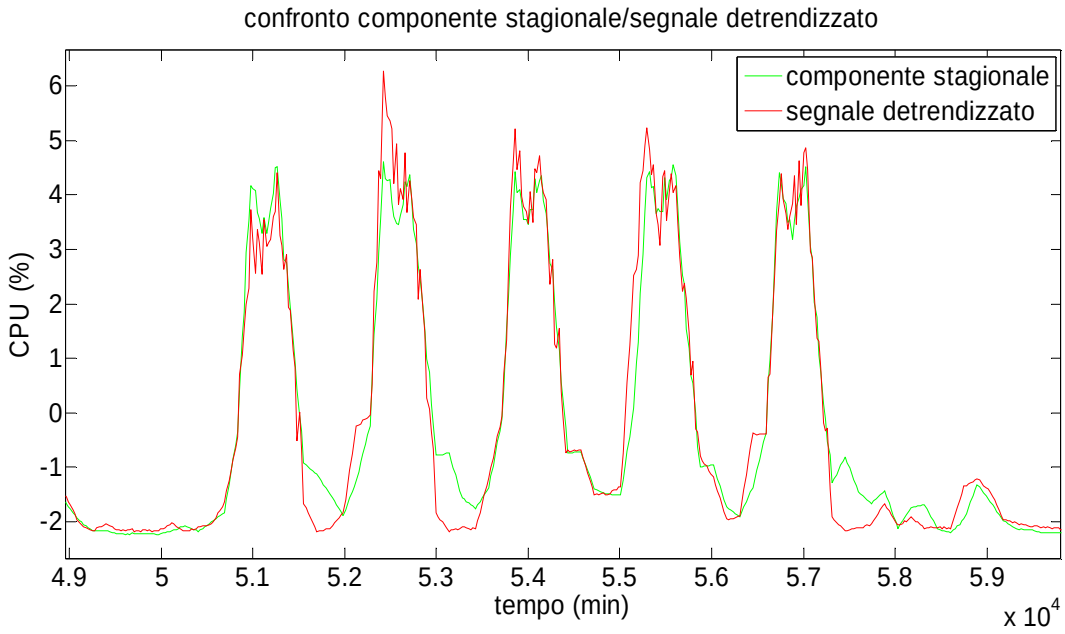


Figura 4.5: confronto tra il segnale depurato da trend (nona settimana) e la componente stagionale (utilizzo di CPU)

Come si può notare, in particolare per quanto riguarda il traffico, in corrispondenza di alcuni giorni si ha una discrepanza sostanziale tra l'andamento del segnale e la componente stagionale: ad esempio, nell'immagine che fa riferimento alla stagionalità aggiornata alla nona settimana, si ha un picco anomalo di attività il giovedì notte. Se si fossero considerate solamente 3 settimane tale anomalia avrebbe influenzato pesantemente l'andamento della stagionalità, portando a previsioni erronee. Avendo a disposizione 8 settimane, invece, il contributo dato dai valori anomali risulta molto minore, facendo sì che nelle settimane successive la stagionalità non si attesti su valori insolitamente alti durante le ore notturne del giovedì.

Ottenute le componenti deterministiche del segnale siamo ora in grado di ricavare la parte stocastica del segnale, su cui andare ad identificare il modello ARMA:

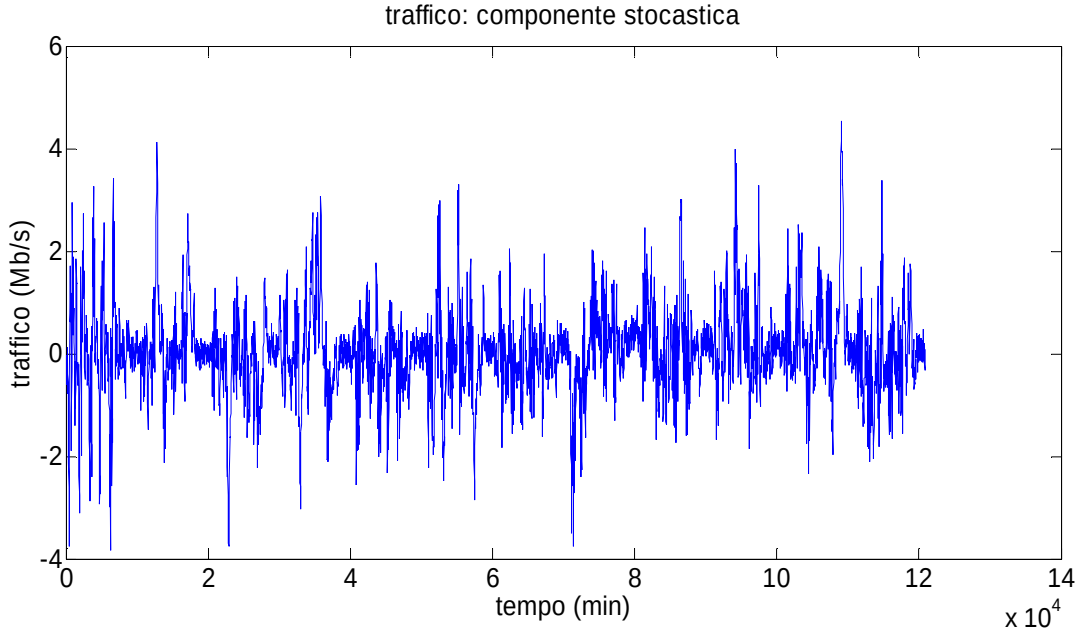


Figura 4.6: componente stocastica del segnale (traffico)

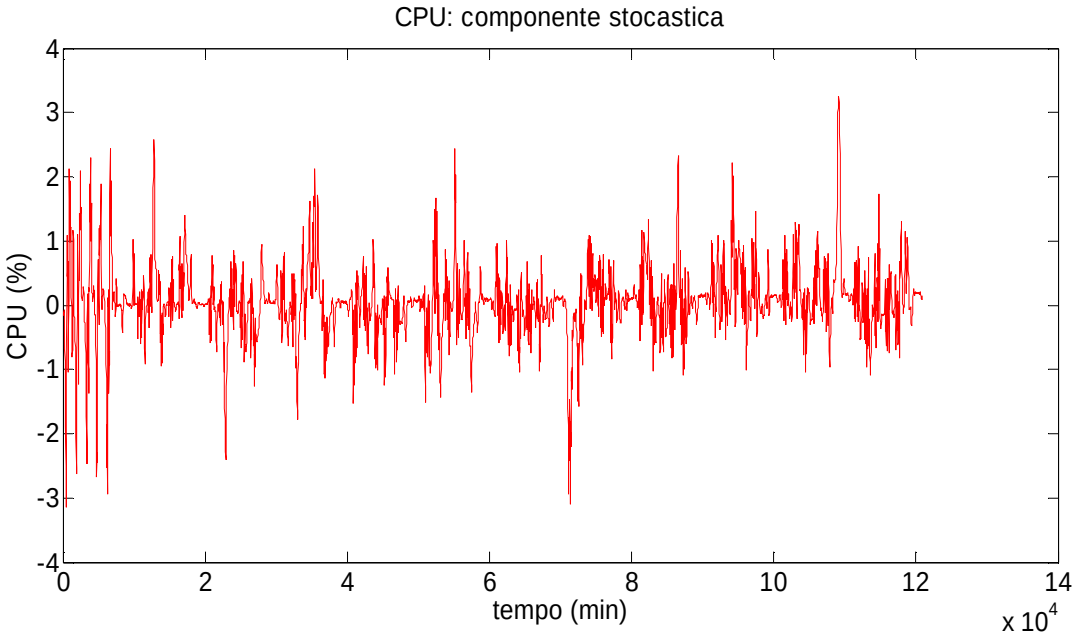


Figura 4.7: componente stocastica del segnale (CPU)

Da notare che tali segnali non corrispondono unicamente ai rumori bianchi aggiunti al segnale originale: se così fosse sarebbe impossibile identificare un modello ARMA con buone prestazioni. La parte stocastica è sì formata dal rumore aggiunto, ma anche dalla differenza tra il segnale originale e le componenti deterministiche, calcolate sul segnale totale (originale più rumore bianco aggiunto). In questo modo si ottengono delle componenti stocastiche δu e δy diverse da dei white noise. Inoltre si nota come, dato che all'aumentare del numero di settimane considerate la stagionalità si assesta su valori più o meno simili, gli andamenti della componente stocastica risultino molto diversi se si confrontano le prime 2 settimane tra di loro, mentre le restanti settimane mostrano andamenti simili.

4.2.2 Identificazione del modello

Si passa quindi ad identificare il modello ARMA, concentrandosi sul segnale di traffico δu. Considerando l'equazione di un predittore ARMA:

$$\varphi(z^{-1})\delta u(t) = \vartheta(z^{-1})e(t)$$

Si devono identificare gli ordine delle parti AR ($\varphi(z^{-1})$) e MA ($\vartheta(z^{-1})$) ed i relativi parametri. Per prima cosa si determina l'ordine. Seguendo il procedimento utilizzato nel capitolo recedente, consideriamo le seguenti cifre di merito:

- Loss function: $J = \frac{1}{N}\sum_{i=1}^{N}(\delta u_i - \delta\hat{u}_i)^2$

- Loss function normalizzata sulla varianza: $errore = \frac{1}{N}\frac{\sum_{i=1}^{N}(\delta u_i - \delta \hat{u}_i)^2}{var(\delta u)}$
- Akaike information criterion: $AIC = 2\frac{n}{N} + \ln(J)$
- Final prediction error: $FPE = \frac{N+n}{N-n} J$

In cui N è pari a 3360 (numero di punti campionando ogni 24 minuti per 8 settimane), *n* è invece l'ordine del modello che di volta in volta si considera.

Si riportano qui i risultati per la sola seconda cifra di merito considerata, in appendice 4.2 sono presenti i risultati per tutte le cifre di merito rimanenti.

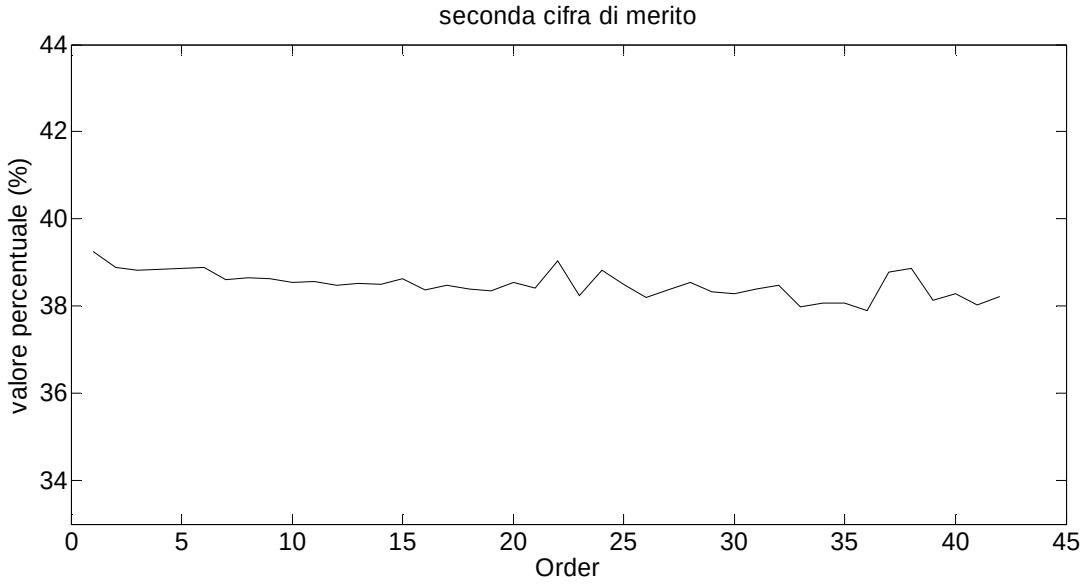


Figura 4.8: Andamento della cifra di merito considerando la componente stocastica del segnale

Come si può osservare, all'aumentare dell'ordine considerato non si ha un concreta diminuzione della cifra di merito, ma solo un miglioramento minimo. Inoltre, a differenza del capitolo precedente, dove l'errore si attestava intorno a valori del 16-17%, in questo caso si ha uno scarto molto maggiore, segno che il modello sarà in grado di approssimare solo in modo discreto le dinamiche della componente stocastica del traffico.

Per decidere quale sia l'ordine in grado di approssimare meglio il segnale si ricavano le mappe di poli e zeri e tiene conto di quante cancellazioni polo-zero si abbiano per ciascun ordine. Consideriamo come ordini-campione gli ordine 3 e 4 (esempi di ordini bassi) e l'ordine 16 (ordine considerato elevato) e si ricavano le relative mappe di zeri e poli.

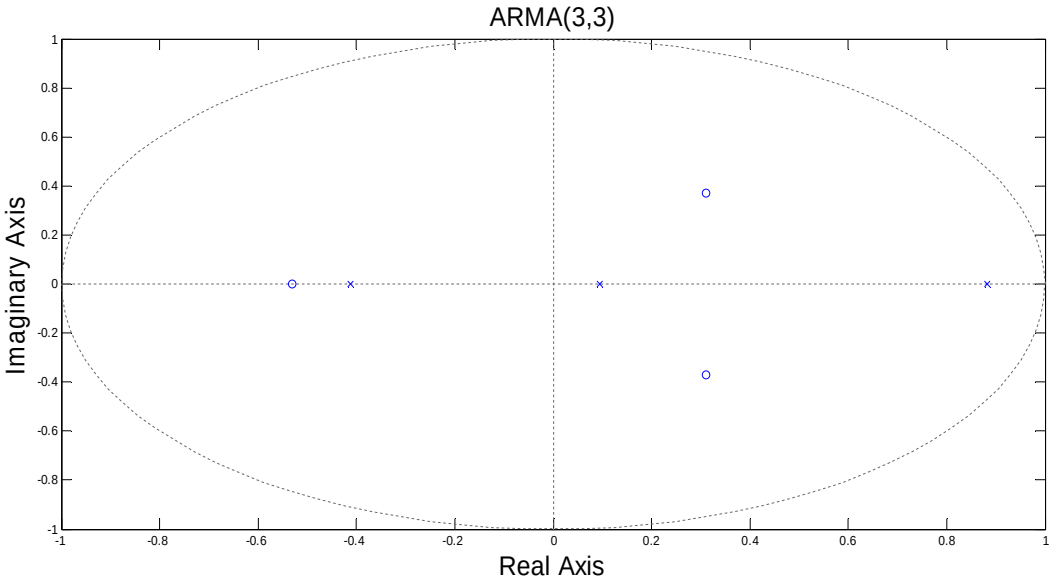


Figura 4.9 Mappa di poli e zeri relativa al modello ARMA (3,3)

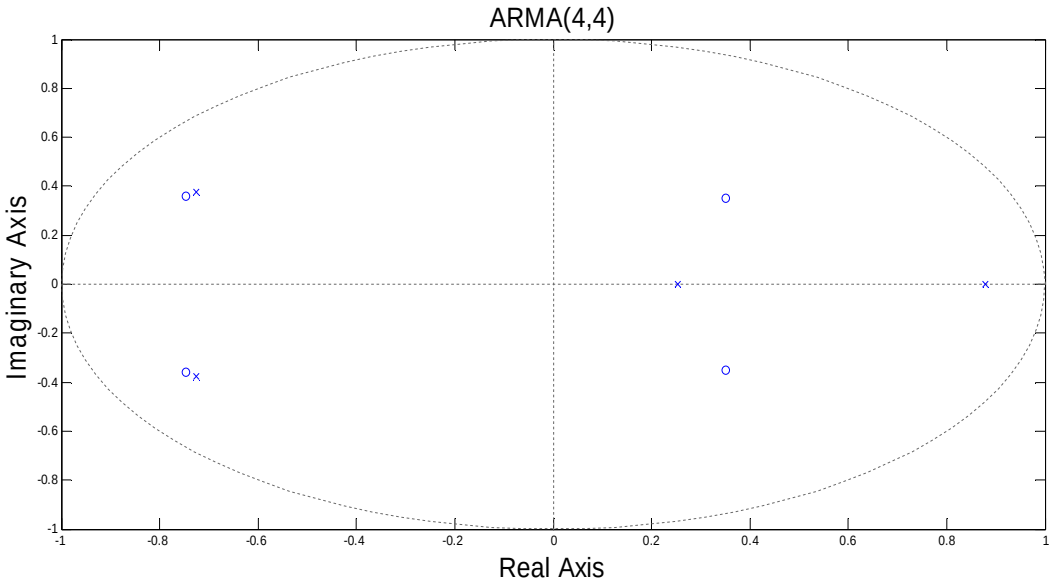


Figura 4.10 Mappa di poli e zeri relativa al modello ARMA (4,4)

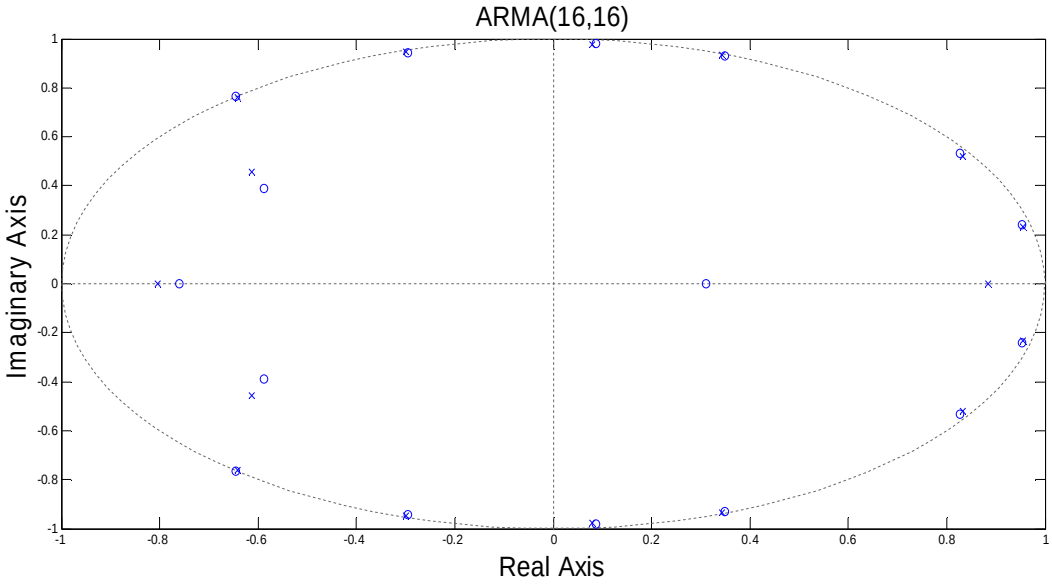


Figura 4.11 Mappa e poli e zeri relativa al modello ARMA (16,16)

Come si può osservare il numero di poli che non viene cancellato risulta anche in questo caso particolarmente basso, e si attesta ad un valore compreso tra 2 e 4, facendo pendere la scelta per un ordine basso, dal momento che all'aumentare di quest'ultimo si ha solo un aumento delle cancellazioni senza un sostanziale miglioramento delle prestazioni.

Da ultimo si considerano la funzione di trasferimento ricavata dal traffico e comparata con quelle dei modelli considerati:

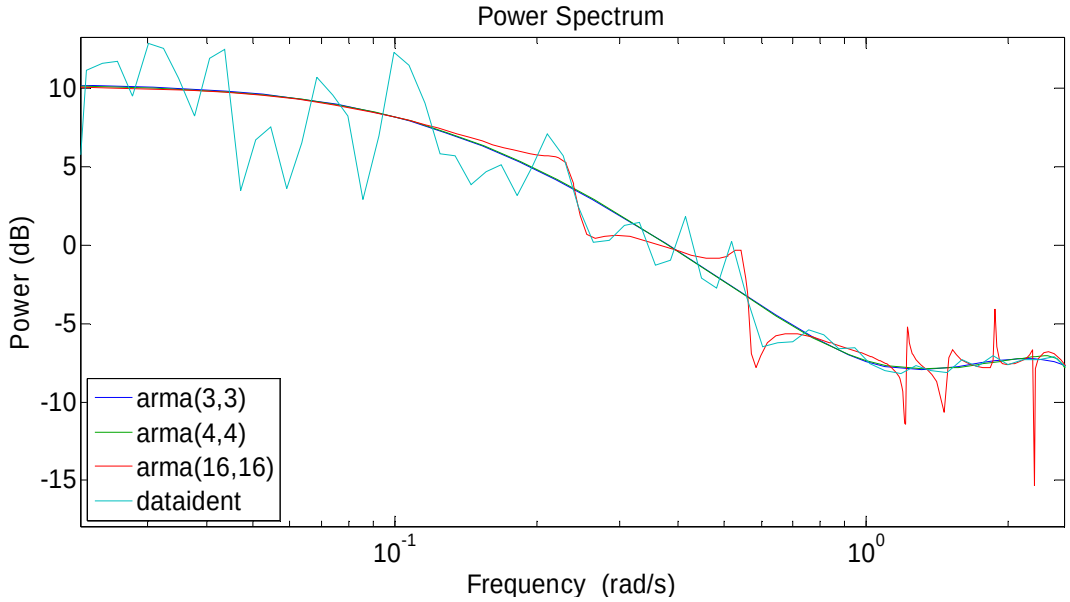


Figura 4.12: confronto tra la funzione di trasferimento ricavata a partire dai dati e le relative ai modelli identificati

Come si può vedere dalla figura 4.12, gli andamenti delle funzioni di trasferimento per gli ordini 3 e 4 sono simili, mentre quello del modello di ordine 16 presenta una funzione di trasferimento più

irregolare, in particolare nell'ultimo tratto considerato in figura, portando ad uno scostamento dalla funzione ricavata dai dati.

4.2.3 Validazione

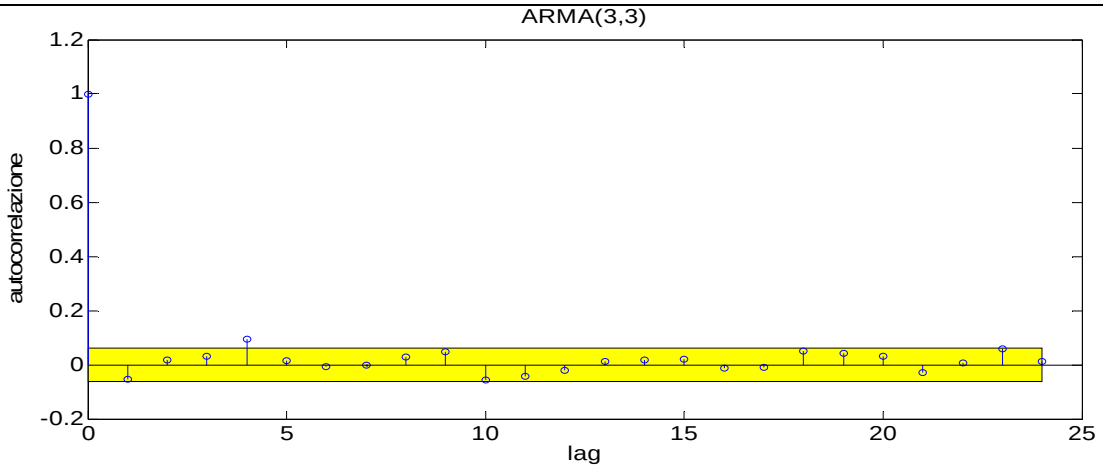
Dal momento che le prestazioni del modello per diversi ordini risultano simili e l'unica indicazione per decidere quale sia il più adatto risulta essere il numero di cancellazioni di polo-zero, si passa a considerare il set di validazione: qualora anche in questo caso le performance dovessero essere simili, si sceglierebbe l'ordine basso che presenta miglior fitting, seppur di poco rispetto agli altri.

Per poter validare i modelli si ricavano i predittori ad un passo per ogni modello e li si confronta con i dati del set, che rappresentano un totale di 4 settimane di samples. Come fatto in precedenza, si propone un analisi dei residui mediante un test di bianchezza per poter verificare che i predittori calcolati siano ottimi, in modo tale da poter essere sicuri che le percentuali di fitting che si otterranno esprimano fedelmente la bontà dei modelli e non siano condizionati da eventuali errori nella determinazione del predittore. Si definisce un residuo come la differenza tra l'uscita del predittore ad un passo ($\hat{u}(t+1|t)$) ed il relativo valore reale ($u(t+1)$).

Se la correlazione tra due qualsiasi residui relativi ad istanti di tempo diversi è pari a zero allora il test è superato e il predittore ricavato è effettivamente quello ottimale, in caso contrario il test è negativo; ciò significa che è possibile ottenere un predittore migliore.

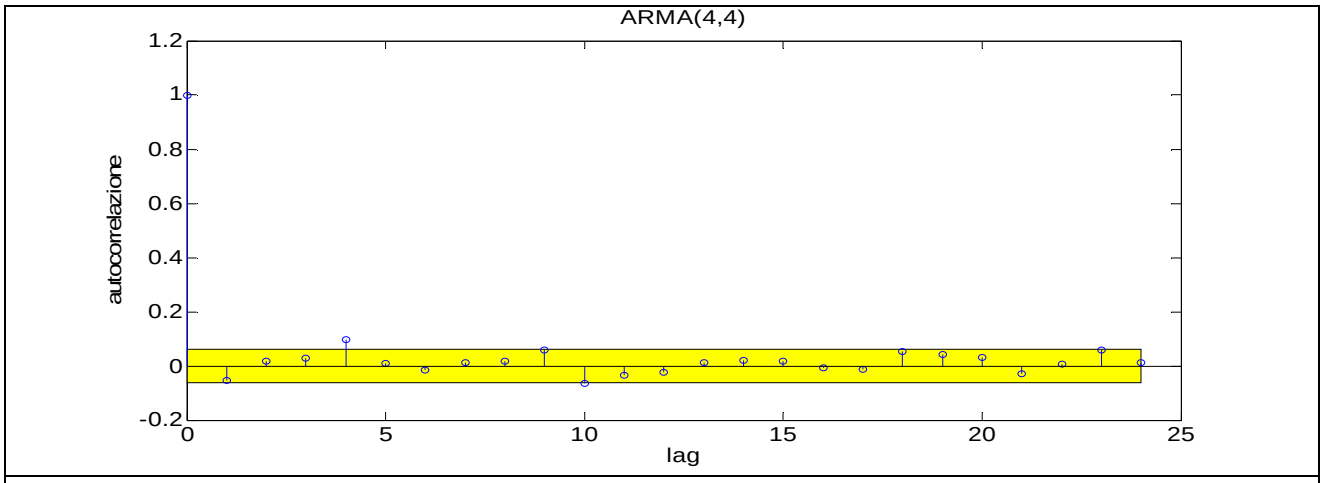
Viene definito inoltre per ogni test un intervallo di tolleranza entro il quale il residuo è considerato essere circa pari a zero.

