Report n. 152

Reti Neurali e Analisi delle Serie Storiche: un modello per la previsione del BTP future

Emilio Barucci, Paolo Cianchi, Leonardo Landi, Anna Lombardi

Pisa, Novembre 1999

Reti Neurali e Analisi delle Serie Storiche: un modello per la previsione del BTP future

Emilio Barucci

Dipartimento di Statistica e Matematica Applicata all'Economia,

Università di Pisa,

Paolo Cianchi e Leonardo Landi

AMAT srl,

via T. Fiesoli 89F, Campi Bisenzio (FI)

Anna Lombardi

Linköping University

ITN/Campus Norrköping, SE-601 74 Norrköping

October 6, 1999

Abstract

In questo lavoro le reti neurali (*RN*) vengono applicate alla previsione del prezzo