In figura 4.13, 4.14 e 4.15 sono mostrati i risultati:



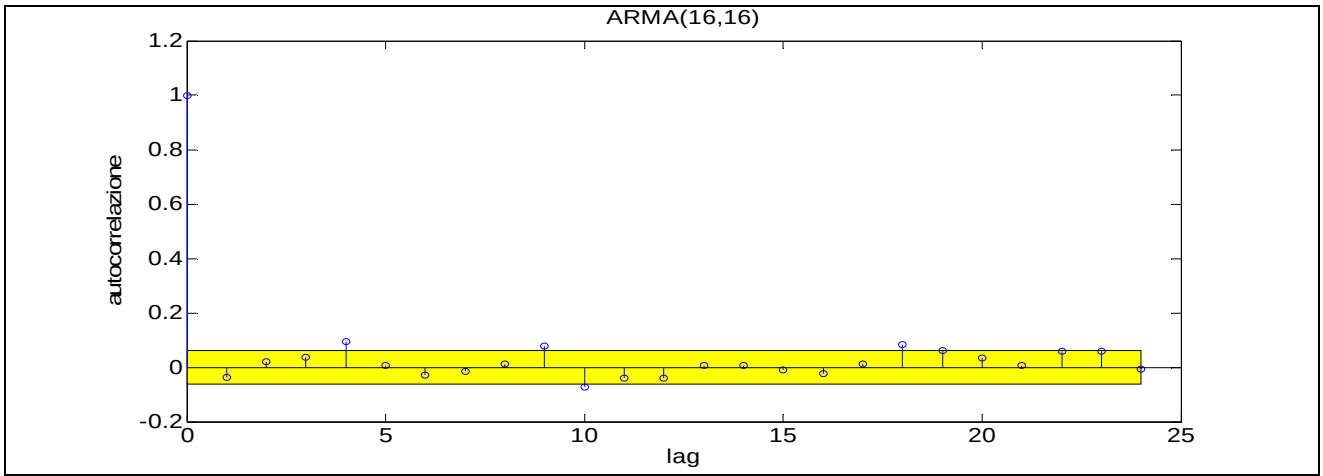
Residui relativi al predittore ricavato dal modello ARMA(3,3)

Figura 4.13



Residui relativi al predittore ricavato dal modello ARMA(4,4)

Figura 4.14



Residui relativi al predittore ricavato dal modello ARMA(16,16)

Figura 4.15

Dal momento che, per ogni modello, quasi tutti i valori di correlazione sono all'interno dell'intervallo di tolleranza, si possono considerare ottimi i predittori ricavati.

Si procede quindi a ricavare le prestazioni dei modelli considerati per il set di validazione:

Ordine del modello	Fitting su componente stocastica (δu)	Fitting su segnale totale (u)
ARMA(3,3)	65.578%	95.932%
ARMA(4,4)	65.505%	95.923%
ARMA(16,16)	65.044%	95.867%

Come si può osservare dai risultati, anche per quanto riguarda il set di validazione non si hanno sostanziali miglioramenti all'aumentare dell'ordine considerato. Per quanto riguarda la percentuale di fitting sul segnale depurato da trend e stagionalità non si hanno variazioni, dal momento che l'errore di identificazione si attestava intorno al 35%. Dal momento che il modello di ordine 3 porta ad un fitting leggermente migliore, e che è il più basso tra tutti quelli considerati, si

decide di considerare d'ora in poi tale modello per la descrizione del segnale di traffico. Da sottolineare come anche nel precedente capitolo si era giunti a una conclusione analoga. In figura 4.16 si può osservare il confronto grafico tra i dati (in rosso) ed l'uscita del modello (in blu).

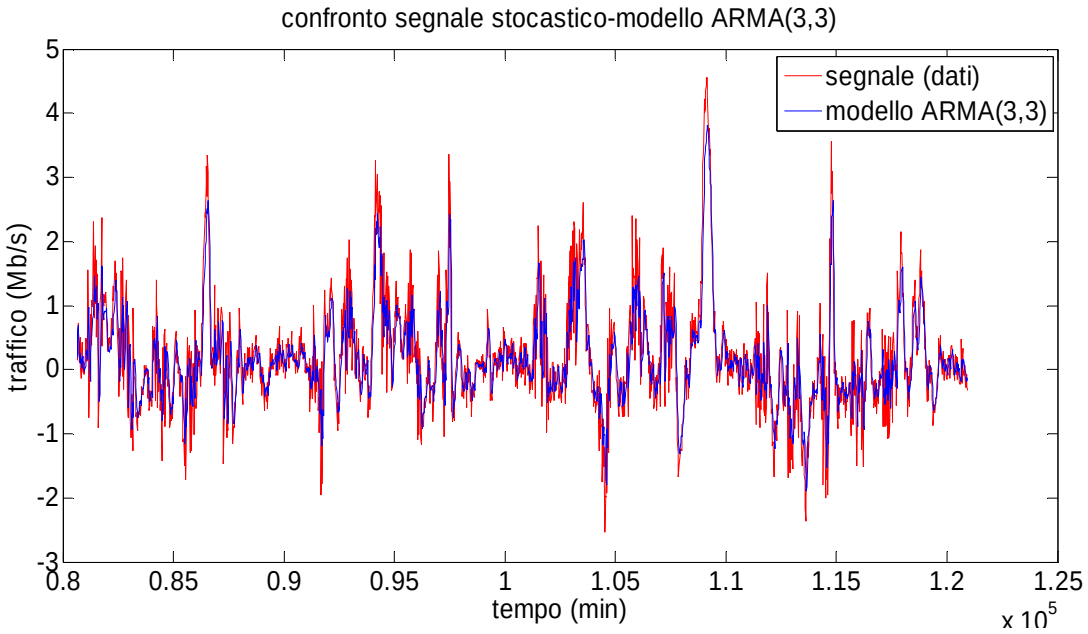


Figura 4.16: confronto tra segnale stocastico ricavato a partire dai dati e l'uscita del modello (traffico)

Come si può vedere, l'errore principale è dovuto al fatto che il modello non riesce ad eguagliare i valori di picco che i dati assumono, riuscendo però a seguire in generale la dinamica, seppur non in modo ottimale come nel capitolo precedente. Il non riuscire a raggiungere i valori massimi del segnale non va ad impattare in modo critico nella stima dei massimi valori raggiunti dal traffico totale (u), in quanto la differenza tra dato e uscita del modello risulta molto inferiore rispetto alla grandezza totale del segnale, come mostrato in figura 4.17, in cui si considera la seconda delle 4 settimane del set di validazione:

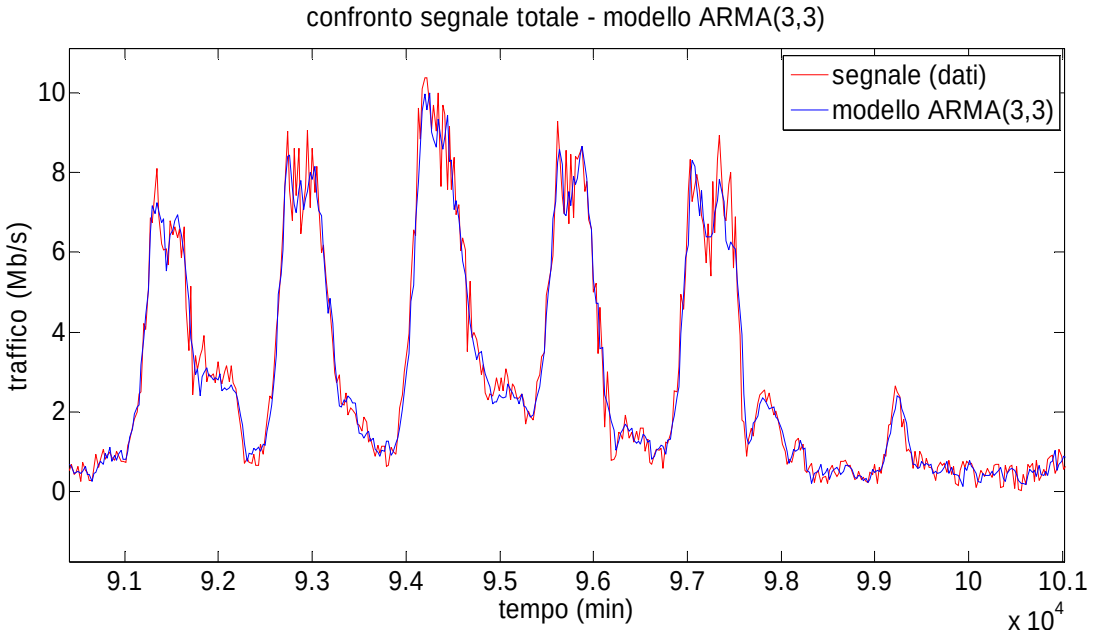


Figura 4.17: confronto tra i dati e l'uscita del modello (segnale totale di traffico)

Deciso quindi l'ordine del modello si mostrano i parametri dei polinomi φ e ϑ.

Per quanto riguarda il polinomio della parte autoregressiva del modello:

$$\varphi_1 = 0.5634 \varphi_2 = -0.3192 \varphi_3 = 0.0342$$

Mentre per la parte a media mobile:

$$\vartheta_1 = -0.0913 \vartheta_2 = -0.09776 \vartheta_3 = 0.1239$$

Terminata la parte di identificazione del modello si passa quindi a ricavare il predittore per il traffico.

4.2.4 Predizione del traffico in ingresso ad un router

Determinato un modello che sia in grado di descrivere gli andamenti del segnale, si procede a ricavare il predittore per il traffico. Ci si pone come obiettivo la predizione di un lasso di tempo pari a 4 settimane, considerato un periodo sufficientemente lungo per poter andare ad apportare modifiche sulla rete qualora i risultati dovessero mostrare andamenti di traffico, e conseguentemente allocazione di risorse, tali per cui si rischierebbe di entrare in zona di sovraccarico dei router e quindi avere una riduzione delle prestazioni. In questo modo i risultati di questa tesi saranno direttamente applicabili in un'ottica di capacity planning.

Per poter ottenere la predizione voluta è necessario iterare la lunga divisione tra i due polinomi del modello ARMA per un numero di volte pari all'orizzonte temporale h che ci si è prefissi. Dal momento che si considera un segnale campionato con frequenza pari a 24 minuti, un orizzonte temporale pari a 4 settimane corrisponde ad avere h=1680. A partire dell'equazione del modello ARMA si deve perciò iterare la lunga divisione per un numero di volte pari ad h e salvare il risultato di ogni iterazione; successivamente, si ricavano per ogni istante di predizione il corrispondente predittore, dal momento che, come logico, il predittore per l'istante $t + h$ con $h = 10$, ad esempio, dovrà necessariamente essere diverso da quello per $h = 100$.

Ricavati i valori predetti ci si rende però conto di come il modello ARMA fornisca una previsione inadeguata per la componente stocastica del segnale: il predittore tende alla media del processo in breve tempo, ed essendo la media della componente stocastica nulla, la predizione si riduce ad essere pari a 0 in un numero di passi pari a circa 35 (pari a 840 minuti, quindi 13 ore circa). Sostanzialmente si può quindi affermare che la predizione della componente stocastica non dà alcun apporto alla predizione del traffico di tipo long-range.

La predizione per il segnale nella sua totalità si riduce perciò ad essere pari alle sole componenti deterministiche. Si prende quindi in considerazione il segnale totale, la cui predizione coinciderà con i valori di trend e stagionalità calcolati sulle otto settimane del set di identificazione.

In figura 4.18 è riportato il confronto tra i dati e la predizione:

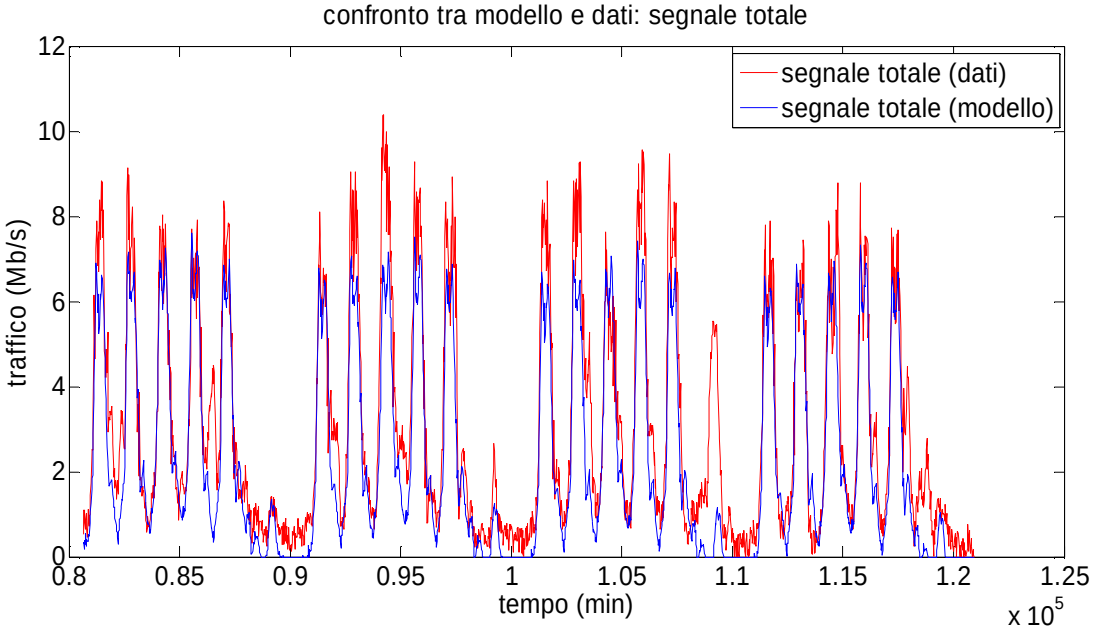


Figura 4.18 confronto tra i dati di traffico e la predizione del modello (segnale totale)

In cui è stata considerata anche in caso una saturazione per far sì che la predizione non possa assumere valori negativo:

$$\hat{u}(t) = \begin{cases} \hat{u}(t) & \text{se } \hat{u}(t) \geq 0 \\ 0 & \text{se } \hat{u}(t) < 0 \end{cases}$$

Si può osservare come in generale l'andamento della previsione segua quello dei dati, pur con alcune differenze: dal momento che il contributo della parte stocastica è nullo, il valore massimo raggiunto dalla predizione risulterà minore di quello assunto dai dati. Inoltre, eventuali aumenti dell'attività avvenuti unicamente nel set di validazione non potranno essere descritti, dal momento che la stagionalità viene calcolata sul set di identificazione e si assesta su valori minori, ciò porta ad un ulteriore differenza tra i valori di picco dei due segnali considerati (come si vede osservando la seconda e la terza settimana).

Infine considerando il sabato della terza settimana di predizione, si vede come eventi inaspettati non possano essere previsti, indipendentemente dalle prestazioni del modello ARMA: un'attività intensa in un giorno come il sabato, in cui per tutto il resto del set di dati considerato non si hanno anomalie, non può essere in alcun modo predetta, indipendentemente dal modello che si consideri per ricavare la predizione.

Considerando la cifra di merito usata fino ad ora per valutare la bontà della predizione, si ottiene:

$$fitting = \left(1 - \frac{1}{N}\frac{\sum_{i=1}^{N}(u_i - \hat{u}_i)^2}{var(u)}\right) * 100 = 78.87\%$$

In cui N vale 1680 (numero di campioni in 4 settimane). Un fitting intorno all'80% equivale a dire che il modulo al quadrato dell'errore è pari ad un quinto della varianza del segnale, perciò la predizione ottenuta a partire dalla stagionalità risulta molto migliore di quella che si otterrebbe

utilizzando la media del segnale e considerando un intervallo (ottenuto sommando e sottraendo la deviazione standard del segnale) in cui dovrebbe andare a cadere la predizione del segnale.

Si potrebbe quindi considerare soddisfacente la predizione ottenuta a partire dalla sola parte deterministica del segnale, anche tenendo conto del fatto che, in questo modo, ricavare tale predizione per il traffico diventa immediato.

4.2.5 Predizione di utilizzo di CPU

Si passa infine a predire l'andamento dell'utilizzo di CPU, utilizzando come nel capitolo precedente la relazione statica che individua un coefficiente per ciascuna componente del segnale. Per ricavare tali coefficienti si considera nuovamente il set di identificazione e si ottiene:

$$K_c^o = \sum_{i=1}^{N} \delta y_i \delta u_i \Big/ \sum_{i=1}^{N} {\delta u_i}^2 = 0.6224[\%/(\text{Mb/s})]$$

$$K_s^o = \sum_{i=1}^{N} s_{y,i} s_{u,i} \Big/ \sum_{i=1}^{N} {s_{u,i}}^2 = 0.9334[\%/(\text{Mb/s})]$$

$$K_t^o = \sum_{i=1}^{N} \bar{y}_i \bar{u}_i \Big/ \sum_{i=1}^{N} {\bar{u}_i}^2 = 1.0772[\%/(\text{Mb/s})]$$

In cui N è pari a 3360 (numero di campioni in 8 settimane). Da notare che per completezza si è ricavato anche il coefficiente dedicato alla parte stocastica del segnale, per quanto il suo apporto alla predizione risulti praticamente nullo come già detto nel paragrafo precedente.

Moltiplicate le predizione di traffico per i corrispondenti coefficienti, si ottiene una predizione per la CPU nella forma:

$$\hat{y} = K_c^o \delta u + K_s^o s_u + K_t^o \bar{u}$$

Anche in questo caso viene introdotta una saturazione, in modo tale che la predizione non assuma valori inferiori all'1%, per caratterizzare il fatto che una minima parte delle risorse rimane perennemente allocata per i processi interni del router:

$$\hat{Y}(t) = \begin{cases} \hat{Y}(t) & se\ \hat{Y}(t) \geq 1 \\ 1 & se\ \hat{Y}(t) < 1 \end{cases}$$

In figura 4.19 è mostrato il confronto tra valori predetti e dati:

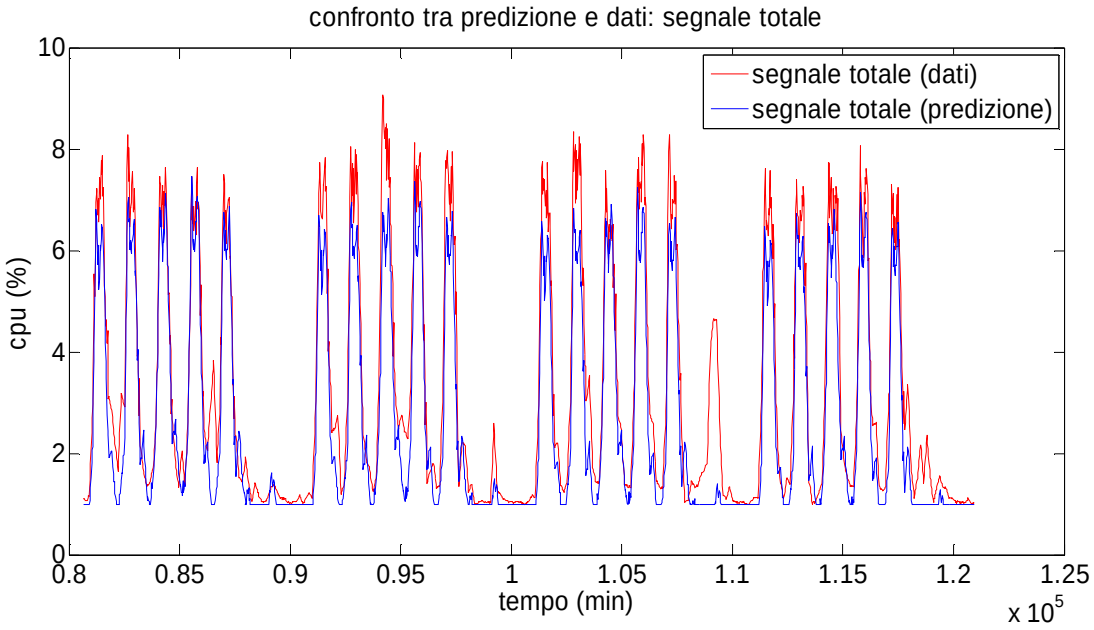


Figura 4.19: confronto tra i dati (utilizzo di CPU) e la predizione del modello

Per la predizione dell'utilizzo di CPU valgono le considerazioni fatte per il traffico:

- Lo scostamento tra dati e predizione nelle ore diurne è dovuto alla mancanza di contributo della componente stocastica.
- Eventi totalmente impredicibili come un'attività anomala in un giorno feriale non possono essere predetti, dal momento che non ce n'è traccia nel set di identificazione.

Ciononostante in generale l'andamento dei dati viene seguito abbastanza fedelmente dal predittore, soprattutto nei periodi del giorno in chi si ha una rapida variazione dell'utilizzo di risorse (quindi nelle prime ore del mattino e sul finire del pomeriggio) dove si può vedere una sovrapposizione tra i due segnali.

Per quanto riguarda i valori che il fitting assume:

$$fitting = \left(1 - \frac{1}{N}\frac{\sum_{i=1}^{N}(y_i - y_i)^2}{var(y)}\right) * 100 = 82.04\%$$

Si può osservare come essi siano simili ai valori che si assumevano per il traffico, con un leggero miglioramento. Da questo si può concludere come la componente fondamentale del segnale sia ancora una volta la stagionalità, e che quindi ci si debba concentrare su di essa per poter avere buone prestazioni in predizione. Minore è invece l'apporto della componente stocastica, dal momento che la sua totale mancanza fa sì che si abbia un errore del 18%; da ciò si può quindi concludere come il suo apporto sia pari ad un quinto del segnale totale.

4.2.6 Conclusioni relative al modello ARMA

Riassumendo, a fronte di quanto visto finora e in base ai risultati ottenuti, si può affermare quanto segue:

- un modello ARMA non è in grado di fornire prestazioni soddisfacenti per la predizione di traffico con un orizzonte temporale di tipo long-range, dal momento che il predittore tende, in un numero di passi relativamente breve, alla media del segnale che descrive. Essendo la componente stocastica a media nulla, l'apporto di un predittore ARMA è ininfluente, se non nei primi istanti di predizione, lungo i quali si ha un transitorio dai valori di partenza verso la media.
- Una predizione formata unicamente dalle parte deterministiche del segnale, ossia trend e stagionalità, è in grado di descrivere in modo adeguato l'andamento futuro del traffico, mostrando dinamiche simili alla realtà, anche se con valori di picco leggermente inferiori, dovuti alla mancanza del contributo della componente stocastica.
- Per la CPU valgono ragionamenti analoghi, dal momento che il passaggio a partire dal traffico si effettua attraverso 3 coefficienti statici.

In base a questi risultati si può osservare come, scegliendo di fare predizione unicamente con le componenti deterministiche del segnale, si potrebbero ottenere delle performance come quelle evidenziate in questo capitolo attraverso un metodo di facile implementazione, dal momento che si eviterebbe di identificare qualsiasi modello, e si dovrebbero solamente ricavare le componenti del segnale e i coefficienti per la conversione.

4.3 **Modello ARIMA**

All'interno di questo capitolo, a differenza del precedente, non viene sviluppato un modello ARIMA per la predizione di tipo long-range. Ciò è dovuto al fatto che, come mostrato nel capitolo precedente, le prestazioni di tale modello degradano sensibilmente all'aumentare dell'orizzonte predittivo. Come evidenziato dalle figure 3.53 e 3.50, tutti i modelli considerati mostravano, come ci si poteva aspettare, un peggioramento delle prestazioni all'aumentare del lasso di tempo di cui si voleva far predizione, ma ciò era particolarmente marcato per il modello ARIMA, mentre risultava meno evidente nel modello ARMA e nel metodo di Holt-Winters.

Nel caso del modello ARIMA sapendo che, una volta che il segnale in ingresso u è stato differenziato, il modello si comporta di fatto come un ARMA, si capisce come la predizione tenderà nuovamente alla media del segnale. Considerando il segnale di traffico differenziato, di cui si mostra la predizione dei primi 3 giorni in figura 4.20, i risultati sono i seguenti:

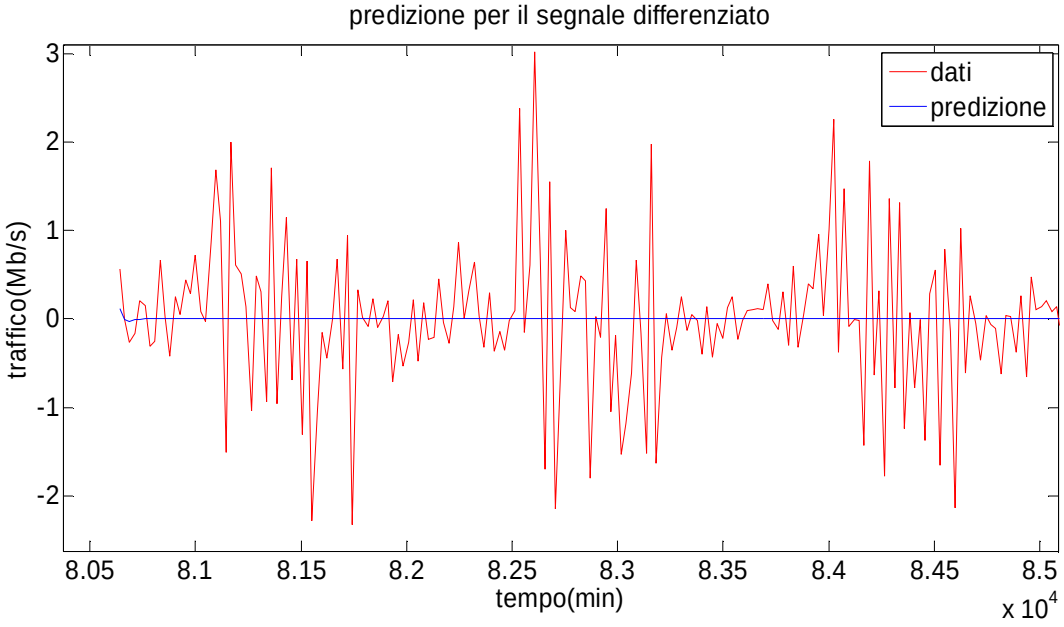


figura 4.20: confronto tra i dati e la predizione del modello ARIMA per il segnale differenziato

Si può notare come la media del segnale sia nulla, così come la predizione per il segnale a parte un breve transitorio iniziale. Si otterrebbe così una predizione per il segnale ∇u del tutto insoddisfacente.

Essendo il segnale ∇u ottenuto per differenziazione, per ottenere il segnale originale u si dovrà effettuare l'operazione inversa: dal momento che $\nabla u(t) = u(t) - u(t-1)$, per riottenere $u(t)$ si dovrà considerare il primo valore che il segnale assume ad andare poi a sommare di volta in volta il segnale $\nabla u(t)$ (dato che $u(t) = \nabla u(t) + u(t-1) = \nabla u(t) + \nabla u(t-1) + u(t-2) = \ldots$). Se però la serie predetta $\nabla \hat{u}(t)$ presenta valori diversi da zero solo per un breve transitorio, si capisce come la predizione per il segnale di traffico u mostrerà un transitorio iniziale seguito da un valore costante, predicendo quindi un andamento totalmente diverso da quello che il segnale assume. In

figura 4.21 si mostra la predizione (per la prima delle 4 settimane che si vogliono predire) che si otterrebbe utilizzando un modello ARIMA(3,1,3), cioè avente lo stesso ordine del modello utilizzato per la predizione di tipo short-range, dal momento che si è osservato come, per il modello ARMA, l'ordine del modello identificato rimanga identico:

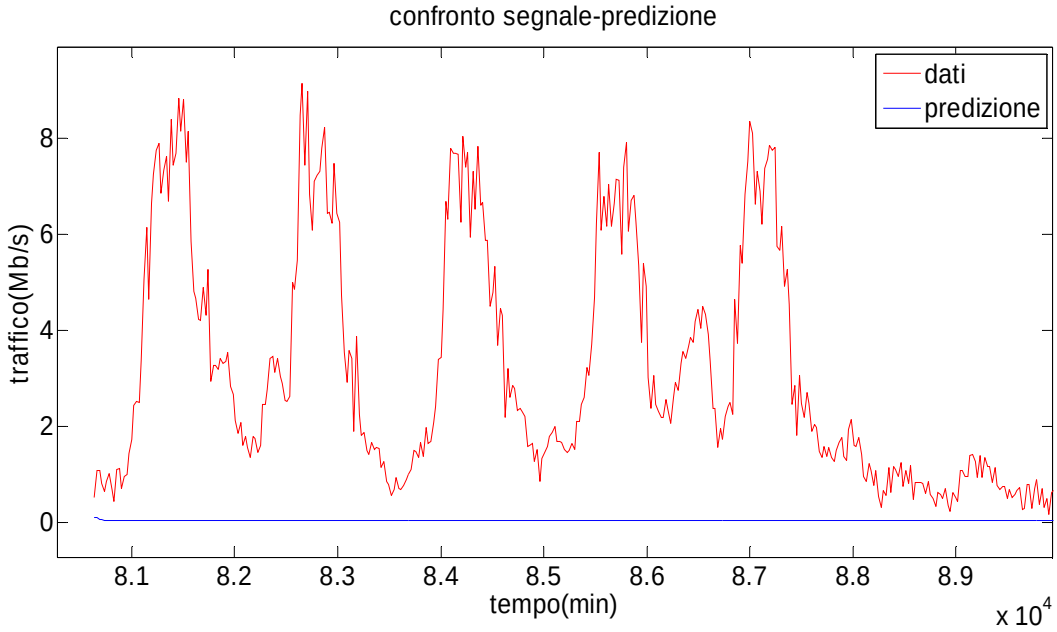


figura 4.21: confronto tra i dati e la predizione del modello ARIMA per il segnale totale: si noti come essa differisca completamente dall'andamento proprio dei dati

Si può osservare come si sia ottenuto ciò che si era anticipato: l'uscita del predittore tende ad assestarsi ad un valore costante, portando ad una predizione totalmente erronea del segnale. Per questi motivi si è deciso quindi di non utilizzare tale modello, ma di concentrarsi sul modello ARMA e sul metodo di Holt-Winters.

4.4 **Metodo di Holt-Winters**

Si considera ora il metodo di Holt-Winters, di cui si sono mostrati i fondamenti teorici nel capitolo precedente, per una predizione di tipo long-range. In questo caso le 12 settimane di dati vengono così suddivise: le prime 3 vengono usate per ottenere i valori iniziali di stagionalità e trend per poter permettere alle equazioni del modello di partire da una condizione iniziale con una stagionalità ottenuta da tre campioni per ogni media (e quindi considerata attendibile); le 5 settimane seguenti vengono usate per la taratura dei parametri λ_i, sfruttando la minimizzazione dell'errore di predizione per diversi orizzonti h; infine le ultime 4 settimane vengono confrontate con la predizione per ottenere un indice delle prestazioni dei risultati ottenuti.

A differenza del modello ARMA il metodo di Holt-Winters non prevede una predizione vera e propria per la componente stocastica, infatti nella predizione il valore di $L(t)$ che si va a sommare a trend e stagionalità non è altro che un valore costante (pari all'ultimo valore assunto dalla serie temporale $L(t)$ nell'ultimo istante disponibile) che può essere più o meno vicino al valor medio del segnale originale a seconda della lisciatura effettuata (e quindi della scelta di λ_1). Ciò che va a formare quindi il maggior contributo della predizione sono il trend $F(t)$ e la stagionalità $S(t)$; il fatto che queste due componenti rappresentino la parte preponderante del segnale considerato rende tale metodo molto vantaggioso per la predizione a lungo termine.

Al contrario della situazione presentata nel capitolo precedente ora si dispone di sufficienti settimane per poter dividere la fase di inizializzazione da quella di taratura, si ricorda infatti come dati usati nell'ambito della predizione short-range non fossero sufficienti per separare le due fasi in due diversi set di dati, portando all'utilizzo di un'inizializzazione inusuale, ottenuta filtrando la prima settimana di traffico a disposizione. Con diverse inizializzazioni si ottengono diversi parametri in fase di taratura, il che indica la forte dipendenza del metodo dalle condizioni iniziali. Potendo ora dividere il set di identificazione in 2 insiemi separati ci si aspetta che i parametri non ricadano nei casi limite di valori pari a 0 o 1 poiché queste situazioni sono indice di uno sfruttamento ridotto delle potenzialità del metodo. In particolare quando $\lambda_i = 0$ si evince come i valori iniziali siano considerati ottimi per la predizione, non sfruttando tuttavia la capacità di aggiornamento automatico del metodo; viceversa per $\lambda_i = 1$ si aggiorna continuamente il valore della serie indicando che i dati si discostano molto da quelli usati per l'inizializzazione. Si fa eccezione per il valore di λ_2 che non ha motivo di avere valori diversi da 0, dal momento che il trend ricavato in fase di inizializzazione utilizzando 3 settimane di dati risulta sempre migliore di quello ricavato dalla differenza tra 2 valori consecutivi, portando quindi a non aggiornare mai il valore della serie $F(t)$.

Come visto in precedenza le tre serie $L(t)$, $F(t)$ e $S(t)$ sono ottenute a partire dalle equazioni ricorsive:

$$\begin{cases} L(t) = \lambda_1\big(u(t) - S(t-s)\big) + (1-\lambda_1)\big(\tilde{u}(t-1) + F(t-1)\big) \\ F(t) = \lambda_2\big(\tilde{u}(t) - \tilde{u}(t-1)\big) + (1-\lambda_2)F(t-1) \\ S(t) = \lambda_3\big(u(t) - L(t)\big) + (1-\lambda_3)S(t-s) \end{cases}$$

In cui il parametro s è il periodo di stagionalità, che per il nuovo set di dati vale 420 (numero di campioni in una settimana, campionando con periodo pari a 24 minuti). $S(t)$ è inizializzata, come detto, coi dati delle prime 3 settimane, perciò ogni punto della serie viene ricavato come:

$$S(t)_{1\le t\le s} = \frac{u(t) + u(t+s) + u(t+2s)}{3}$$

Per ottenere il valore di trend da assegnare come inizializzazione si ricorre nuovamente alle funzioni Matlab polyfit e polival.

Nella predizione di tipo short-range il predittore era ottenuto per un numero fisso di passi, quindi ogni punto predetto era calcolato avendo a disposizione i dati fino agli h passi precedenti. Nella predizione long-range invece tutti i punti predetti vengono ottenuti incrementando l'orizzonte temporale a partire dai valori aggiornati al tempo t (considerato l'ultimo istante in cui si hanno a disposizione i dati) delle tre serie temporali $L(t)$, $F(t)$ e $S(t)$ secondo la formula del predittore che, si ricorda, è la seguente:

$$\hat{u}(t+h|t) = L(t) + h \cdot F(t) + S(t+h-qs), \quad \text{con } q = \left[\frac{h}{s}\right]$$

4.4.1 Taratura dei parametri $\lambda_1, \lambda_2, \lambda_3$

Come nel caso della predizione short-range per la scelta dei parametri si utilizza come cifra di merito l'errore di predizione ad h passi.

$$errore = \frac{1}{N}\frac{\sum_{t=h}^{N}\left(u(t) - \hat{u}(t|t-h)\right)^2}{var(u)} \cdot 100$$

Come visto in precedenza per h crescente ci si aspetta di ottenere una maggiore lisciatura del segnale, corrispondente ad un valore sempre più basso di λ_1. I dati usati per inizializzazione e taratura ammontano quindi complessivamente a 8 settimane, come l'identificazione per il modello ARMA, in modo da mettere a confronto le predizioni dei diversi modelli. I parametri λ_i vengono ottenuti come descritto nel capitolo precedente calcolando la cifra di merito al variare dei valori di λ_i, calcolati fino alla terza cifra decimale.

Anche in questo caso si ricorre all'utilizzo di una saturazione, per evitare predizioni di traffico negative prive di significato fisico.

Nell'algoritmo di calcolo dei parametri λ_i i valori calcolati dal predittore vengono saturati a 0 prima del calcolo della cifra di merito in modo che i valori predetti negativi non influenzino il calcolo dell'errore.

In tabella sono mostrati i valori di λ_i ottenuti per cifre di merito con diversi orizzonti predittivi.

	$h = 1$	$h = 2$	$h = 5$	$h = 10$	$h = 100$
λ_1	0.469	0.492	0.485	0.042	0.014
λ_2	0	0	0	0	0
λ_3	0.255	0.286	0.297	0.284	0.289

Come si può notare λ_1 diminuisce all'aumentare dell'orizzonte predittivo h, mentre λ_3 ha una lieve crescita seguita da un assestamento; gli andamenti dei λ_i sono riportati in figura 4.22.

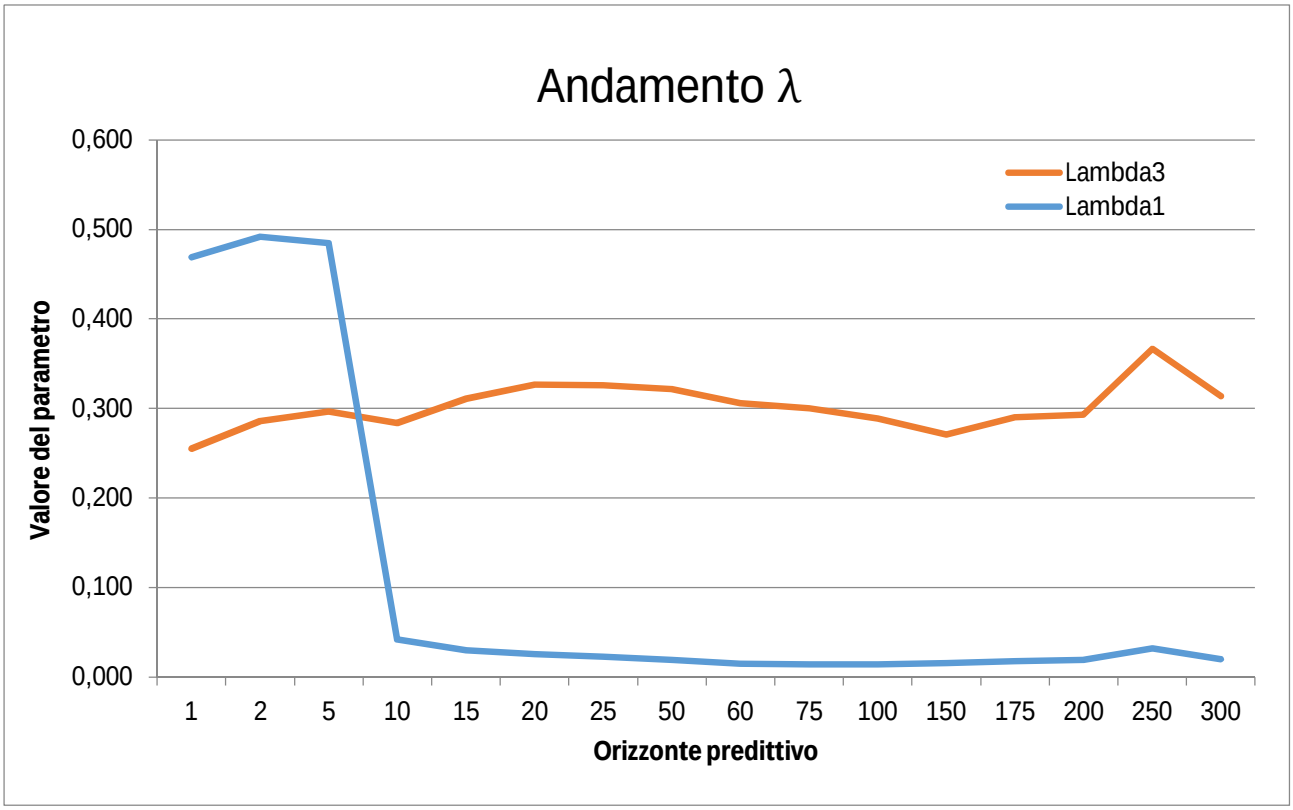


figura 4.22: andamento del variare dei parametri $\boldsymbol{\lambda_1}$ e $\boldsymbol{\lambda_3}$ all'aumentare dell'orizzonte predittivo considerato

In corrispondenza di $h = 10$ si ha per λ_1 un brusco calo dovuto a quanto che si è spiegato nella teoria di Holt-Winters nel capitolo 3: se λ_1 fosse rimasto ad un valore simile a quelli corrispondenti a orizzonti meno estesi, il predittore avrebbe assunto valori meno filtrati e con andamento più simile ai dati, ma con un ritardo di 10 passi, poiché il predittore assume per la componente $L(t)$ l'ultimo valore della serie lisciata; evidentemente 10 è un numero di passi sufficiente a far sì che i dati si discostino di molto rispetto ai valori del predittore, quindi la cifra di merito risulta minima per serie maggiormente filtrate, più vicine al valor medio dei dati.

Inoltre, se il segnale predetto perde di rumorosità, essendo la serie $L(t)$ maggiormente filtrata, sarà la serie $S(t)$ a dare un contributo più rumoroso, attraverso l'uso di un valore di λ_3 più vicino a 1 che conferisce maggior peso alla componente di riaggiornamento dei valori della serie: questa può essere la spiegazione dell'incremento del valore di λ_3, incremento proporzionato all'ordine di

grandezza della serie relativa a tale parametro, dal momento che la serie $S(t)$ contiene valori di circa un ordine di grandezza maggiore rispetto alla serie $L(t)$. Per questo una variazione anche minima del valore di λ_3 causerà delle variazioni sui valori finali del predittore.

Si mostra in figura 4.23 la differenza tra la stagionalità iniziale e quella calcolata automaticamente per effetto della terza equazione ricorsiva di Holt-Winters nel caso di λ_i ottenuti con una cifra di merito calcolata con $h = 100$.

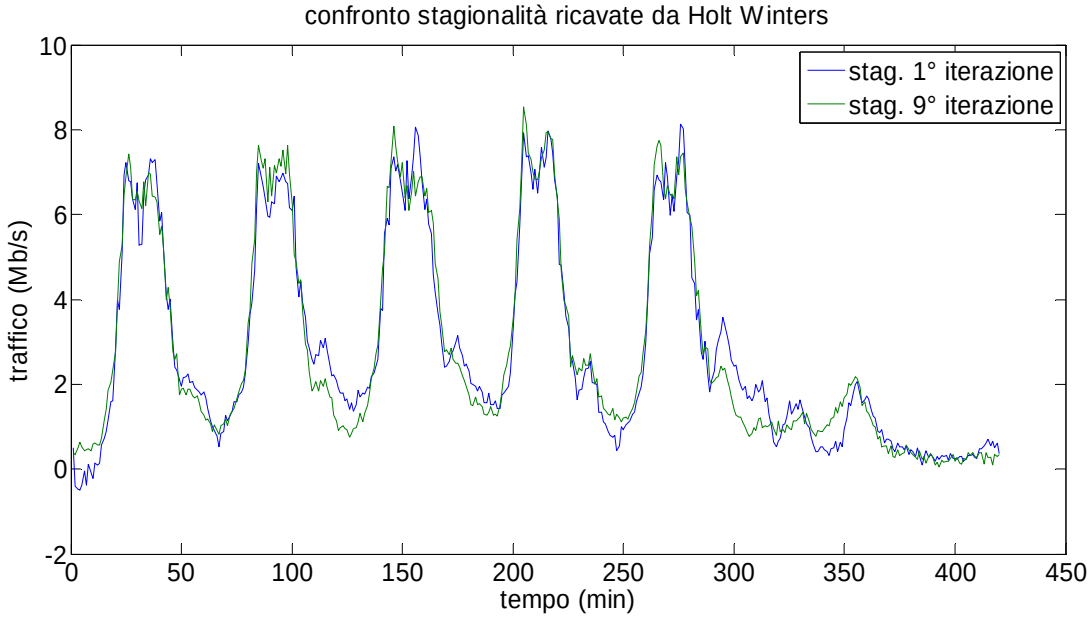


figura 4.23: confronto tra stagionalità ricavate dal modello di Holt-Winters ad iterazioni diverse, si può notare come l'aggiornamento ricorsivo porti a cambiamenti evidenti del segnale ricavato

4.4.2 Predizione del traffico in ingresso ad un router

Nella seguente tabella sono presentati i valori di fitting ricavati per gli orizzonti predittivi utilizzati per ricavare i parametri λ_i.

	h = 1	h = 2	h = 5	h = 10	h = 100
Fitting %	82.86%	82.84%	82.86%	85.27%	86.05%

Si ricorda che in questo caso i valori di fitting sono legati al valore di h con cui viene calcolata la cifra di merito in fase di taratura dei parametri, che non ha nulla a che fare con il valore di h usato per ottenere il segnale predetto. Infatti a differenza della predizione short-range la predizione è ottenuta a partire dall'ultimo istante del set di identificazione (t = 80640 min) incrementando ad ogni passo h di un'unità fino ad ottenere quattro settimane di predizione. Dai valori in tabella si nota che non ci siano grosse variazioni tra i valori di fitting, a differenza della predizione a breve termine: questo è dovuto al fatto che la predizione long-range dipende unicamente dagli ultimi valori calcolati delle serie $L(t)$ e $F(t)$ e l'ultima settimana della serie $S(t)$, quindi le quattro settimane predette risultano identiche tra loro (a meno del contributo della serie $F(t)$). Si noti come sia la serie $S(t)$ ad avere maggior incidenza sul segnale predetto poiché presenta valori di circa un ordine di grandezza superiore rispetto alle altre serie, il cui contributo alla predizione incide in maniera minore. In secondo luogo, osservando la tabella dei valori assunti dai parametri λ_i, si nota come il valore di λ_3 non vari molto, perciò le serie $S(t)$ calcolate per diversi h avranno valori e andamenti simili.

Nelle figure 4.24 e 4.25 sono riportati i risultati dei predittori a lungo termine con λ_i calcolati per h = 1 e h = 100.

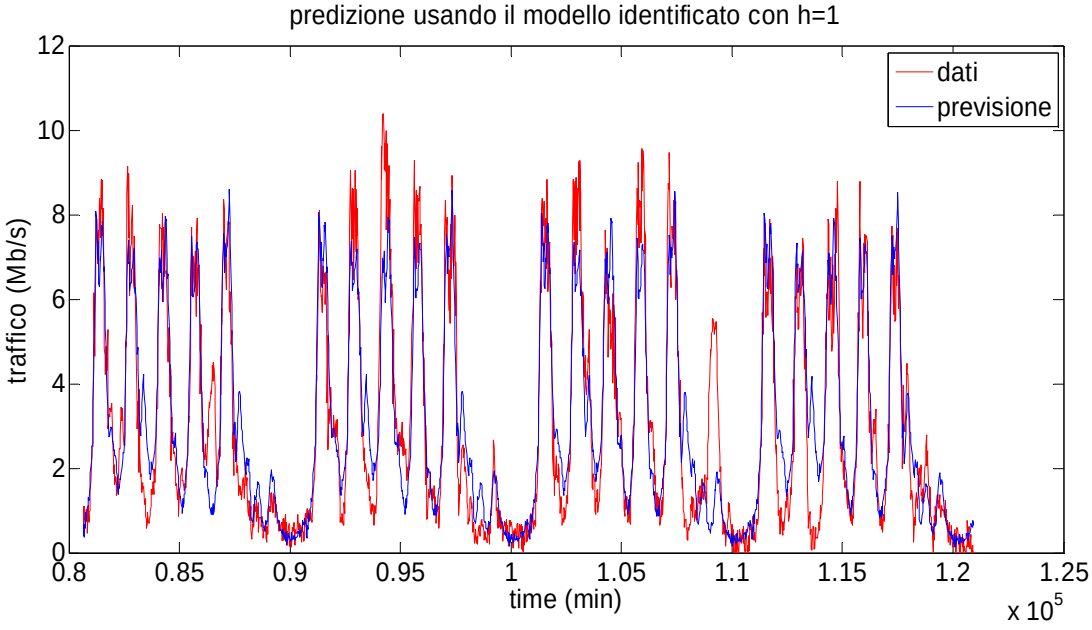


figura 4.24: confronto tra i dati (traffico) e la predizione del modello ricavata con un valore h di pari a 1

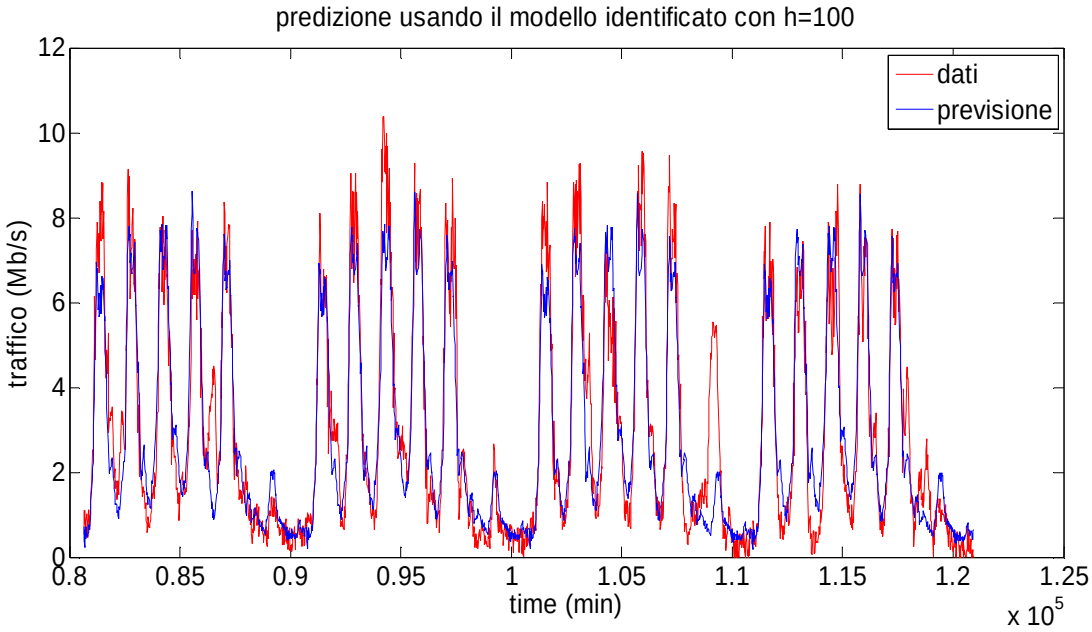


figura 4.25: confronto tra i dati (traffico) e la predizione del modello ricavata con un valore h di pari a 100

4.4.3 Predizione di utilizzo di CPU

Passando ora la previsione dell'utilizzo di CPU, si ricorda quanto ricavato nel capitolo precedente: se, utilizzando un modello di Holt-Winters anche per il segnale y, si hanno valori dei parametri λ_i uguali, o per lo meno molto simili, a quelli ricavati per il traffico è possibile calcolare un coefficiente K_i^O per ciascuna serie temporale serie $L(t), F(t)$e $S(t)$ e stabilire quindi una relazione tra le singole componenti dei segnali. In caso contrario occorre invece ricavare una relazione considerando traffico e CPU nella loro totalità, avendo prestazioni inferiori.

Vengono quindi usati i dati di CPU, divisi in un set di inizializzazione e uno per la taratura dei parametri di grandezza uguali a quelli del traffico, per poter ricavare i coefficienti λ_i e stabilire quale relazione si debba utilizzare per ricavare la previsione per l'utilizzo di CPU.

In tabella sono mostrati i valori dei λ_i risultanti per cifre di merito calcolate per gli stessi valori di h delle tabelle precedenti.

	h = 1	h = 2	h = 5	h = 10	h = 100
$\boldsymbol{\lambda_1}$	0.603	0.636	0.644	0.309	0.022
$\boldsymbol{\lambda_2}$	0	0	0	0	0
$\boldsymbol{\lambda_3}$	0.283	0.324	0.375	0.229	0.296

Come si nota i λ_i risultano diversi da quelli di traffico, soprattutto λ_1 in cui, oltre alla differenza dei valori, si nota che il calo repentino dei valori assunti avviene per un valore di h maggiore di 10, mentre per il traffico avveniva tra $h = 5$ e $h = 10$.

Dal momento che i dati relativi alla CPU sono molto meno rumorosi rispetto a quelli di traffico, ci si aspetta che il filtraggio, almeno nel calcolo dell'errore ad orizzonti bassi, sia minore rispetto a quello subito dai dati di traffico: confrontando i valori delle tabelle di traffico e di CPU si nota come questa supposizione sia confermata.

A questo punto, contrariamente a quanto fatto per la predizione short-range, si ottiene la previsione di CPU calcolando il rapporto tra il segnale di traffico e quello di CPU complessivi. Il coefficiente che mette in relazione traffico e CPU viene quindi ottenuto come:

$$K^O = \sum_{i=1}^{N} y_i u_i \Big/ \sum_{i=1}^{N} {u_i}^2 = 1.00283[\%/(\mathrm{Mb/s})]$$

Moltiplicando per K^O i valori delle quattro settimane di traffico predette col metodo di Holt-Winters, si ottiene la predizione long-range di CPU; i risultati lungo le 4 settimane di predizione sono mostrati in figura 4.26 e 4.27, ricavati rispettivamente con la cifra di merito per $h = 1$ e $h = 100$.

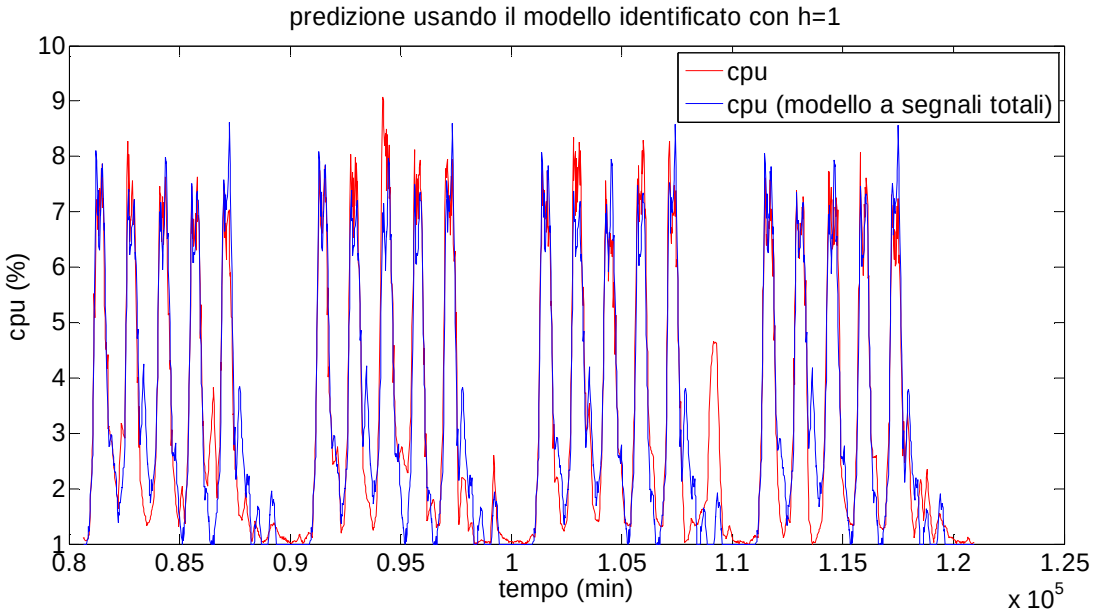


figura 4.26: confronto tra i dati (utilizzo di CPU) e la predizione del modello ricavata con un valore h di pari a 1

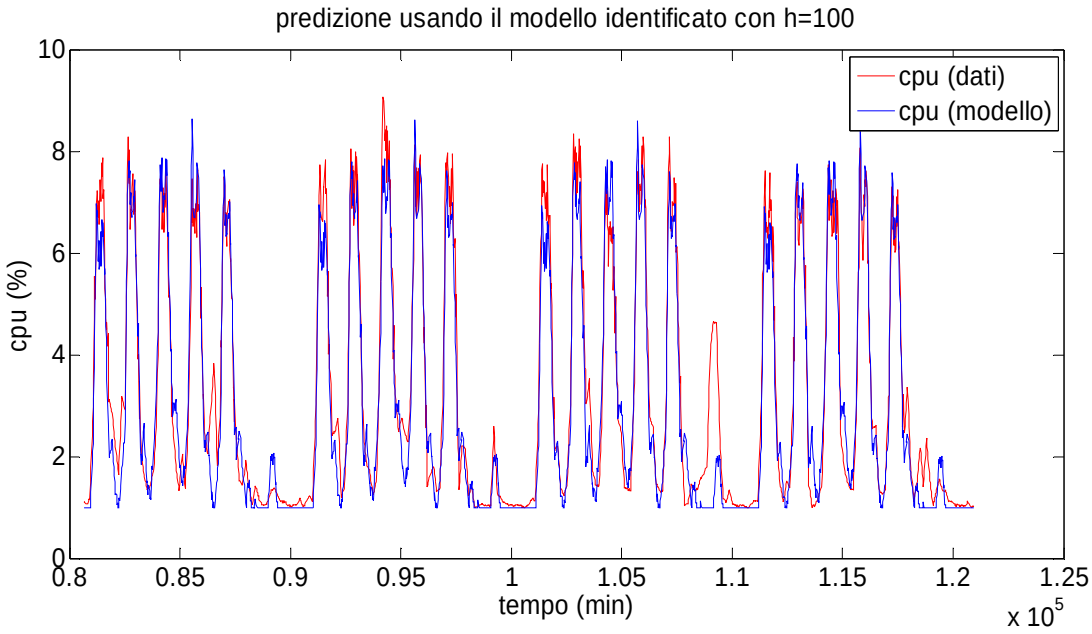


figura 4.27: confronto tra i dati (utilizzo di CPU) e la predizione del modello ricavata con un valore h di pari a 100

I valori di Fitting ottenuti sono mostrati nella seguente tabella, ricordando che ad ogni valore di h corrispondo i valori di λ_i visti in precedenza per i dati di traffico.

	h = 1	h = 2	h = 5	h = 10	h = 100
Fitting %	89.07%	89.06%	89.10%	92.13%	92.04%

Anche in questo caso si nota come i valori di fitting non siano molto diversi tra loro al variare di h, attestandosi su valori soddisfacenti considerando il fatto che si sta cercando di predire l'andamento del segnale su un orizzonte temporale elevato, pari a 4 settimane.

Tutti i valori di fitting per traffico e CPU visti finora sono relativi ai segnali di traffico e al processore del router Adderley, nell'appendice 4.3 sono esposti i risultati ottenuti anche per i router Hancock, Frisell e Monk.

4.4.4 Confronto tra modello ARMA e metodo di Holt-Winters

Per valutare le prestazioni del metodo di Holt-Winters, si confrontano i risultati con quelli ottenuti dal modello ARMA. Si ricorda che la predizione del modello ARMA coincide sostanzialmente con la parte deterministica del segnale: tale confronto risulta quindi utile in quanto, se il modello dovesse mostrare prestazioni inferiori, il suo utilizzo non risulterebbe vantaggioso, dal momento che si utilizzerebbe un modello al posto di ricavare più semplicemente una stagionalità senza ricavarne un vantaggio in termini di prestazioni. Viceversa, se le prestazioni del modello di Holt-Winters risultassero migliori, sarebbe conveniente utilizzare tale modello leggermente più complesso dal momento che si avrebbero effettivamente dei vantaggi in predizione.

Per quel che riguarda il traffico, la predizione long-range ricavata con l'ARMA raggiunge un fitting pari al 78.87%, a fronte di un fitting di Holt-Winters che (per una taratura effettuata con h = 100) vale 86.05%.

Da questo risultato si capisce che il metodo di Holt-Winters ottenga una predizione migliore dovuta alla sua capacità di aggiornare settimanalmente la stagionalità; le prestazioni risultano quindi legate ai valori di λ_i che regolano tale aggiornamento. Nel caso del router Adderley non si notano particolari differenze nelle prestazioni al variare di h; nell'appendice 4.3 si mostra come a volte sia necessario usare h relativamente alti (10,100), poiché per avere buone prestazioni è preferibile avere un valore di λ_1 basso, così da avere un filtraggio superiore. Un filtraggio "pesante" garantisce buone predizioni a lungo termine, poiché l'ultimo valore della serie $L(t)$ assume una valore più vicino alla media del segnale e non all'ultimo valore dei dati, in caso contrario la componente additiva, rappresentata dall'ultimo valore di $L(t)$ nella formula del predittore, potrebbe portare a sensibili scostamenti tra i dati e la predizione.

Si considera poi la predizione di CPU, che nel caso del modello ARMA viene ottenuta con 3 coefficienti che mettono in relazione i 3 contributi e cioè componente stocastica, trend e stagionalità, mentre nel caso di Holt-Winters viene ottenuta con un solo coefficiente che lega traffico e CPU considerando i segnali totali. Il fitting della predizione long-range di CPU utilizzando il modello ARMA è pari al 82.04% mentre usando il metodo di Holt-Winters si ottiene, nel caso in cui h = 100 un fitting del 92.04%. Anche in questo caso i risultati del metodo Holt-Winters superano quelli del modello ARMA facendo comprendere l'importanza di questo metodo che, coi dati usati per lo svolgimento di questa tesi, ha sempre dato performance vicine o addirittura superiori agli altri modelli meglio conosciuti. Uscendo dal contesto considerato (cioè la rete Acme) sarebbe importante verificare le prestazioni del metodo quando i segnali presentino outliers che, come spiegato in [22], costituiscono una problematica complessa in fase di taratura dei parametri.

4.4.5 Conclusioni relative al modello di Holt-Winters

Nell'ambito della predizione long range il metodo di Holt-Winters è risultato essere il più performante tra i modelli presi in esame, sia per la predizione di traffico che per la predizione di allocazione di CPU. Per ottenere tali risultati sono stati calcolati i valori delle costanti di lisciamento λ_i per diverse cifre di merito variando il valore di h (orizzonte predittivo usato nel calcolo dell'errore in fase di taratura dei parametri) a partire dal segnale di traffico.

Per ogni tripla di λ_i sono state calcolate le tre serie temporali $L(t)$, $F(t)$ e $S(t)$ da cui è stato ricavato il predittore long-range previsto dalla teoria del metodo Holt-Winters, in seguito sono stati calcolati i valori di fitting (un valore per ogni h) utilizzando i dati relativi alle ultime 4 settimane a disposizione.

Per ottenere la predizione di utilizzo di CPU a partire da quella di traffico è stato calcolato un coefficiente proporzionale statico attraverso il metodo dei minimi quadrati usando i dati delle 8 settimane di identificazione. Moltiplicando le predizioni di traffico (ottenute per ogni h) per il coefficiente trovato si è ottenuta la predizione long-range dell'utilizzo CPU. Le predizioni di CPU sono state confrontate coi dati nelle 4 settimane successive a quelle usate per l'identificazione ottenendo indicazioni sulle performance del modello. Infine i valori di fitting sono stati comparati con quelli ottenuti usando il modello ARMA. Dalla comparativa il metodo di Holt-Winters risulta superiore nelle predizioni a lungo termine, ottenendo risultati più vicini ai dati reali.

CONCLUSIONI

Nel corso di questo capitolo si è affrontato il problema della predizione long-range per traffico e utilizzo di CPU ricorrendo a diversi modelli per poter ottenere dei risultati soddisfacenti.

Per prima cosa si è lavorato sul segnale a disposizione: il basso numero di sample con il quale esso era stato campionato non permetteva di poter arrivare ad una previsione realistica delle dinamiche proprie di traffico e CPU, per cui si è ricorso ad un artificio per ottenere un campionamento sufficientemente fitto e un andamento che rispecchiasse maggiormente la realtà.

Ottenuto un set di dati su cui poter lavorare, si è scelto come dividerlo: dal momento che si voleva predire per un orizzonte temporale elevato, si è scelto di usare le ultime 4 settimane per verificare la bontà della predizione, e utilizzare le prime 8 per identificare i modelli considerati.

Successivamente si sono considerati 3 diversi modelli, utilizzati per la predizione di tipo short range nel capitolo precedente.

Il modello ARMA considera il segnale nelle sue singole componenti, andando a fare previsione della sola componente stocastica per poi sommare in un secondo momento le componenti deterministiche. La previsione su un orizzonte predittivo elevato di un modello di questo genere coincide, dopo un breve transitorio pari a circa 13 ore, alla media del segnale. Dal momento che la media del segnale è pari a zero, la predizione coincide sostanzialmente con la stagionalità e il trend del segnale calcolati dal set di identificazione (pari a 8 settimane di dati). I risultati in termini di fitting sono soddisfacenti, anche se, volendo utilizzare una previsione di questo tipo, è più conveniente disinteressarsi della parte stocastica del segnale concentrandosi unicamente sulla parte deterministica, dal momento che in questo modo le performance ottenute risultano sostanzialmente identiche, a fronte di un facilità e velocità di impiego maggiori.

Tale modello può essere usato come benchmark per i modelli successivi: dal momento che la predizione coincide con la stagionalità che può essere facilmente ricavata, se gli altri modelli avessero ottenuto prestazioni inferiori non sarebbe stato conveniente utilizzarli, dal momento che un aumento della complessità (dovuto all'uso del modello) non avrebbe portato ad un incremento delle prestazioni. Viceversa, un aumento delle prestazioni rispetto a quelle ottenute con la stagionalità avrebbe portato a ritenere accettabile l'aumento della complessità derivante dall'uso di un modello.

Se quest'ultimi avessero ottenuto prestazioni simili o peggiori, il loro utilizzo non sarebbe risultato conveniente se paragonato all'uso della stagionalità del segnale, a causa della sua immediatezza.

Ciò si è verificato nell'implementazione del modello ARIMA, che è risultato inadeguato per il tipo di predizione considerato: dal momento che, una volta differenziato il segnale, il modello si comporta come un ARMA, il predittore tenderà nuovamente alla media del segnale, che si è verificata essere anche in questo caso pari a zero. Non considerando però solamente la parte stocastica del segnale in questo caso la predizione risulta pessima: una volta ricavata la predizione del segnale a partire da quella del segnale differenziato, si può notare come la stagionalità venga completamente tagliata, portando a performance pessime del segnale.

Il modello di Holt-Winters, invece, è risultato il più performante tra quelli considerati. Dal momento che si avevano un numero di settimane sufficienti, è stato possibile in questo caso dividere il set di identificazione in un primo (di 3 settimane) per potere inizializzare le serie $S(t)$, $F(t)$ e $L(t)$ e di un secondo (pari a 5 settimane) per poter tarare i parametri λ_i del modello. Per poter far ciò, si è impiegata nuovamente una cifra di merito ricavata a partire dall'errore di predizione ad h passi, variando tale valore da un minimo di un passo ad un massimo di 420 (pari al numero di campioni presenti in una settimana). Si è potuto osservare come in generale le prestazioni siano circa simili al variare del valore di h considerato, anche se per poter essere sicuri di ottenere la massima prestazione conviene utilizzare un valore pari o superiore a 10 (a conferma del fatto che si sta considerando una predizione long-range. In questo modo la serie $L(t)$ risulta essere maggiormente lisciata, garantendo una predizione, per tale componente, più vicino al suo valor medio. In caso contrario, qualora l'ultimo valore della serie risultasse molto lontano dalla media, si avrebbe una predizione più imprecisa, con una conseguente performance non ottimale. Da notare come il modello di Holt-Winters ottenga prestazioni migliori del modello ARMA, pur utilizzando, per la conversione da traffico a CPU, un unico coefficiente.

Ciò porta quindi a concludere che il modello migliore, nell'ambito della predizione di tipo long-range, risulta essere il modello di Holt-Winters.

5

Conclusioni e sviluppi futuri

Nel corso di questa tesi si sono affrontate problematiche legate alla predizione a scopo di allocazione ottima di risorse in un router all'interno di una rete di telecomunicazione.

Tutta la trattazione presentata utilizza dati provenienti da una rete di medie dimensioni forniti da British Telecom e potrà essere adattata e implementata su altre reti, per capire come poter generalizzare i risultati ottenuti.

Gli obiettivi che ci si era prefissati erano il costruire un modello in grado di catturare la relazione tra il traffico proveniente da una rete ed entrante in router e la relativa occupazione di risorse e la predizione di tali segnali per determinati orizzonti predittivi.

Per prima cosa si è identificata una relazione che legasse le grandezze considerate: si è potuto verificare come tale legame possa essere ben modellizzato attraverso l'uso di una semplice relazione statica e lineare nella forma $y = Ku$, in cui u rappresenta il flusso di dati entrante e y la quantità di risorse occupata. Un modello di questo genere può essere ricavato agevolmente attraverso la minimizzazione di una cifra di merito, andando ad identificare il valore ottimale K^o. Si sono considerati due diversi approcci: un primo che stabilisce una relazione per ciascuna delle componenti che formano i segnali (trend, stagionalità, parte stocastica), un secondo che considera i segnali complessivi. Si è verificato come il primo approccio sia più vantaggioso tra i due, dal momento che utilizza un numero di gradi di libertà maggiori, e che richiede solamente una scomposizione del segnale nelle sue componenti, operazione che può essere svolta facilmente utilizzando funzioni Matlab.

L'innovazione fornita da tale modello consiste nel poter determinare con buona approssimazione, nota la quantità di traffico che interessa il router considerato, il relativo consumo di risorse: tale risultato può essere impiegato all'interno di attività come il capacity planning e il load balancing descritte all'interno di questa tesi, potendo determinare in modo immediato se un router stia presentando carichi di lavoro che generano congestionamenti oppure sia utilizzata una quantità di risorse accettabile.

Il secondo argomento trattato riguarda la predizione di allocazione delle risorse: utilizzando quanto ricavato precedentemente, il problema non consiste più nel prevedere i valori futuri di CPU a partire da quelli passati, ma nel prevedere l'andamento del traffico entrante nel router e successivamente sfruttare la relazione statica lineare per ottenere la previsione di CPU.

Considerando una previsione con un orizzonte predittivo breve (short-range), si sviluppano tre modelli:

- Un modello ARMA: nonostante il traffico presenti trend e stagionalità, è possibile ricondursi ad un segnale stazionario ricavando le singole componenti e considerando per il modello ARMA la sola componente stocastica. In questo modo è possibile utilizzare tale modello per poter far previsione di traffico e successivamente ricondursi all'utilizzo di CPU attraverso l'uso di un coefficiente per ogni componente.
- Un modello ARIMA, preso in considerazione in quanto, rispetto al modello ARMA, non presenta il bisogno di una fase preliminare in cui scomporre il segnale: in questo modo i dati provenienti dal database possono essere utilizzati senza bisogno di modifiche preliminari.
- Un modello di Holt-Winters: tale metodo risulta impiegato per la previsione in campo economico, ma non ancora utilizzato nel campo delle telecomunicazioni. Il vantaggio rispetto ai modelli precedenti consiste nella capacità del modello di poter riaggiornare automaticamente gli andamenti di trend e stagionalità, cosa che deve essere fatta manualmente nel modello ARMA.

Dai risultati ottenuti nel capitolo 3 si può osservare come i modelli abbiano performance differenti: mentre il modello ARMA e il metodo di Holt-Winters raggiungono percentuali di fitting simili, il modello ARIMA porta a prestazioni inferiori, seppur soddisfacenti. Da ciò si può concludere che, relativamente ai dati utilizzati in questa tesi, è preferibile utilizzare un modello ARMA o in alternativa di Holt-Winters per una predizione di tipo short-range.

Il contributo innovativo fornito da questi risultati può essere utilizzato nell'ambito del load balancing: poter prevedere lungo un orizzonte temporale di qualche minuto è sufficiente per poter allocare il carico di lavoro di una server farm in modo efficiente tra tutti i server a disposizione, andando a suddividere la mole di lavoro non in base al carico presente, ma a quello previsto nell'immediato futuro.

Ponendo successivamente l'attenzione ad un orizzonte temporale elevato si è trattato il problema della predizione long-range. Come già affermato precedentemente, l'applicazione che più necessita di tale predizione per l'utilizzo di CPU è il capacity planning: in base ai livelli di occupazione dei vari device considerati in una rete, si deve essere in grado di decidere dove andare a potenziare le componenti hardware già presenti. Si capisce quindi quale importanza rivesta l'avere una predizione affidabile.

Per poter sviluppare dei metodi adeguati, per prima cosa si è dovuto considerare quale potesse essere la frequenza di campionamento adeguata: dal momento che non si ha bisogno di conoscere l'andamento puntuale del segnale, come per la previsione short-range, non si necessita di campionare con frequenza pari ad un minuto. Si è scelto di considerare quindi un periodo di campionamento pari a 24 minuti, in modo da avere 60 campioni per giorno ed essere in grado di descrivere con sufficiente fedeltà l'andamento giornaliero dei segnali.

Si è potuto verificare come il modello ARMA non fornisca buoni risultati per la previsione della parte stocastica del segnale, dal momento che l'uscita del predittore tende a convergere alla media (pari a zero) dopo un numero di passi limitato. La predizione coincide quindi con le componenti deterministiche del segnale.

Il metodo di Holt-Winters ottiene invece risultati migliori: attraverso l'aggiornamento ricorsivo della stagionalità e un adeguato filtraggio della serie $L(t)$ si ottengono prestazioni in termini di fitting migliori rispetto all'uso della sola stagionalità, giungendo alla conclusione che lo sviluppo di tale modello sia giustificato dalle proprie performance.

In questo modo si è in grado di fornire quindi uno strumento valido per l'attività di capacity planning.

Un primo sviluppo futuro è la comparativa tra due diversi capacity plan applicati alla medesima rete, di cui il primo ottenuto implementando i risultati di questa tesi, il secondo utilizzando le tecniche presentate nello stato nell'arte, per verificare i vantaggi che si potranno trarre da quanto sviluppato.

Un secondo possibile studio a seguito di questa trattazione è l'applicazione dei risultati ottenuti considerando diverse reti di medie e grandi dimensioni, in modo da poter generalizzare i risultati ottenuti a reti differenti da quella presa in esame.

Un terzo approfondimento riguarda lo studio della relazione traffico-CPU in condizioni di sovraccarico del router considerato, per poter determinare un modello avente condizioni di validità differenti da quelle considerate, da poter impiegare in casi di congestione della rete.

Altro sviluppo potrebbe essere l'implementazione di algoritmi ricorsivi in grado di adattare, in caso di necessità, il modello identificato qualora si dovesse avere una variazione sostanziale del traffico (dovuta ad esempio a modifiche nella topologia della rete o ad un progressivo incremento/cambiamento del traffico in entrata al device) tale da richiedere una modifica del modello.

Infine un ulteriore ambito di sviluppo riguarda la previsione di traffico e CPU di un'intera rete e non del singolo router: considerando le relazioni che intercorrono tra i traffici in entrata ai vari router (dal momento che il traffico uscente da ciascun router va a contribuire a formare il traffico in entrata degli altri device della rete) sarebbe possibile considerare solamente i traffici in ingresso ad un sottoinsieme limitato di router e successivamente, mediante la conoscenza della topologia della rete, ricostruire i traffici su ogni link delle rete e quindi ricondursi ad una previsione di CPU.

Appendice 2.1

L'argomento trattato in questa appendice riguarda i comandi da impartire in ambiente RRDtool per creare i database per traffico e CPU da noi utilizzati all'interno di questa tesi.

Vengono mostrate le linee di codice utilizzate per creare tali database.

CREAZIONE DATABASE TRAFFICO:

```
rrdtool create IF-MIB::ifInOctets.1.rrd
    --start 1367877600
            step=60
    DS:traffico:COUNTER:600:U:U
    RRA:AVERAGE:0.5:1:40320
            RRA:AVERAGE:0.5:6:672
            RRA:AVERAGE:0.5:24:732
            RRA:AVERAGE:0.5:144:1460
```

```
rrdtool create CISCO-PROCESS-MIB::cpmCPUTotal1min.1.rrd
    --start 1367877600
            step=60
    DS:CPU:GAUGE:600:U:U
    RRA:AVERAGE:0.5:1:40320
            RRA:AVERAGE:0.5:6:672
            RRA:AVERAGE:0.5:24:732
            RRA:AVERAGE:0.5:144:1460
```

In cui:

--Start determina l'istante di partenza in cui si comincia a raccogliere dati. L'istante temporale deve essere fornito in Epoch Time. Per definizione l'Epoch Time è espresso in secondi a partire dal 1 gennaio 1970 a mezzanotte.

--Step indica l'intervallo, in secondi, tra un campione e il successivo (nel primo AVERAGE considerato; nei successivi, dal momento che i dati vengono mediati, l'intervallo sarà equivalente ad un multiplo dello step).

--DS (data source) indica l'elemento da cui vengono forniti i dati, nel nostro caso traffico e utilizzo di CPU.

--DATASOURCE:heartbeat:min:max il primo termine specifica che tipo di data source si vuole selezionare (nel nostro caso COUNTER per traffico e GAUGE per CPU). Heartbeat, min e max sono parametri del data source da impostare, nello specifico:

-heartbeat: definisce il numero massimo di secondi che possono passare tra il salvataggio di un dato e il successivo, oltre il quale il valore salvato è di default "unknown". In questo caso è impostato a 600 (valore di default)

-min e max: definiscono i valori che i dati possono assumere; nel nostro caso, non essendoci limiti fisici al massimo valore del traffico, si imposta il valore ad U (unlimited). Anche per la CPU non viene posto limite, aspettandosi comunque che i valori non superino mai il 100%.

--RRA:CF:xff:steps:rows definiscono gli archivi Round Robin (RRA). Nel nostro caso sia per traffico che per CPU si usano solamente consolidation function di tipo AVERAGE. Per quanto riguarda i campi propri della consolidation function:

-xff: l'x-file factor definisce il numero massimo di dati che possono essere unknown sul totale dei dati; viene definitivo tra 0 e 1. E' stato scelto il valore di default 0.5.

-steps: definisce il numero di steps tra un valore e il successivo in un archivio. Se pari a 1, non si ha l'operazione di mediazione, e ogni dato nell'archivio è pari a ciò che fornisce il data source; se maggiore di 1, si calcola la media utilizzando un numero di sample pari al valore di steps.

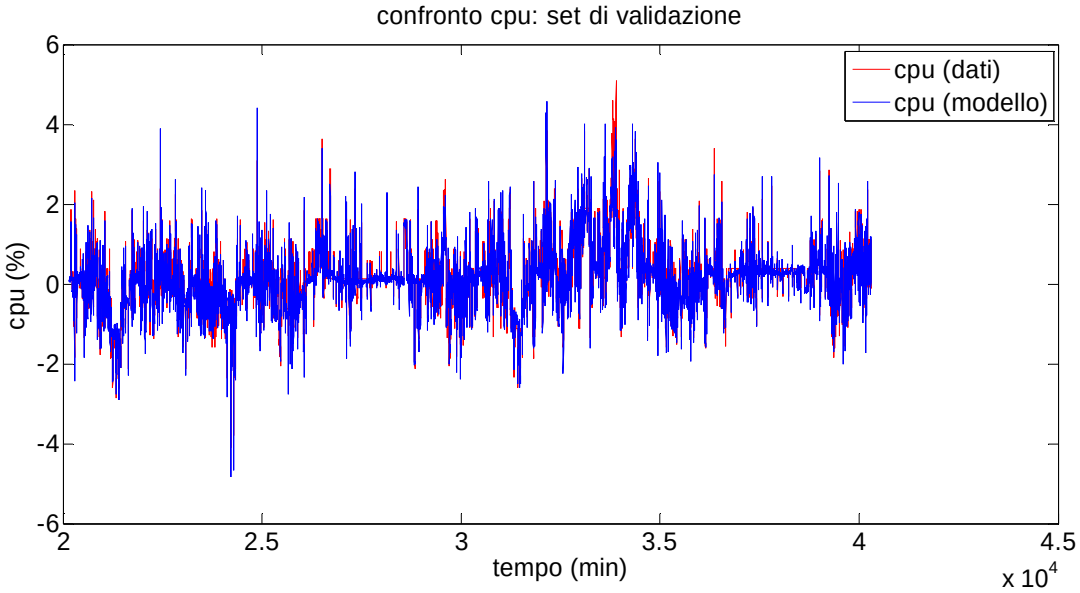
-rows: indica il numero di dati all'interno di un archivio RRA. Quando il primo archivio risulta completamente riempito, comincia l'operazione di mediazione, che porta a salvare i risultati nell'archivio successivo e libera spazio nell'archivio iniziale.

Appendice 2.2

Si riportano nello specifico i risultati relativi al router Adderley nei casi 2 (2 settimane per l'identificazione e 2 settimane per la validazione) e 3 (1 settimana per l'identificazione e 3 settimane per la validazione).

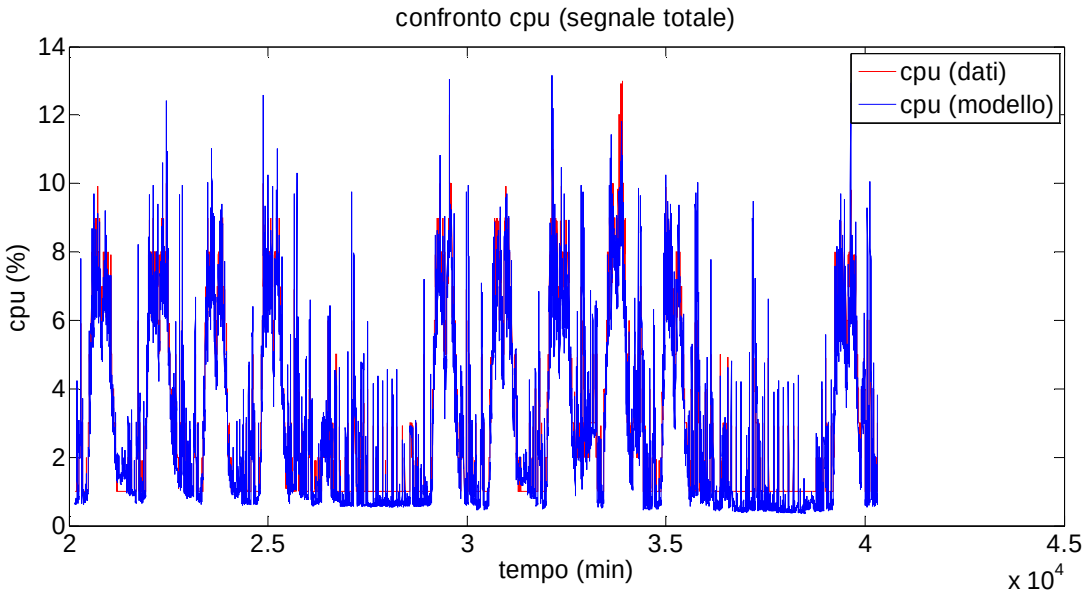
CASO 2:

Considerando i segnali depurati da trend e stagionalità, per quanto riguarda il confronto tra l'uscita del modello e i dati del set di validazione:



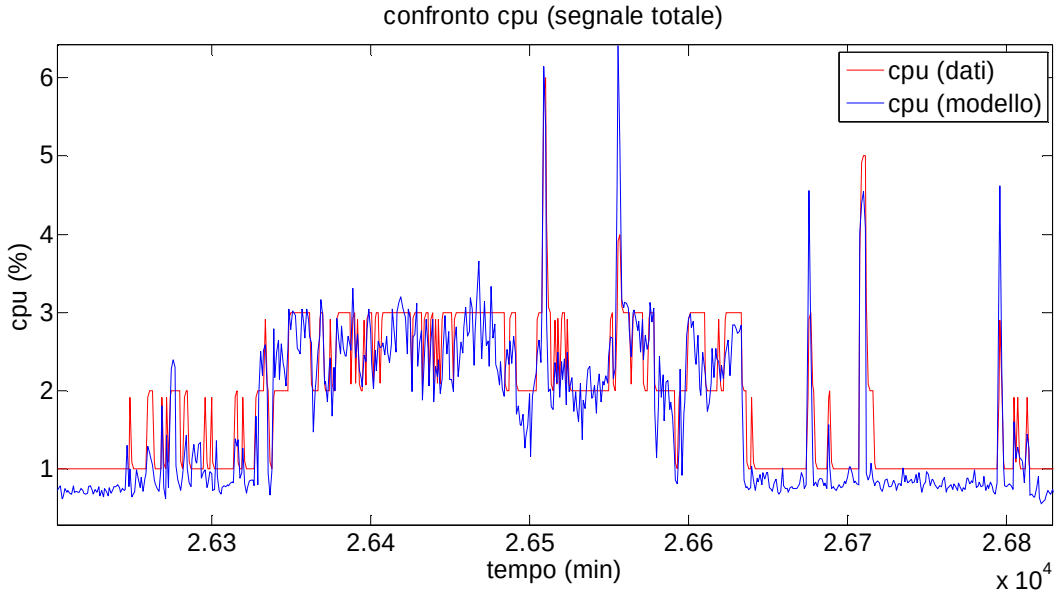
Confronto tra la componente stocastica del segnale ricavata dai dati e l'uscita del modello

Passando al segnale totale, confrontando l'uscita con i dati del set di validazione:



Confronto tra il l'utilizzo di CPU del set di dati e l'uscita del modello (segnale totale)

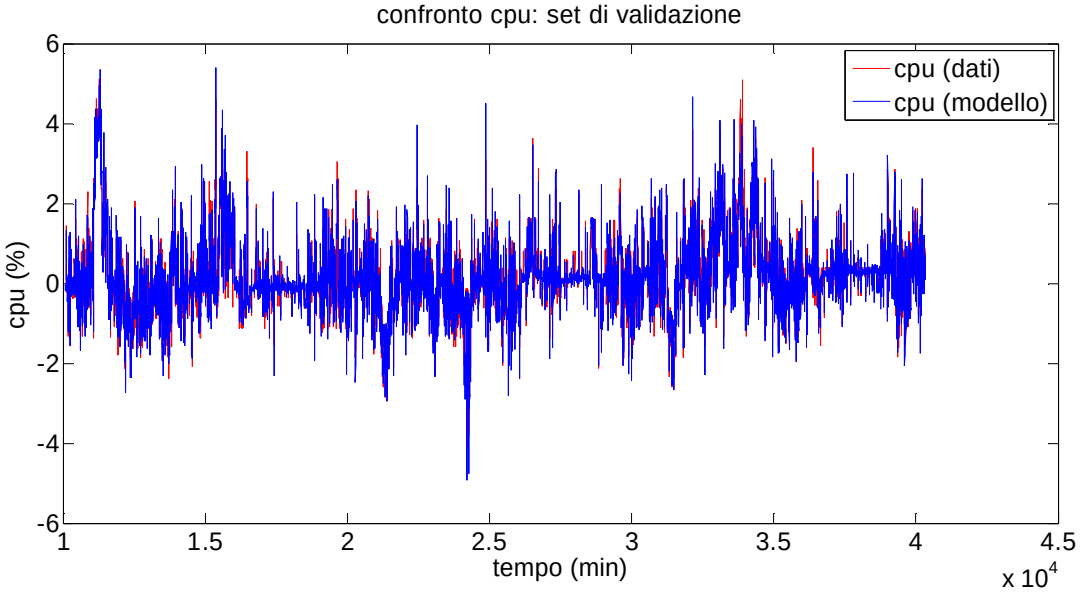
Osservando nel particolare:



Particolare del confronto tra modello e dati: si nota come ad ogni picco di CPU dei dati corrisponda un picco dell'uscita del modello

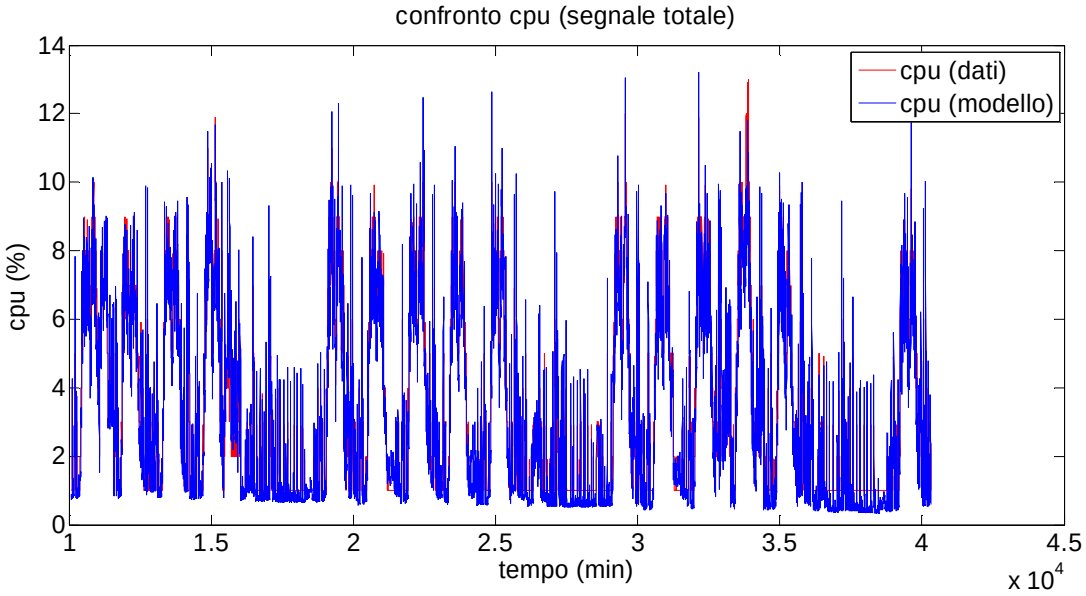
CASO 3:

Considerando nuovamente i segnali depurati da trend e stagionalità, per quanto riguarda il confronto tra l'uscita del modello e i dati del set validazione:



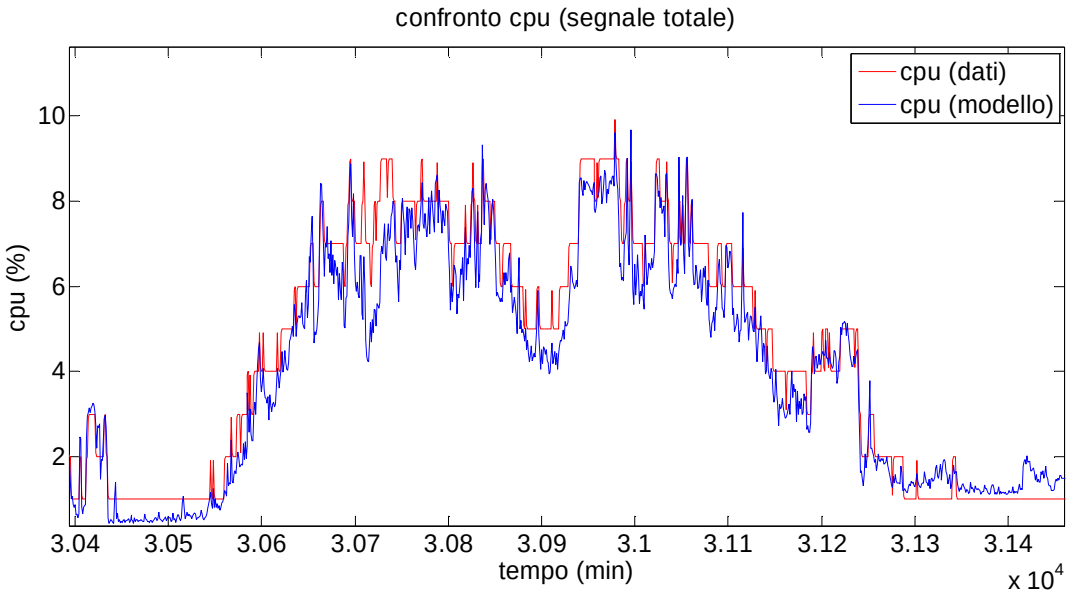
Confronto tra la componente stocastica del segnale ricavata dai dati e l'uscita del modello

Passando al segnale totale:



Confronto tra il l'utilizzo di CPUdel set di dati e l'uscita del modello (segnale totale)

Osservazioni più nel particolare:



Particolare del confronto tra modello e dati: si nota come ad ogni picco di CPU dei dati corrisponda un picco dell'uscita del modello

Appendice 2.3

Di seguito si riportano i risultati relativi all'identificazione dei parametri K_i^O che caratterizzano la relazione traffico-utilizzo di CPU per tutti i router della rete considerati. Nel capitolo 2 si è trattato il caso in cui, delle 4 settimane a disposizione, si usavano le prime 3 per fare identificazione e l'ultima per validare; in questa appendice vengono mostrati i risultati considerando 2 settimane per l'identificazione in un caso (prime due tabelle), una sola settimana per l'identificazione e tre per la validazione nell'altro (terza e quarta tabella).

CASO 2: 2 SETTIMANE PER L'IDENTIFICAZIONE-2 SETTIMANE PER LA VALIDAZIONE

	Num. interfacce	K_c^O (componente stocastica) [%/Mb/s]	K_s^O (stagionale) [%/Mb/s]	K_t^O (trend) [%/Mb/s]	K^O (segnale totale) [%/Mb/s]
Adderley	2	0.584	0.866	1.060	0.925
Frisell	2	1.228	1.657	1.632	1.585
Hancock	3	8.039	12.318	14.918	13.254
Garbarek	5	K1: 3.720 K2: 3.937	4.772	6.083	-
Monk	6	1.683	2.066	3.671	2.531

	Fitting (componente stocastica)	Fitting segnale totale (con 3 coeff)	Fitting segnale totale (con 1 coeff)	Fitting segnale totale (a partire dai dati)
Adderley	86.60%	93.34%	87.71%	98.82%
Frisell	83.18%	94.54%	93.88%	95.16%
Hancock	77.14%	94.48%	88.19%	97.04%
Garbarek	72.91%	86.57%	-	90.35%
Monk	72.61%	96.27%	80.72%	97.95%

CASO 3: 1 SETTIMANA PER L'IDENTIFICAZIONE-3 SETTIMANE PER LA VALIDAZIONE

	Num. interfacce	K_c^O (componente stocastica) [%/Mb/s]	K_s^O (stagionale) [%/Mb/s]	K_t^O (trend) [%/Mb/s]	K^O(segnale totale) [%/Mb/s]
Adderley	2	0.596	0.866	1.045	0.936
Frisell	2	1.199	1.657	1.622	1.578
Hancock	3	8.022	12.318	14.819	12.783
Garbarek	5	K1: 3.688 K2: 6.059	4.772	5.976	-
Monk	6	1.685	2.066	3.546	2.553

	Fitting (componente stocastica)	Fitting segnale totale (con 3 coeff)	Fitting segnale totale (con 1 coeff)	Fitting segnale totale (a partire dai dati)
Adderley	88.38%	93.39%	87.31%	98.83%
Frisell	87.80%	94.66%	93.96%	95.26%
Hancock	75.16%	94.96%	90.19%	97.13%
Garbarek	70.34%	85.95%	-	90.34%
Monk	75.43%	95.17%	80.37%	97.58%

Appendice 3.1

Cifre di merito relative alla trattazione del modello ARMA per la predizione short-range

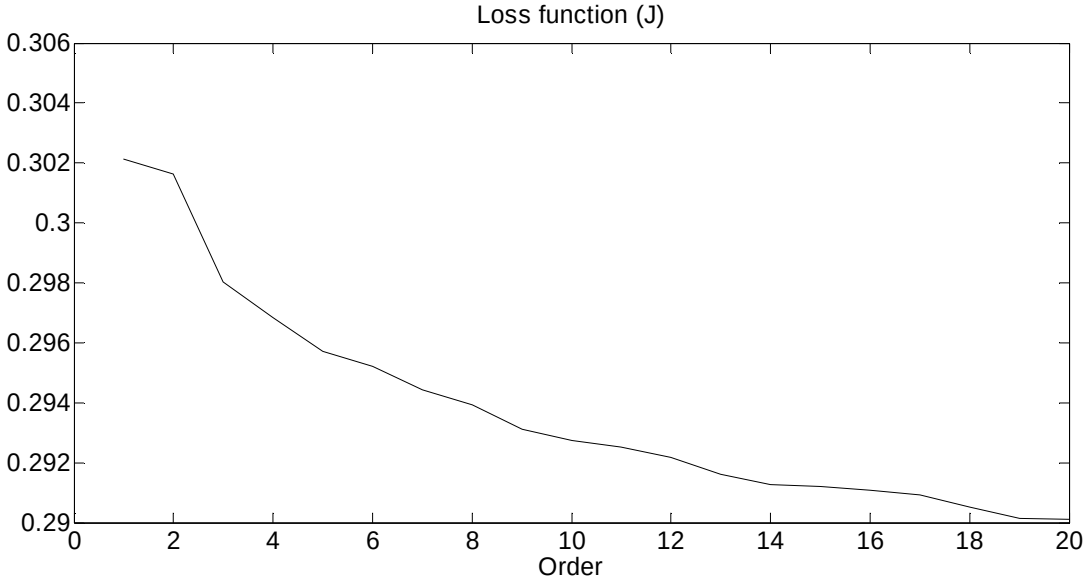
Di seguito si riportano i grafici rappresentanti le cifre di merito considerate nella modellizzazione del modello ARMA per la predizione short-range di traffico e CPU e non inserite direttamente nel capitolo 3 per brevità.

Le cifre considerate sono:

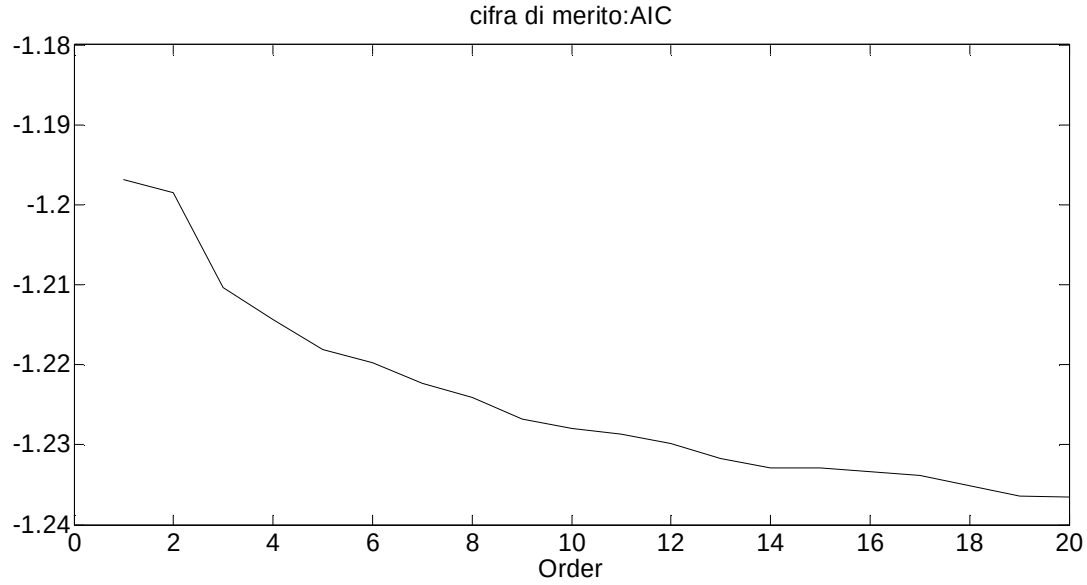
- Loss function: $J = \frac{1}{N}\sum_{i=1}^{N}(\delta u_i - \delta\hat{u}_i)^2$
- Loss function normalizzata sulla varianza del segnale: $errore = \frac{1}{N}\frac{\sum_{i=1}^{N}(\delta u_i - \delta\hat{u}_i)^2}{var(\delta u)}$
- Akaike information criterion: $AIC = 2\frac{n}{N} + \ln(J)$
- Final prediction error: $FPE = \frac{N+n}{N-n}J$

In cui δu_i rappresenta la componente stocastica del traffico misurata all'istante *i*, *n* l'ordine del modello di volta in volta considerato ed N il numero totale di campioni raccolti, pari a 30240 (numero di minuti in 4 settimane) nel caso dell'identificazione e a 10080 (pari al numero di minuti in una settimana) considerando la validazione.

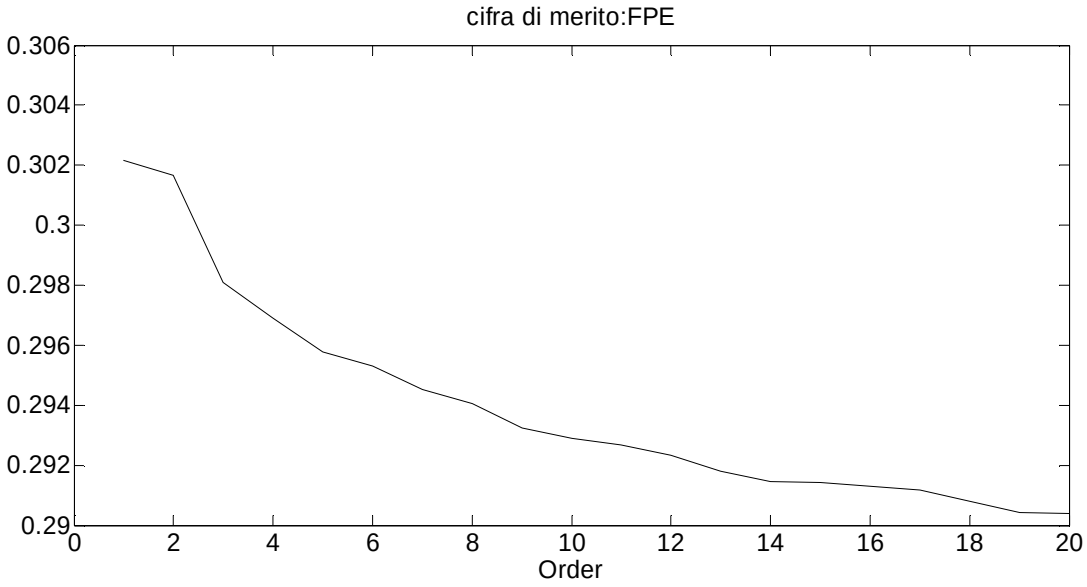
Considerando inizialmente un modello AR(*n*), il risultato relativo alla seconda cifra di merito è mostrato nel capitolo 3. I risultati per le cifre di merito rimanenti sono i seguenti:



Andamento della cifra di merito Loss function considerando un modello AR(n) identificato sulla componente stocastica del segnale



Andamento della cifra di merito AIC considerando un modello AR(n) identificato sulla componente stocastica del segnale



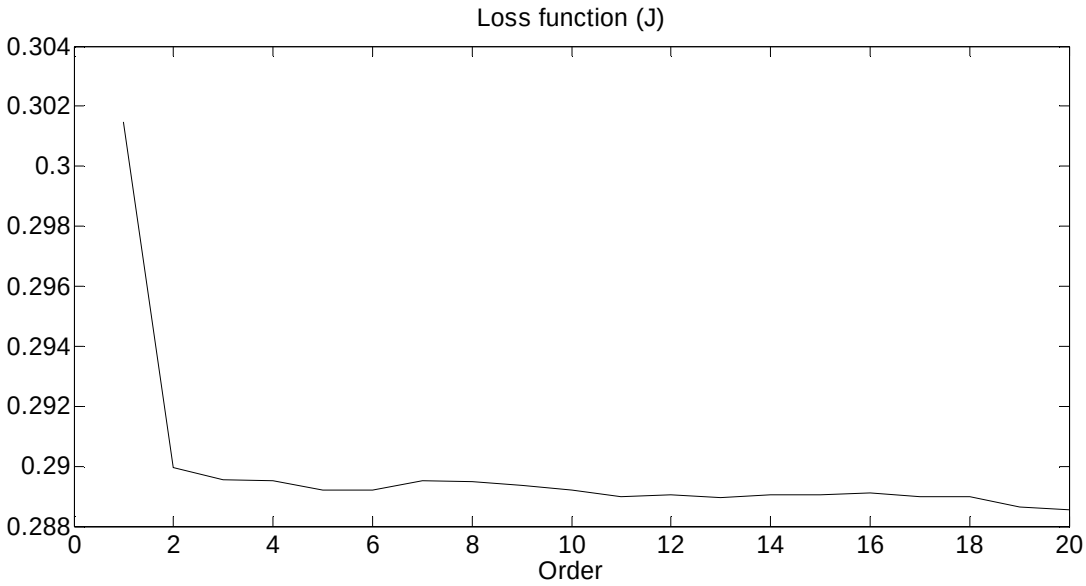
Andamento della cifra di merito FPE considerando un modello AR(n) identificato sulla componente stocastica del segnale

Come già detto nel capitolo 3, tutte le cifre di merito hanno andamento simile, portando alla conclusione che ad un aumento dell'ordine del modello corrisponde un miglioramento delle prestazioni, ma in misura minima oltre un certo ordine.

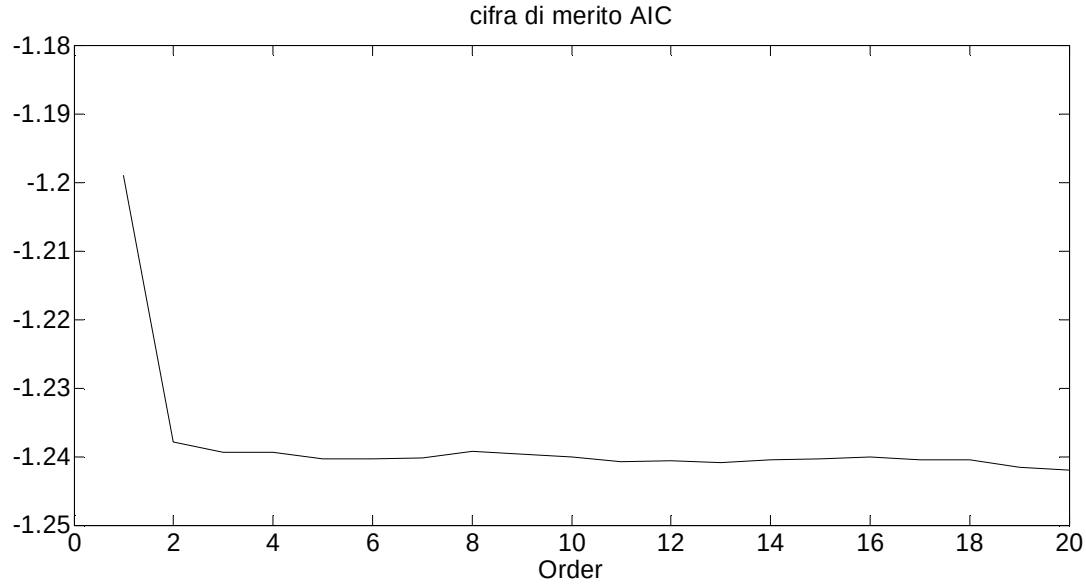
Per quanto riguarda la cifra di merito AIC ci si aspetterebbe, conoscendo l'andamento che generalmente la cifra di merito assume, un'iniziale diminuzione dei valori via via assunti (dovuta alla preponderanza del termine $\ln(J)$ sul termine $2\frac{n}{N}$) fino ad un minimo in corrispondenza del valore ottimale dell'ordine, e un successivo aumento in corrispondenza di un ulteriore aumento dell'ordine (dovuto al fatto che il termine preponderante ora risulta essere $2\frac{n}{N}$). Osservando il grafico si nota come tale aumento non avvenga: ciò è spiegabile ricordando che l'ordine n massimo considerato è pari a 20, mentre il termine N è pari a 30240; perciò in figura il contributo

del primo termine della cifra di merito non risulta essere mai preponderante. Si è comunque verificato, usando un numero N di dati totali pari a circa 200, come la cifra di merito AIC assumesse l'andamento descritto qui sopra.

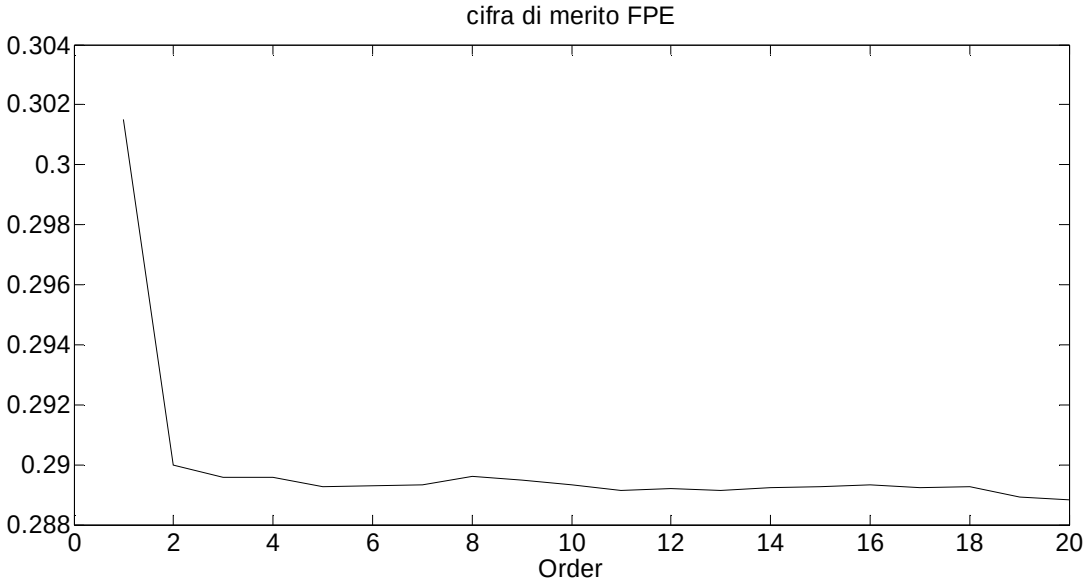
Si passa ora a illustrare i risultati ottenuti per le cifre di merito nel caso in cui si considerasse un modello ARMA:



Andamento della cifra di merito Loss function considerando un modello ARMA(p,q) identificato sulla componente stocastica del segnale



Andamento della cifra di merito AIC considerando un modello ARMA(p,q) identificato sulla componente stocastica del segnale



Andamento della cifra di merito FPE considerando un modello ARMA(p,q) identificato sulla componente stocastica del segnale

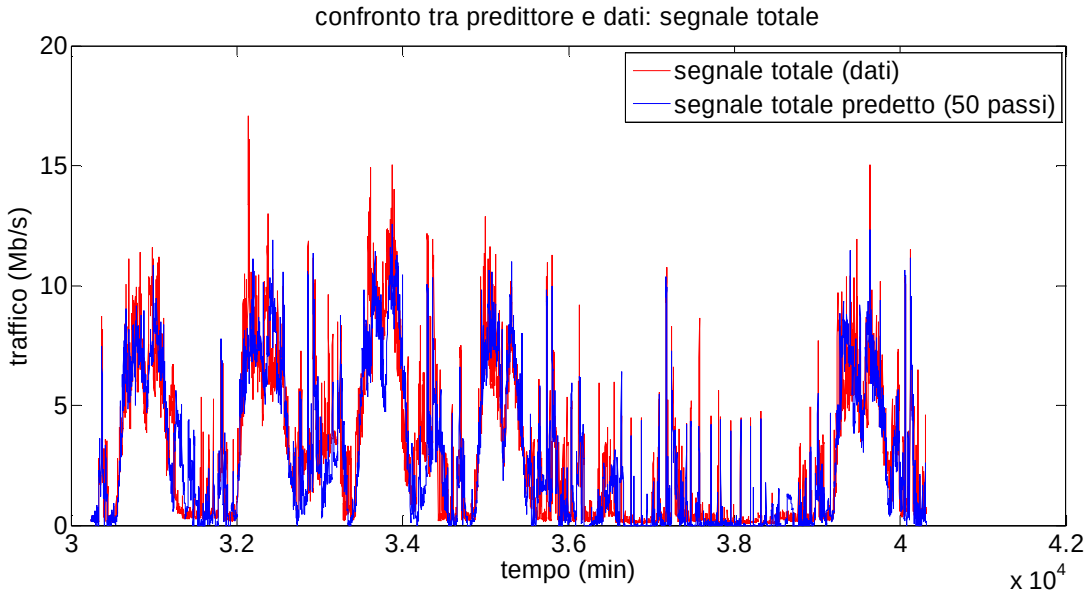
Come già detto per il modello AR, le cifre di merito mostrate portano alla conclusione che un aumento eccessivo dell'ordine considerato non migliora se non minimamente le prestazioni, andando invece ad aumentare la complessità del modello. In questo caso ciò è ancora più evidente, data la presenza del "gomito" che assume la cifra di merito per ordine superiori al secondo/terzo.

Appendice 3.2

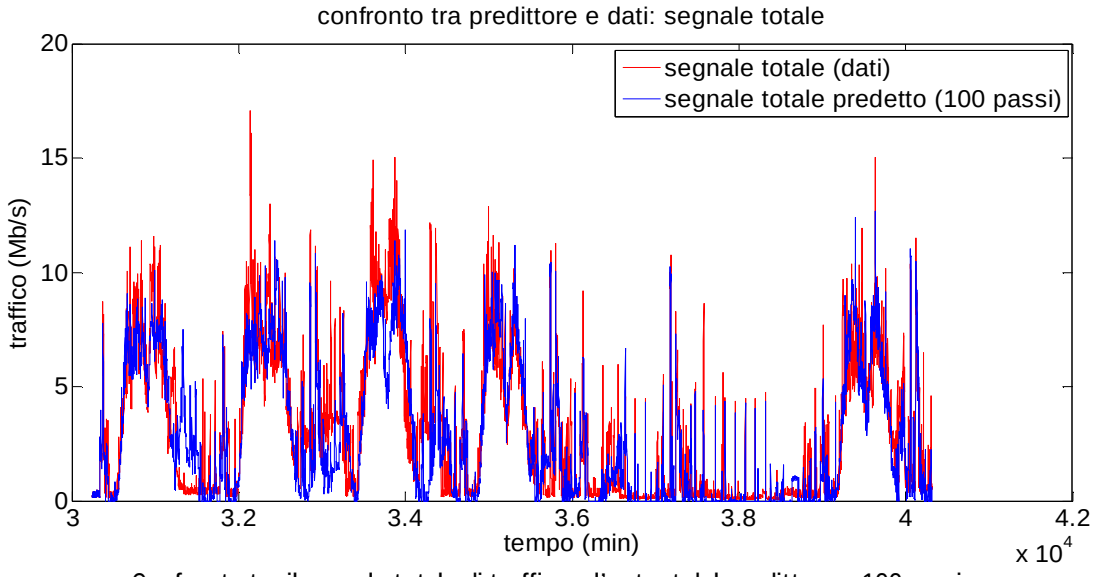
Prestazioni dei predittori ARMA per traffico e CPU

In quest'appendice vengono riportati i risultati grafici per i predittori ricavati nella trattazione del modello ARMA relativi a traffico e utilizzo di CPU. Gli orizzonti considerati sono 50 e 100 passi, non rientrando quindi in una predizione di tipo short-range; vengono mostrati per avere un riferimento su come la predizione del modello ARMA evolva all'aumentare del numero di passi considerato.

Per prima cosa si considerano i predittori per il traffico totale u.



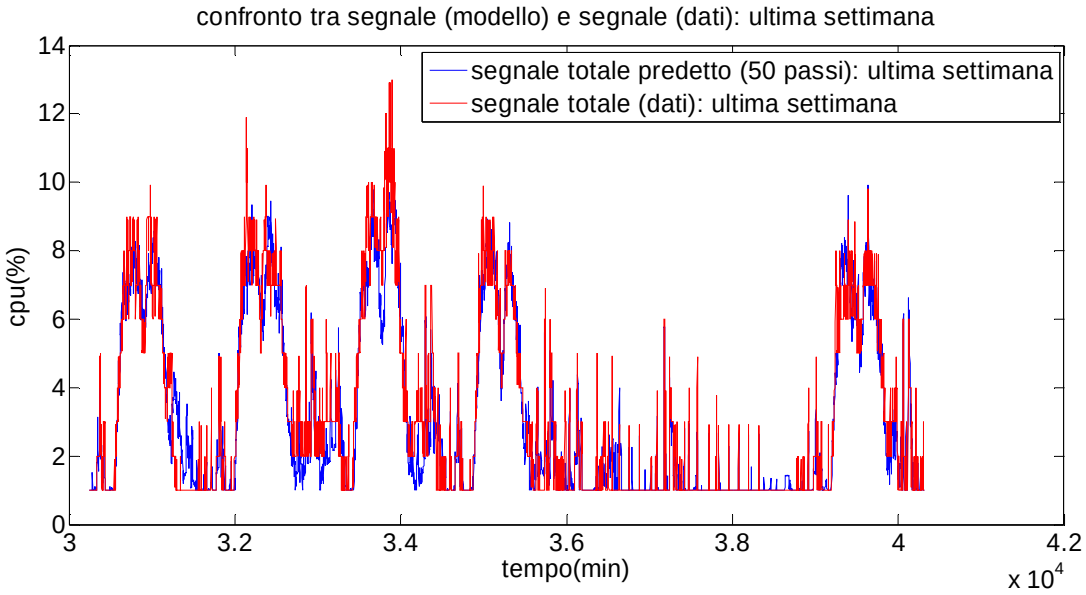
Confronto tra il segnale totale di traffico e l'output del predittore a 50 passi



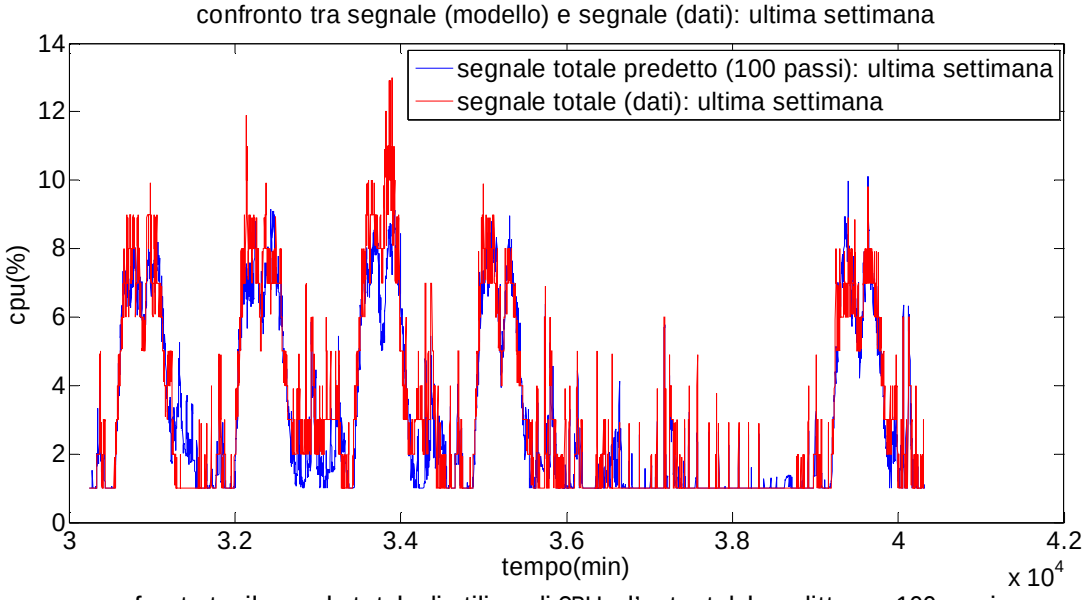
Confronto tra il segnale totale di traffico e l'output del predittore a 100 passi

Si può osservare come la bontà della predizione, confrontata con i risultati relativi ai predittori a 2, 5 e 10 passi mostrati all'interno del capitolo 3, degradi (si noti ad esempio la differenza nel predittore a 100 passi tra dati e modello nelle ore diurne dei primi 3 giorni); ciò è dovuto al fatto che il predittore ARMA sta tendendo alla media (pari a zero), portando ad una predizione che risulta essere simile alla somma delle sole componenti deterministiche del segnale (trend e stagionalità).

Si passa ora a considerare la predizione di utilizzo di CPU. Di nuovo, gli orizzonti predittivi considerati sono 50 e 100 passi:



confronto tra il segnale totale di utilizzo di CPU e l'output del predittore a 50 passi



confronto tra il segnale totale di utilizzo di CPU e l'output del predittore a 100 passi

In questo caso la differenza tra dati e predizione risulta meno marcata ma, osservando le ore centrali della giornata (per esempio quelle del terzo giorno) si osserva come si abbia una disparità sostanziale dei due segnali, cosa che non succede per orizzonti temporali limitati.

Appendice 3.3

Cifre di merito relative alla trattazione del modello ARIMA per la predizione short-range

Si riportano i grafici delle cifre di merito calcolate utilizzando un modello ARIMA, come indicato all'interno del capitolo 3.

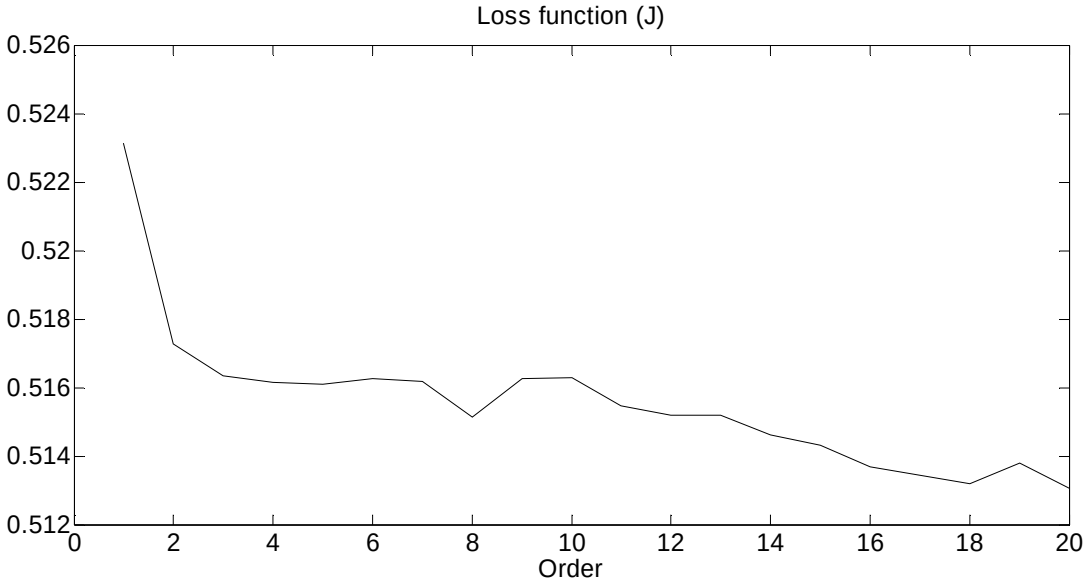
Ricordando le cifre di merito utilizzate, esse sono:

Le cifre considerate sono:

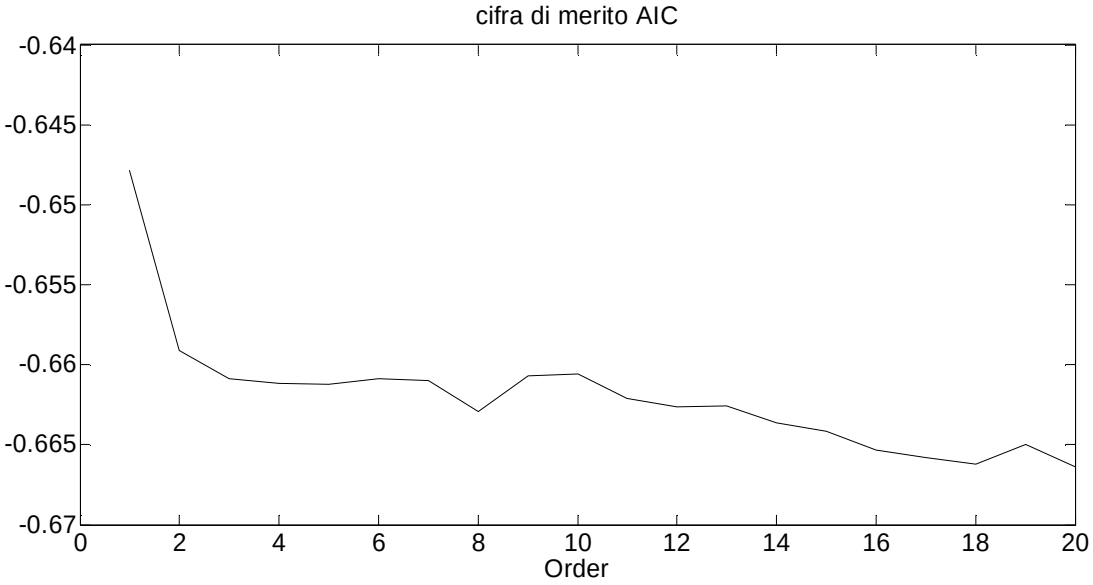
- Loss function: $J = \frac{1}{N}\sum_{i=1}^{N}(u_i - \hat{u}_i)^2$
- Loss function normalizzata sulla varianza del segnale: $errore = \frac{1}{N}\frac{\sum_{i=1}^{N}(u_i-\hat{u}_i)^2}{var(u)}$
- Akaike information criterion: $AIC = 2\frac{n}{N} + \ln(J)$
- Final prediction error: $FPE = \frac{N+n}{N-n} J$

In cui u_i rappresenta la componente stocastica del traffico misurata all'istante *i, n* l'ordine del modello di volta in volta considerato ed N il numero totale di campioni raccolti, pari a 30240 (numero di minuti in 4 settimane) nel caso dell'identificazione e a 10080 (pari al numero di minuti in una settimana) considerando la validazione.

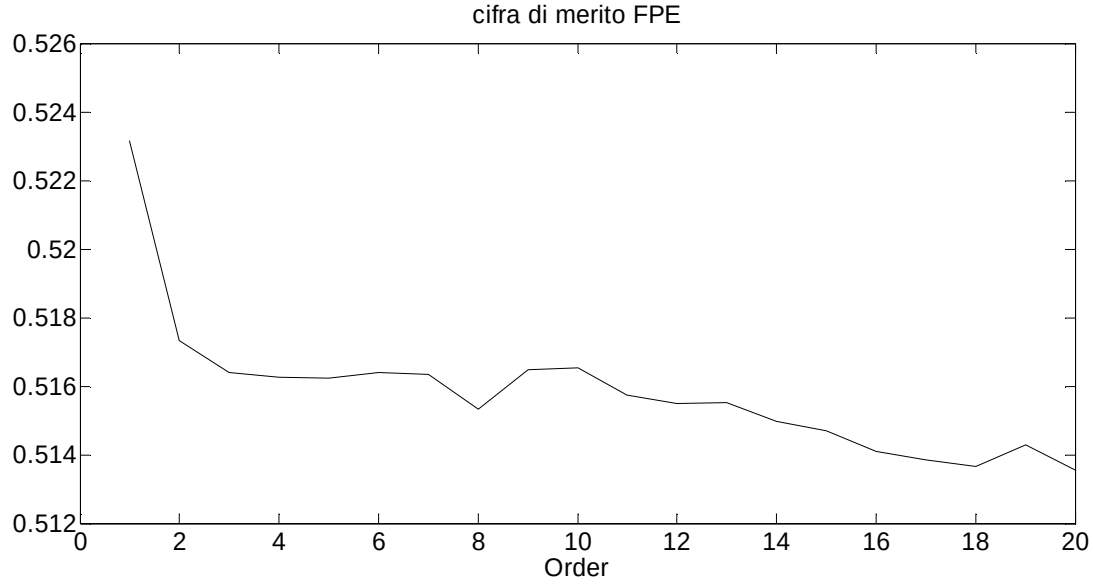
Vengono mostrati qui i risultati grafici per le cifre J, AIC e FPE, essendo già stata mostrata all'interno del capitolo 3 la seconda cifra di merito.



Andamento della cifra di merito loss function considerando un modello ARIMA(p,d,q) identificato sulla componente stocastica del segnale



Andamento della cifra di merito AIC considerando un modello ARIMA(p,d,q) identificato sulla componente stocastica del segnale



Andamento della cifra di merito FPE considerando un modello ARIMA(p,d,q) identificato sulla componente stocastica del segnale

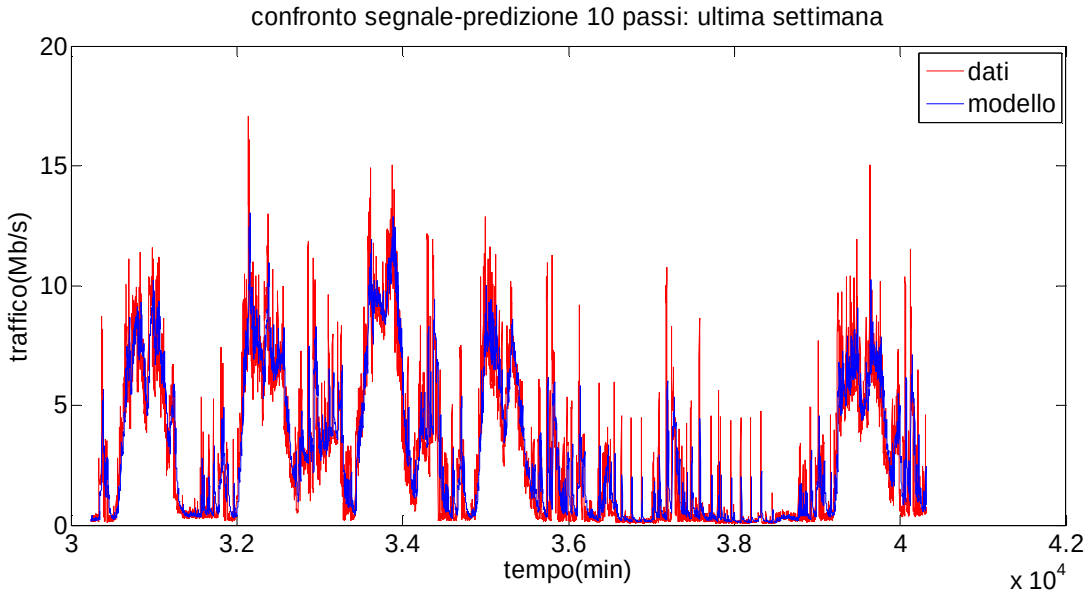
Ricordando quanto detto all'interno dell'appendice 3.1, l'andamento atteso per la cifra di merito AIC dovrebbe essere un iniziale diminuzione della cifra di merito (essendo preponderante il termine $\ln(J)$), seguito da un minimo e un successivo innalzamento (essendo ora preponderante il termine $2\frac{n}{N}$).

In realtà, essendo l'ordine n considerato piccolo rispetto al numero di dati impiegati per l'identificazione, nel grafico giustamente non è presente la seconda fase di aumento del valore della cifra di merito.

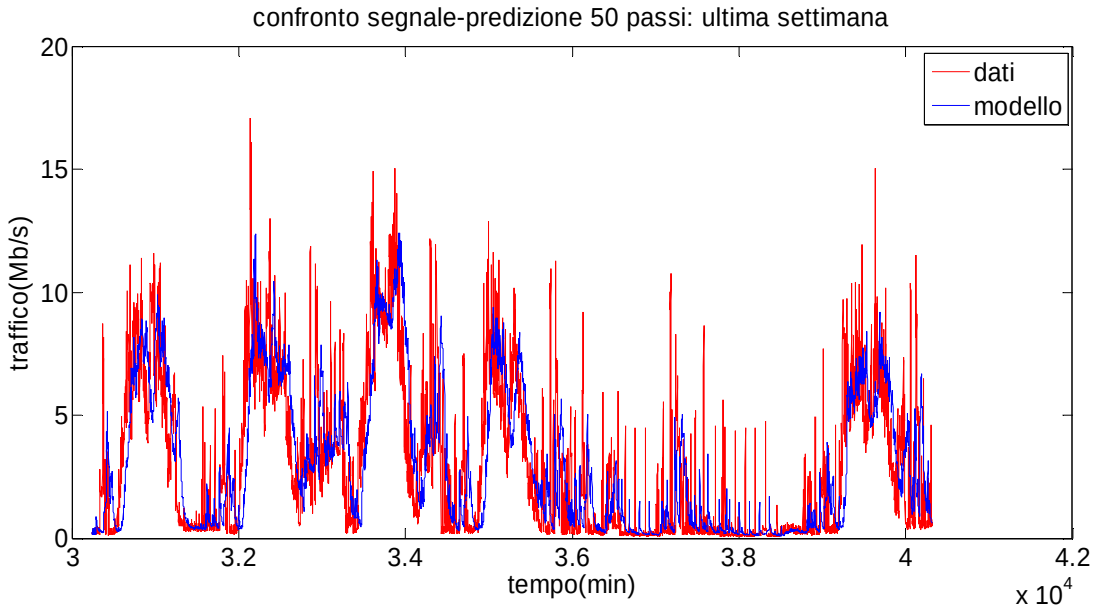
Appendice 3.4

Prestazioni dei predittori ARIMA per traffico e CPU

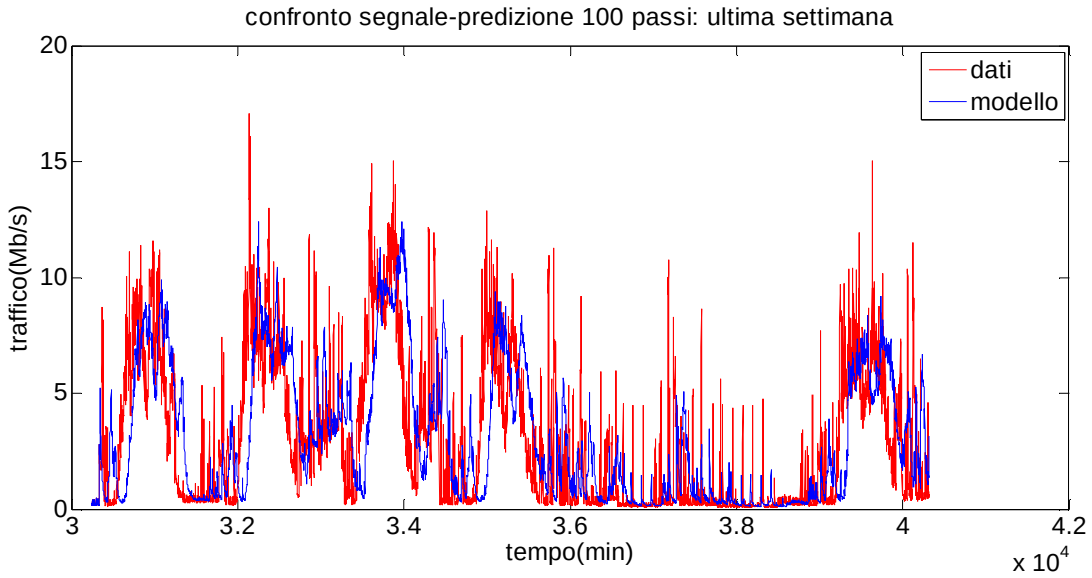
Si riportano di seguito i confronti tra predittori ARIMA e traffico e utilizzo di CPU.
Essendo stati già mostrati nel capitolo 3 i confronti dei predittori per orizzonti temporali pari a 2 e 5 passi, ci si limita a mostrare i confronti per predittori a 10, 50 e 100 passi. Come già osservato nell'appendice relativa al modello ARMA, 50 e 100 non sono orizzonti considerati short-range, vengono inseriti per poter osservare l'evoluzione della predizione del modello paragonata con l'andamento dei dati.
Per quanto riguarda i predittori di traffico:



Confronto tra il segnale totale di traffico e l'output del predittore a 10 passi



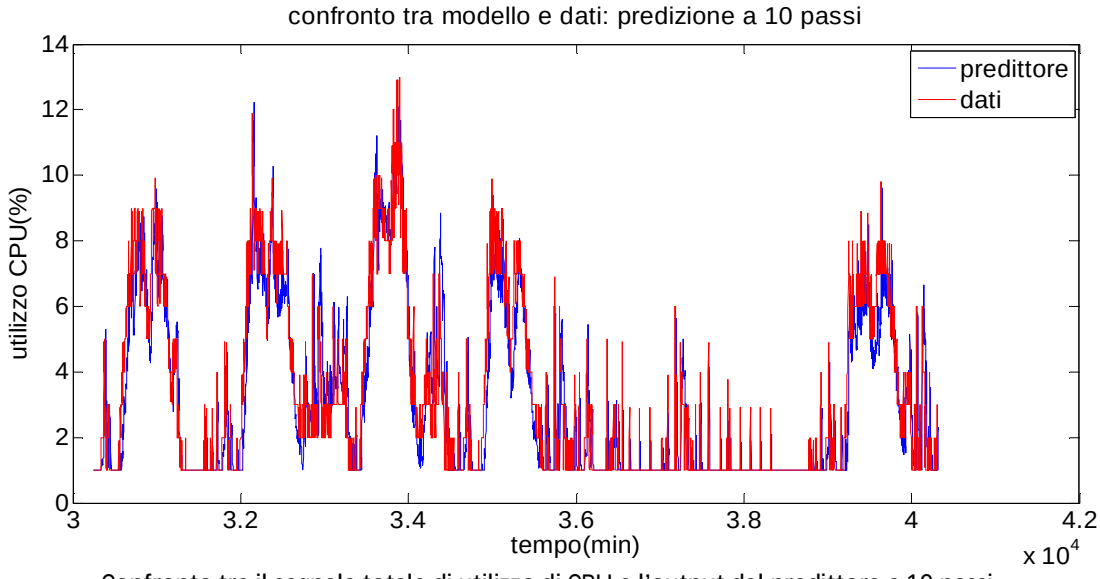
Confronto tra il segnale totale di traffico e l'output del predittore a 50 passi



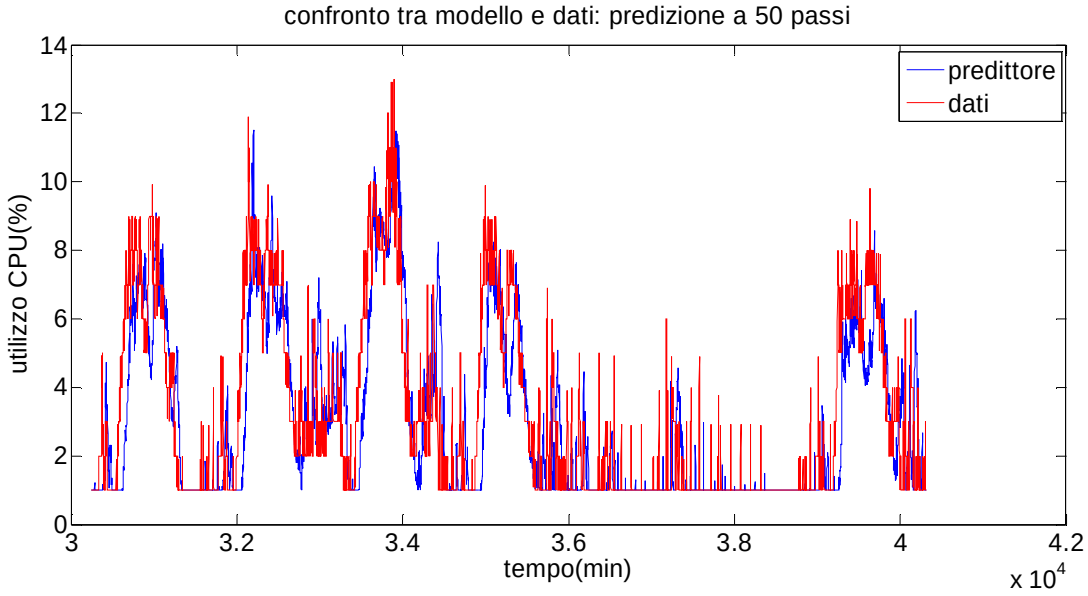
Confronto tra il segnale totale di traffico e l'output del predittore a 100 passi

Da notare la differenza rispetto ai corrispondenti predittori di traffico dell'appendice 3.2: mentre il predittore ARMA tende a dare contributo nullo alla predizione totale, facendo sì che quest'ultima vada a coincidere con la parte deterministica del segnale, nel predittore ARIMA ciò non accade; infatti, per un orizzonte predittivo abbastanza elevato (lo si può osservare nel predittore a 50 passi) il predittore tende a essere ritardato rispetto al segnale reale, dal momento che la predizione *h* passi in avanti risulta circa simile al valore che i dati hanno assunto nell'ultimo istante di tempo di cui si hanno delle informazioni.

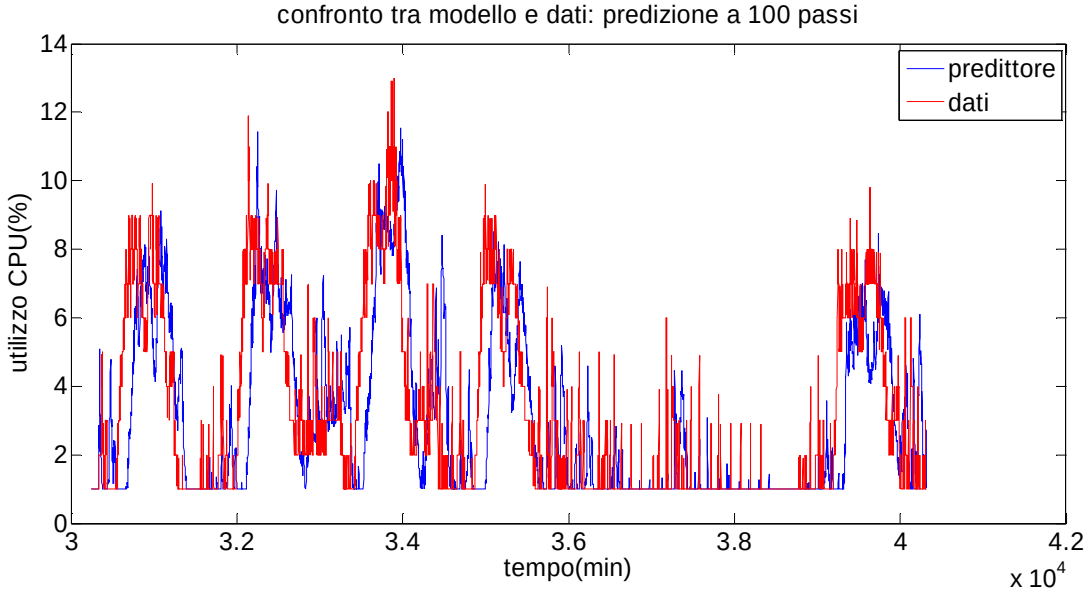
Per quanti riguarda i predittori di utilizzo di CPU si mostrano nuovamente solamente i predittori non visti nel capitolo 3:



Confronto tra il segnale totale di utilizzo di CPU e l'output del predittore a 10 passi



Confronto tra il segnale totale di utilizzo di CPU e l'output del predittore a 50 passi



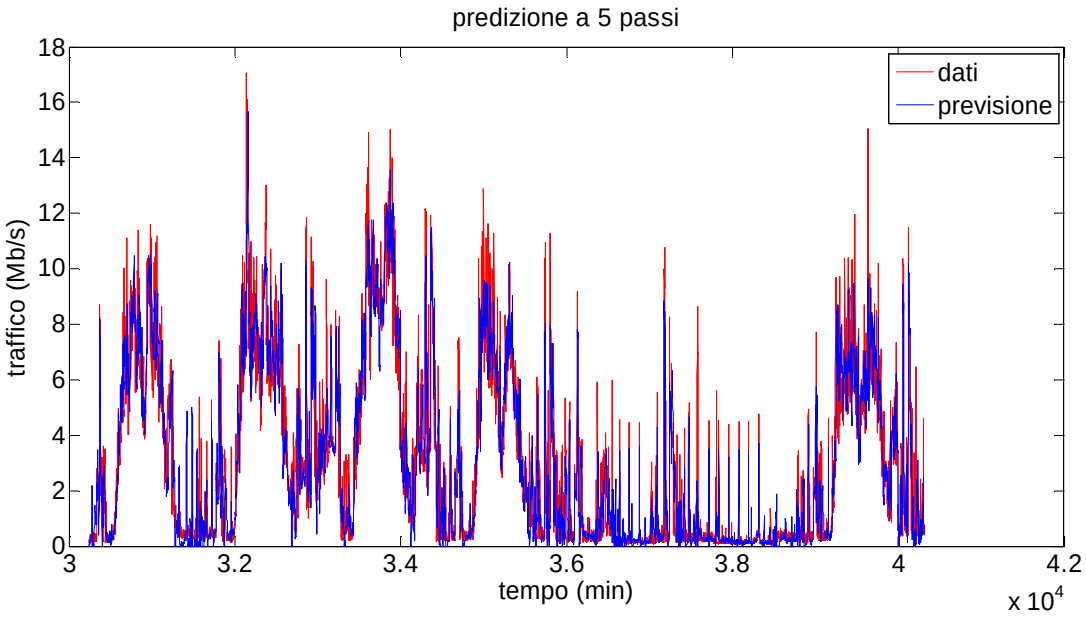
Confronto tra il segnale totale di utilizzo di CPU e l'output del predittore a 100 passi

Essendo tali predittori ottenuti a partire da quelli di traffico attraverso una relazione statica, il fatto che quest'ultimi presentino un comportamento simile ad un ritardo si riscontra, come ci si potrebbe aspettare, anche per i nuovi predittori ricavati qui sopra; a conferma del fatto che il modello ARIMA è in grado di dare buoni risultati solo per una predizione short-range (quindi di pochi passi, si noti come il predittore a 10 passi dà risultati nettamente migliori rispetto a quello a 50 passi) e che le sue prestazioni degradano velocemente all'aumentare dell'orizzonte predittivo.

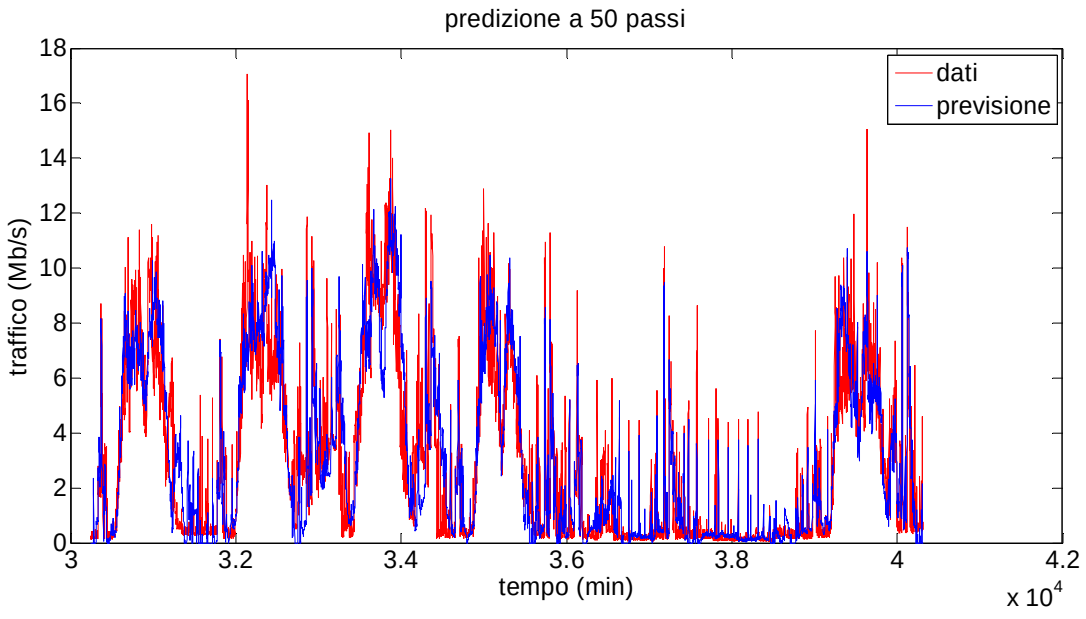
Appendice 3.5

Prestazioni del metodo di Holt-Winters per previsione di traffico e CPU

Si riportano di seguito i confronti tra le previsioni del modello Holt-Winters e i dati di traffico. Partendo dalla previsione di traffico si ottiene quella di CPU usando il modello statico descritto nel capitolo 2; i risultati ottenuti vengono di seguito confrontati con i dati.
Essendo stati già mostrati nel capitolo 3 i confronti dei predittori per orizzonti temporali pari a 2 e 10 passi, ci si limita a mostrare i confronti per predittori a 5 e 50 passi.



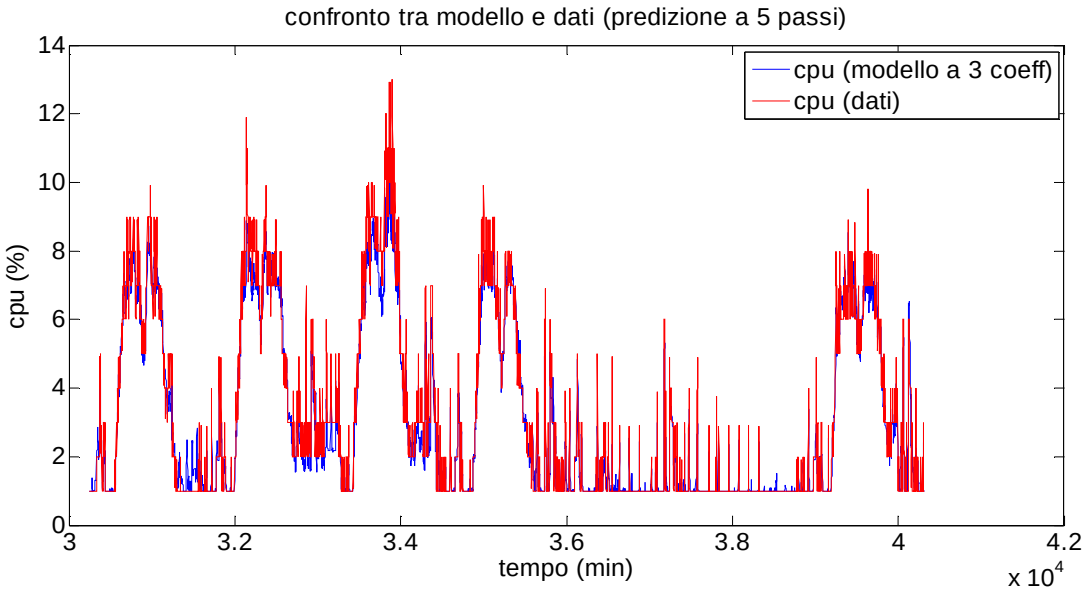
Confronto tra il segnale totale di traffico e l'output del predittore a 5 passi



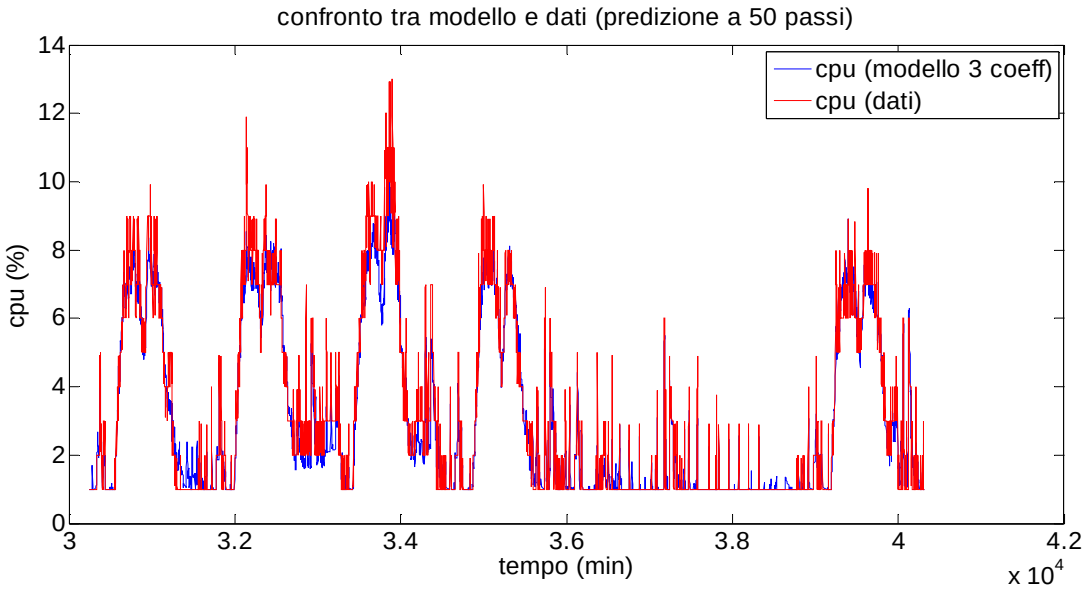
Confronto tra il segnale totale di traffico e l'output del predittore a 50 passi

Si nota come all'aumentare dell'orizzonte predittivo il modello restituisca dei valori sempre più filtrati. Ciò si può ricondurre al maggiore lisciamento al crescere dell'orizzonte predittivo. Infatti in fase di taratura, poiché la cifra di merito è calcolata rispetto all'errore del predittore ad *h* passi, i parametri ottenuti saranno tanto più bassi tanto più *h* sarà grande e, dato che a parametri bassi corrisponde un maggiore filtraggio, il segnale uscente dal modello sarà meno rumoroso rispetto ai dati

Anche per quanti riguarda i predittori di utilizzo di CPU si mostrano solamente i predittori non visti all'interno del capitolo 3:



Confronto tra il segnale totale di utilizzo di CPU e l'output del predittore a 5 passi



Confronto tra il segnale totale di utilizzo di CPU e l'output del predittore a 50 passi

I valori di CPU predetti vengono ottenuti attraverso la relazione statica descritta nel capitolo 3; la predizione mostra buone prestazioni, confrontabili con quelle della predizione di CPU per il modello ARMA, che viene ottenuta calcolando tre relazioni statiche tra le tre componenti (stocastica, trend, stagionalità).

Si ricorda che la predizione di CPU in questo caso è ottenuta attraverso una singola relazione statica non tra i segnali totali ma tra le tre componenti che vanno a formare, nel metodo di Holt-Winters, i due segnali di traffico e CPU; si è osservato come questo porti ad un vantaggio in termini di fitting rispetto all'uso di un solo coefficiente: parlando di predizione di CPU ARMA e Holt-Winters hanno valori di fitting simili; questo rappresenta un punto a favore del metodo di Holt-Winters.

Appendice 3.6

Generalizzazione dei risultati ai router della rete

ROUTER FRISELL: 2 INTERFACCE

MODELLO ARMA

IDENTIFICAZIONE					
Ordine considerato	ARMA(3,3)	ARMA(4,4)	ARMA(5,5)	ARMA(10,10)	ARMA(20,20)
FITTING componente stocastica	85.14%	85.15%	85.16%	85.21%	85.21%
FITTING segnale totale	98.179%	98.181%	98.182%	98.189%	98.187%

Ordine scelto: ARMA(5,5)

PREDIZIONE TRAFFICO			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	94.71%	91.69%	89.45%

PREDIZIONE UTILIZZO DI CPU			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	93.79%	92.03%	90.74%

MODELLO ARIMA

Il parametro d è stato posto pari a 1.

IDENTIFICAZIONE					
Ordine considerato	ARIMA(3,1,3)	ARIMA(4,1,4)	ARIMA(8,1,8)	ARIMA(13,1,13)	ARIMA(19,1,19)
FITTING segnale totale	97.544%	97.549%	97.558%	97.563%	97.561%

Ordine scelto: ARIMA(4,1,4)

PREDIZIONE UTILIZZO DI CPU			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	95.84%	93.12%	90.55%

PREDIZIONE UTILIZZO DI CPU			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	93.35%	91.50%	89.63%

MODELLO HOLT-WINTERS

IDENTIFICAZIONE DEI PARAMETRI				
Orizzonte predittivo	h=2	h=5	h=10	h=100
λ_1	0.627	0.382	0.149	0.003
λ_2	0	0	0	0
λ_3	0.100	0.121	0.218	0.275

Si può notare come all'aumentare dell'orizzonte di predizione il coefficiente λ_3 tenda ad aumentare, a differenza di quanto succedeva con il router considerato nel capitolo 3: ciò può essere spiegato dal fatto che la stagionalità iniziale assegnata a questo ed agli altri router considerati in questa appendice (ricavata filtrando la prima settimana come spiegato nel capitolo 3) porti ad avere un segnale $S(t)$ che, sommato ad una serie $L(t)$ via via più lisciata, dà dei risultati peggiori rispetto a quelli ottenuti aumentando il relativo coefficiente. In questo modo si dà più peso ai nuovi dati impiegati nel ricalcolo di $S(t)$, portando evidentemente ad una migliore prestazione.

PREDIZIONE DI TRAFFICO				
Orizzonte predittivo	h=2	h=5	h=10	h=100
FITTING segnale totale	94.99%	90.76%	87.60%	79.16%

PREDIZIONE UTILIZZO DI CPU				
Orizzonte predittivo	h=2	h=5	h=10	h=100
FITTING segnale totale	92.30%	88.85%	86.82%	82.45%

ROUTER HANCOCK: 3 INTERFACCE

MODELLO ARMA

IDENTIFICAZIONE				
Ordine considerato	ARMA(3,3)	ARMA(4,4)	ARMA(6,6)	ARMA(20,20)
FITTING componente stocastica	75.900%	80.758%	80.716%	80.762%
FITTING segnale totale	92.817%	94.275%	94.385%	94.266%

Ordine scelto: ARMA(4,4).

PREDIZIONE TRAFFICO			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	86.85%	81.43%	76.3575%

PREDIZIONE UTILIZZO DI CPU			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	89.59%	85.77%	83.06%

MODELLO ARIMA

Il parametro d è stato posto pari a 1.

IDENTIFICAZIONE					
Ordine considerato	ARIMA(3,1,3)	ARIMA(4,1,4)	ARIMA(8,1,8)	ARIMA(12,1,12)	ARIMA(19,1,19)
FITTING segnale totale	92.376%	92.373%	92.385%	92.380%	92.396%

Ordine scelto: ARIMA(3,1,3)

PREDIZIONE TRAFFICO			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	89.51%	84.99%	80.68%

PREDIZIONE UTILIZZO DI CPU			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	89.98%	85.68%	82.12%

MODELLO HOLT-WINTERS

IDENTIFICAZIONE DEI PARAMETRI				
Orizzonte predittivo	h=2	h=5	h=10	h=100
λ_1	0.497	0.156	0.056	0.002
λ_2	0	0	0	0
λ_3	0.072	0.154	0.263	0.360

PREDIZIONE DI TRAFFICO				
Orizzonte predittivo	h=2	h=5	h=10	h=100
FITTING segnale totale	86.11%	76.67%	72.16%	60.48%

PREDIZIONE UTILIZZO DI CPU				
Orizzonte predittivo	h=2	h=5	h=10	h=100
FITTING segnale totale	87.27%	79.45%	72.12%	62.60%

ROUTER MONK: 6 INTERFACCE

MODELLO ARMA

IDENTIFICAZIONE				
Ordine considerato	ARMA(3,3)	ARMA(4,4)	ARMA(6,6)	ARMA(20,20)
FITTING componente stocastica	96.12%	96.13%	96.16%	96.27%
FITTING segnale totale	99.553%	99.556%	99.558%	99.571%

Ordine scelto: ARMA(4,4)

PREDIZIONE TRAFFICO			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	99.17%	98.31%	96.94%

PREDIZIONE UTILIZZO DI CPU			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	97.83%	97.07%	96.24%

MODELLO ARIMA

Il parametro d è stato posto pari a 1.

IDENTIFICAZIONE				
Ordine considerato	ARIMA(3,1,3)	ARIMA(4,1,4)	ARIMA(8,1,8)	ARIMA(13,1,13)
FITTING segnale totale	99.455%	99.458%	99.470%	99.479%

Ordine scelto: ARIMA(4,1,4)

PREDIZIONE UTILIZZO DI CPU			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	98.94%	97.55%	94.94%

PREDIZIONE UTILIZZO DI CPU			
Orizzonte predittivo	h=2	h=5	h=10
FITTING segnale totale	88.31%	87.01%	84.51%

MODELLO HOLT-WINTERS

IDENTIFICAZIONE DEI PARAMETRI				
Orizzonte predittivo	h=2	h=5	h=10	h=100
λ_1	0.852	0.825	0.660	0.0016
λ_2	0	0	0	0
λ_3	0.043	0.117	0.275	0.465

PREDIZIONE DI TRAFFICO				
Orizzonte predittivo	h=2	h=5	h=10	h=100
FITTING segnale totale	98.65%	96.39%	92.33%	75.49%

PREDIZIONE UTILIZZO DI CPU				
Orizzonte predittivo	h=2	h=5	h=10	h=100
FITTING segnale totale	88.01%	85.68%	81.48%	69.01%

Appendice 4.1

Processing dei dati utilizzati nel capitolo 4

Di seguito è riportato il procedimento utilizzato per ricondursi al segnale considerato all'interno del capitolo 4.

La parte del segnale in nostro possesso campionata con frequenza pari a 144 minuti mostra il seguente andamento:

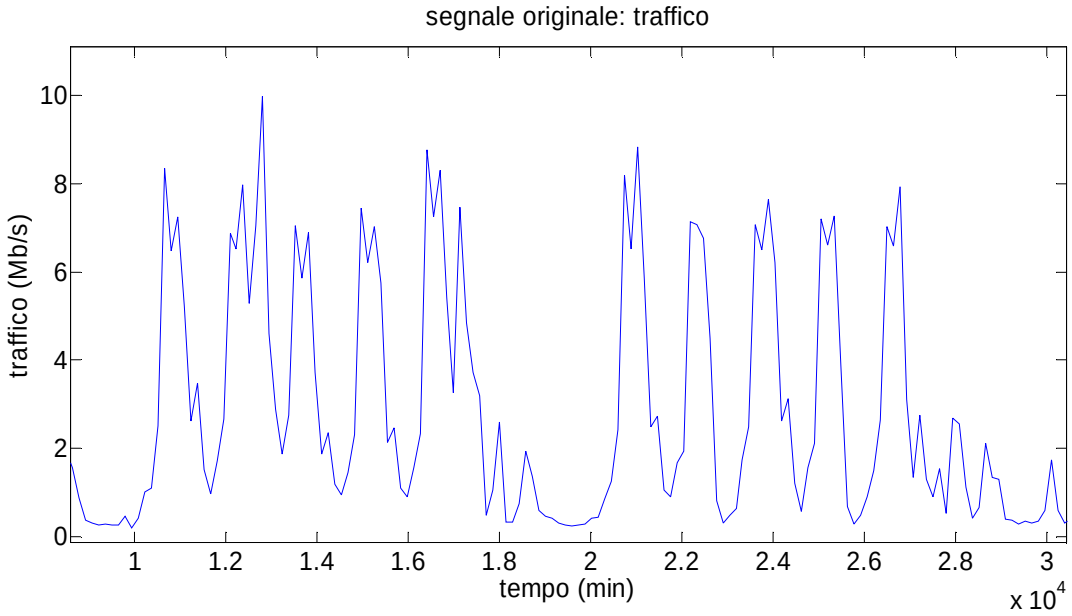


Figura 4.1: segnale originale campionato con frequenza pari a 144 minuti

Per poter ottenere i dati nella forma desiderata, si modifica il segnale come segue:

- In un primo momento si ripopola il segnale, aumentando il numero di punti che lo compongono, in modo tale da avere non più un sample ogni 144 punti, ma uno ogni 24. Per far ciò, si calcola, per ogni coppia di punti consecutivi, la pendenza della retta che li congiunge e, ad intervalli regolari, si inseriscono dei nuovi punti. Dal momento che scegliamo di voler avere un punto ogni 24 minuti (comodo perché sottomultiplo di 144 e perché avere un sample ogni circa 30 minuti è una buona granularità per una predizione di tipo long-range) inseriamo tra ogni coppia di punti consecutivi 5 altri punti. In questo modo l'andamento del segnale non varia: l'obiettivo di questo passaggio non è ottenere un segnale più simile ad uno con campionamento maggiore, ma avere un numero maggiore di punti su cui poi andare a sovrapporre il rumore che caratterizza la natura oscillatoria del traffico (e in maniera minore della CPU).
- Successivamente si procede con l'aggiunta di un rumore fittizio: si sceglie di usare, sia per il traffico che per la CPU, un rumore con distribuzione uniforme ricavato come descritto nell'appendice 4.1.
- La media è scelta pari a zero in quanto non si vuole che i valori medi dei segnali subiscano variazioni, ma solo che varino i singoli punti. Per quanto riguarda la varianza del segnale essa deve essere assegnata con cura: un valore troppo basso porterebbe ad avere variazioni del

segnale praticamente nulle, non apportando miglioramenti; d'altra parte scegliere un valore troppo alto potrebbe nascondere la dinamica del segnale all'interno delle variazioni del white noise.

Inoltre, riflettendo sulle caratteristiche della CPU e soprattutto del traffico osservati nei capitoli precedenti, ci si rende come tale varianza non possa essere costante indipendentemente dal momento della giornata in cui ci si trova, infatti:

- Le oscillazioni del segnale sono più marcate nell'orario di lavoro, quindi approssimativamente dalle 8 di mattina alle 19.30 di sera, mentre si affievoliscono di notte, in cui i segnali mostrano andamenti più costanti. Usare quindi un white noise con un'unica varianza porterebbe ad un segnale fittizio irreale, che mostrerebbe moti troppo poco oscillatori di giorno o troppo di notte.
- Tale discrepanza non si nota solo nel confronto tra giorno e notte, ma anche in quello tra week end e settimana lavorativa: nelle ore diurne corrispondenti a sabato e domenica le oscillazioni si mantengono circa su livelli simili a quelli notturni.
- Si deve considerare il rapporto tra traffico e CPU nell'assegnare i valori delle varianza dei rispettivi segnali.

Dal momento che si vuol ottenere a partire dal traffico una predizione di utilizzo di CPU, si dovrà ricavare il coefficiente che lega tali grandezze, come effettuato nei capitoli precedenti. Se però si assegnassero i valori delle varianze dei rumori indipendentemente l'uno dall'altro, tale coefficiente dovrebbe necessariamente variare. Dal momento che si sta cercando di ottenere un segnale il più vicino possibile alla realtà, ciò deve essere evitato.

Perciò riassumendo quando detto sopra, si ha bisogno di:

- 4 white noise (2 per ogni segnale)
- media nulla per tutti i segnali
- deve valere la relazione:

$$\frac{\vartheta^2_{u,notte}}{\vartheta^2_{y,notte}} = \frac{\vartheta^2_{u,giorno}}{\vartheta^2_{y,giorno}} = \frac{\frac{1}{N}\sum_{i=1}^{N} u_i}{\frac{1}{N}\sum_{i=1}^{N} y_i}$$

In cui, $\vartheta^2_{i,giorno}$ rappresenta la varianza del rumore bianco da usare nelle ore diurne della settimana lavorativa per il traffico (u) e per la CPU (y), $\vartheta^2_{i,notte}$ la varianza del white noise usato sempre nella settimana lavorativa di notte, $\frac{1}{N}\sum_{i=1}^{N} u_i$ rappresenta il valore medio del traffico e $\frac{1}{N}\sum_{i=1}^{N} y_i$ il valor medio della CPU. Per quanto riguarda invece i white noise da usare nel week-end, essendo simili gli andamenti diurni e notturni nei giorni festivi, si sceglie di assegnare una varianza uguale ad entrambi e pari alla varianza notturna assegnata ai giorni feriali $\vartheta^2_{i,notte}$.

Per quanto riguarda invece la parte di dati campionata con frequenza differente, il procedimento risulta molto più immediato: i 12 giorni circa di dati già campionati a 24 minuti non subiscono variazioni, mentre i 2 giorni campionati con frequenza pari a 6 minuti vengono sottocampionati: da notare che non ci si limita a prelevare un campione ogni 4, scartando i restanti 3; ma che ogni nuovo punto è ottenuto come la media di tutti i 4 sample che va a sostituire.

In questo modo i segnali assumono un andamento sufficientemente simile alla realtà. In figura 4.2 e 4.3 vengono mostrati gli andamenti dei segnali ricavati, comparati con i segnali di partenza:

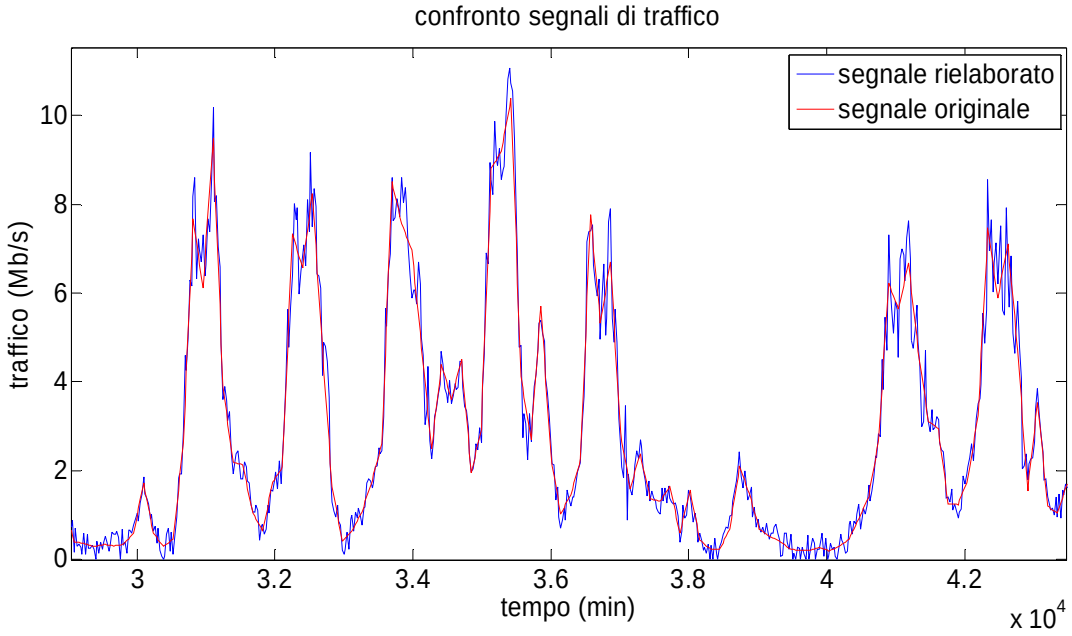


Figura 4.2: confronto tra segnale originale e segnale rielaborato (traffico)

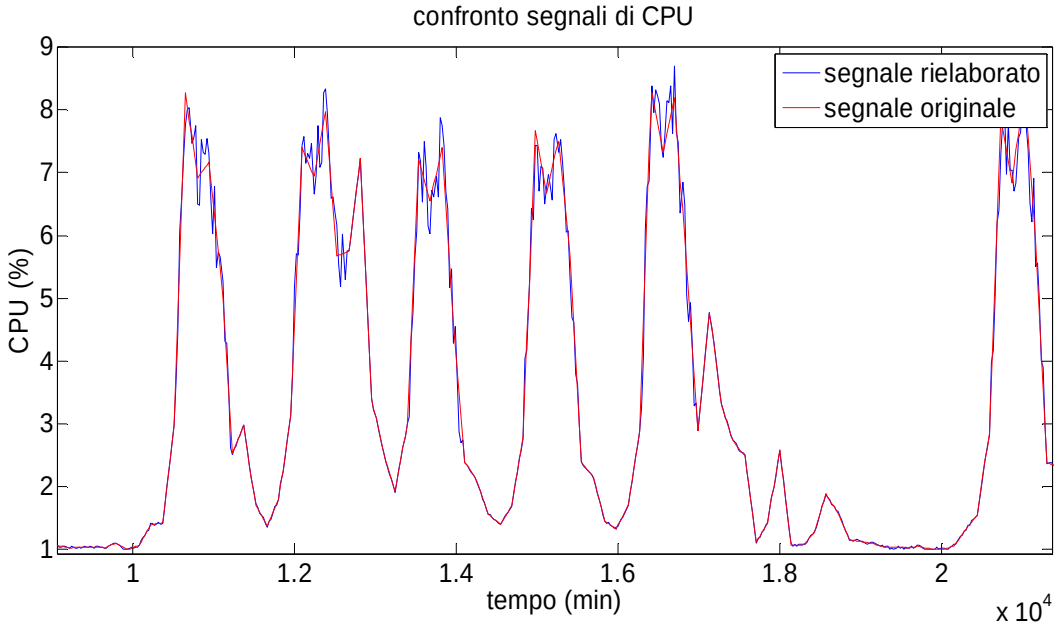


Figura 4.3: confronto tra segnale originale e segnale rielaborato (traffico)

Da notare come, per il segnale di CPU, non ci sia quasi differenza tra il segnale originale e quello rielaborato nelle ore notturne e nel week end: ciò è voluto, in quanto la CPU manteneva valori pressoché costanti in queste fasce orarie nei capitoli precedenti, per cui si è scelto di far in modo di mantenere tale comportamento, limitando le oscillazione notturne.

Appendice 4.2

Cifre di merito relative alla trattazione del modello ARMA per la predizione long-range

Di seguito si riportano i risultati relativi alle cifre di merito considerate per la determinazione dell'ordine del modello ARMA utilizzato per la predizione di tipo long-range. Le cifre di merito considerate sono:

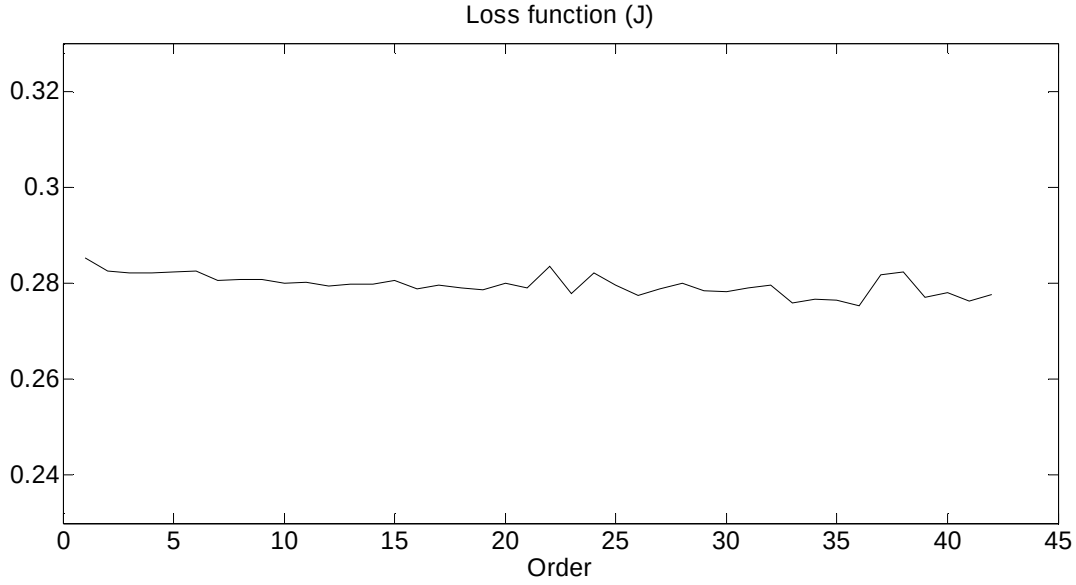
-Loss function: $J = \frac{1}{N}\sum_{i=1}^{N}(\delta u_i - \delta\hat{u}_i)^2$

-Loss function normalizzata sulla varianza del segnale: $errore = \frac{1}{N}\frac{\sum_{i=1}^{N}(\delta u_i - \delta\hat{u}_i)^2}{var(\delta u)}$

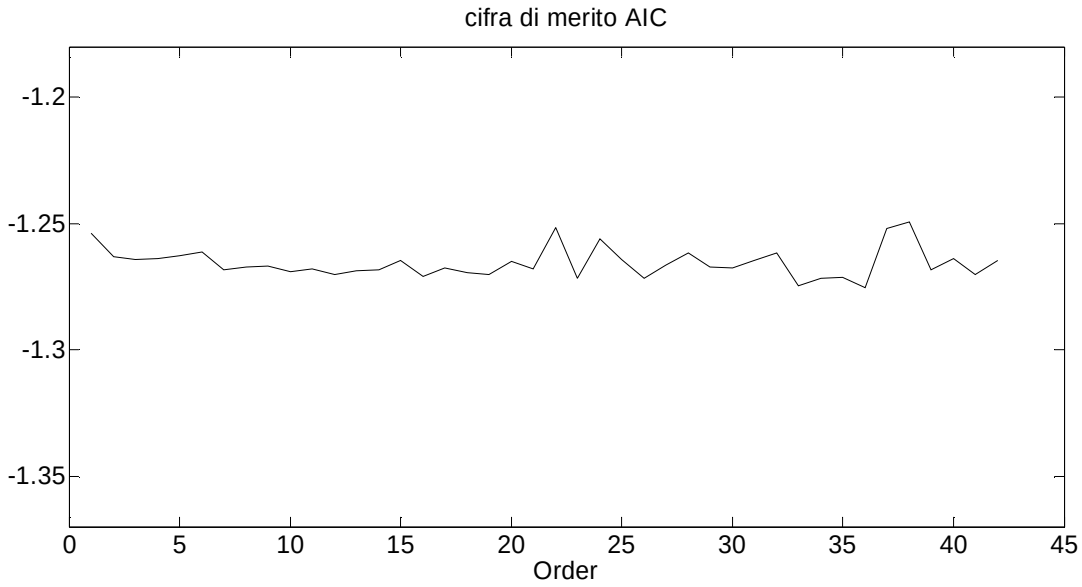
-Akaike information criterion: $AIC = 2\frac{n}{N} + \ln(J)$

-Final prediction error: $FPE = \frac{N+n}{N-n} J$

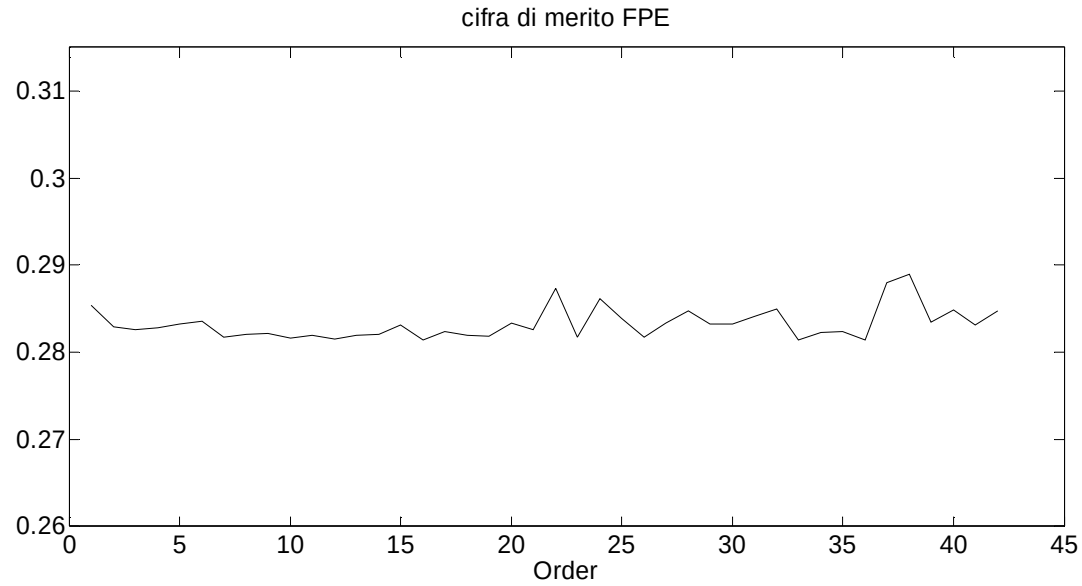
In cui δu_i rappresenta la componente stocastica del traffico misurata all'istante *i, n* l'ordine del modello di volta in volta considerato ed N il numero totale di campioni raccolti, pari a 30240 (numero di minuti in 4 settimane) nel caso dell'identificazione e a 10080 (pari al numero di minuti in una settimana) considerando la validazione. Dal momento che la seconda cifra di merito è già stata mostrata all'interno del capitolo 4, si mostrano qui solamente le restanti tre:



Andamento della cifra di merito Loss function considerando la componente stocastica del segnale



Andamento della cifra di merito AIC considerando la componente stocastica del segnale



Andamento della cifra di merito FPE considerando la componente stocastica del segnale

Come si può osservare, anche per queste cifre decimali vale quanto affermato all'interno del capitolo 4: all'aumentare dell'ordine considerato non si ha un effettivo miglioramento delle prestazioni se non minimo, ma un aumento delle cancellazioni polo-zero oltre a una maggiore complessità del modello di volta in volta considerato.

Appendice 4.3

Generalizzazione dei risultati ai router della rete

ROUTER FRISELL: 2 INTERFACCE

MODELLO ARMA

Ordine considerato	ARMA(4,4)	ARMA(7,7)	ARMA(5,5)	ARMA(15,15)
FITTING componente stocastica	91.750%	91.577%	91.733%	91.572%
FITTING segnale totale	98.939%	98.937%	98.917%	98.916%

Ordine scelto: ARMA(4,4).

PREDIZIONE DI TRAFFICO	
Orizzonte predittivo	h=1680 (4 settimane)
FITTING segnale totale	80.75%

PREDIZIONE DI UTILIZZO DI CPU	
Orizzonte predittivo	h=1680 (4 settimane)
FITTING segnale totale	84.07%

MODELLO HOLT-WINTERS

IDENTIFICAZIONE DEI PARAMETRI						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
λ_1	0.859	0.958	0.931	0.041	0.019	0.05
λ_2	0	0	0	0	0	0
λ_3	0.706	1	1	0.358	0.248	0.340

PREDIZIONE DI TRAFFICO						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
FITTING segnale totale	78.09%	78.48%	78.17%	83.30%	83.25%	83.54%

PREDIZIONE DI UTILIZZO DI CPU						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
FITTING segnale totale	82.59%	83.04%	82.70%	88.18%	88.23%	88.36%

Si può notare come all'aumentare dell'orizzonte di predizione il coefficiente λ_1 tenda ad assumere valori nell'intorno dell'unità in corrispondenza di valori di h bassi, mentre subisca un brusco calo nel passaggio da 5 a 10 passi. Ciò era stato evidenziato anche nel router Adderley all'interno del capitolo 4. La differenza in questo caso riguarda le prestazioni che ne derivano: se nel caso del router Adderley non vi erano sostanziali differenze nel fitting del modello, in questo ci si ritrova ad avere un miglioramento di circa il 5% (per la predizione di traffico) della cifra di merito. Ciò è dovuto all'ultimo valore che $L(t)$ assume al variare di λ_1: per valori di h bassi l'ultimo valore disponibile della serie è tale per cui la predizione risulta essere in modulo più alta dei dati, in particolare nella ore notturne, portando a prestazioni minori. All'aumentare dell'orizzonte h considerato, la serie $L(t)$ viene progressivamente filtrata, andando ad assumere valori minori, con conseguente aumento delle performance della predizione.

ROUTER HANCOCK: 3 INTERFACCE

MODELLO ARMA

IDENTIFICAZIONE				
Ordine considerato	ARMA(3,3)	ARMA(4,4)	ARMA(13,13)	ARMA(21,21)
FITTING componente stocastica	84.280%	84.349%	84.119%	84.144%
FITTING segnale totale	97.055%	97.068%	97.040%	97.029%

Ordine scelto: ARMA(4,4).

PREDIZIONE DI TRAFFICO	
Orizzonte predittivo	h=1680 (4 settimane)
FITTING segnale totale	73.68%

PREDIZIONE DI UTILIZZO DI CPU	
Orizzonte predittivo	h=1680 (4 settimane)
FITTING segnale totale	80.48%

MODELLO HOLT-WINTERS

IDENTIFICAZIONE DEI PARAMETRI						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
λ_1	0.661	0.706	0.655	0.047	0.034	0.024
λ_2	0	0	0	0	0	0
λ_3	0.372	0.507	0.404	0.335	0.264	0.353

PREDIZIONE DI TRAFFICO						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
FITTING segnale totale	74.24%	73.90%	74.57%	76.10%	76.08%	76.68%

PREDIZIONE DI UTILIZZO DI CPU						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
FITTING segnale totale	80.24%	79.83%	80.81%	83.84%	83.75%	83.89%

ROUTER MONK: 6 INTERFACCE

MODELLO ARMA

IDENTIFICAZIONE				
Ordine considerato	ARMA(4,4)	ARMA(5,5)	ARMA(9,9)	ARMA(29,29)
FITTING componente stocastica	62.487%	62.500%	62.129%	61.807%
FITTING segnale totale	95.181%	95.183%	95.136%	95.094%

Ordine scelto: ARMA(5,5).

PREDIZIONE DI TRAFFICO	
Orizzonte predittivo	h=1680 (4 settimane)
FITTING segnale totale	82.28%

PREDIZIONE DI UTILIZZO DI CPU	
Orizzonte predittivo	h=1680 (4 settimane)
FITTING segnale totale	76.20%

MODELLO HOLT-WINTERS

IDENTIFICAZIONE DEI PARAMETRI						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
λ_1	0.475	0.505	0.207	0.070	0.015	0.017
λ_2	0	0	0	0	0	0
λ_3	0.420	0.429	0.416	0.290	0.281	0.215

PREDIZIONE DI TRAFFICO						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
FITTING segnale totale	82.05%	82.28%	82.45%	83.01%	82.96%	83.85%

PREDIZIONE DI UTILIZZO DI CPU						
Orizzonte predittivo	h=1	h=2	h=5	h=10	h=100	h=420
FITTING segnale totale	77.09%	77.56%	80.01%	81.34%	79.52%	79.15%

Bibliografia

[1] Maria Bakardjieva, Internet Society: The internet in everyday life, Sage, 2005.

[2] Andrew Tanenbaum, Computer Networks, 4th edition, Prentice Hall, 2003.

[3] Radia Perlman, Interconnections Bridges Routers Switches and Internetworking Protocols, 2nd edition , Prentice Hall, 1999.

[4] S. Bittanti, Identificazione dei Modelli a Sistemi Adattativi, Pitagora editrice, Bologna, 2005.

[5] Sergio Bittanti, Teoria della predizione e del filtraggio, Pitagora editrice, Bologna, 2000.

[6] H.Zare Moayedi, M.A.Masnadi-Shirazi, Arima Model for Network Traffic Prediction and Anomaly Detection, Shiraz University,2008.

[7] Bo Zhou, Dan He, Zhili Sun and Wee Hock Ng, Network Traffic Modeling and Prediction with ARIMA/GARCH, Centre for Communication System Research,University of Surrey, 2002.

[8] Bollerslev,T.R.Y.Chou, and K.F.Kroner, "ARCH Modeling in Finance: A Review of the Theory and Empirical Evidence," Journal of Econometrics, Vol.52, 1992.

[9] T.Nakatsuma and H.Tsurumi, "ARMA-GARCH models: Bayes Estimation Versus MLE, and BayesNon-stationary Test", department working papers with number 199619, department of Economics, Retgurs University, 1996.

[10] Vassilios C. Moussas, Marios Daglis, and Eva Kolega, Network traffic modeling and prediction using multiplicatie seasonal arima models, Network Operations Centre (NOC), Technological Educational Institution (T.E.I.) of Athens, 2005.

[11] Yen-Wen Chen, Chung-Chi Chou, Traffic Modeling of a Sub-Network by Using ARIMA
National Central University, Jung-Li, Tao-Yuan, 2001.

[12] A. Nucci, K. Papagiannaki: Design, Measurement and Management of Large-Scale IP Networks, Cambridge University Press, 2008.

[13] Hernandez, Weiss, Guido, A First Course in Wavelets. CRC Press, Boca Raton, Florida, 1996.

[14] Daubechies, I. Ten Lectures on Wavelets. Society for Industrial and Applied Mathematics, Philadelphia,1992.

[15] S. Stoev, G. Michailidis, J. Vaughan, Global Modeling and Prediction of Computer Network Traffic, December 2009.

[16] J. Vaughan, S. Stoev, G. Michailidis, Network-wide Statistical Modeling, Prediction and Monitoring of Computer Traffic, October 2011.

[17] P. Lluka, NetSpyGlass version 4: Configuration Guide.

[18] Chandra Kopparapu, Load Balancing Servers, Firewalls and Caches, Wiley Computer Publishing, Canada,2002.

[19] H. Akaike, A new look at the statistical model identification**,** Institute of Statistical Mathematics, Tokio, Japan, 1974.

[20] Zazli Chik, Performance of Order Selection Criteria for Short-Time Series, University of Kuala Lumpur, Malaysia, 2002.

[21] Gelper, Fried, Groux, Robust Forecasting with Exponential and Holt-Winters Smoothing, Leuven, 2007.

[22] Gelper, Fried, Groux, Computational Aspects of Robust Holt-Winters Smoothing Based on M-Estimation, Leuven, 2008.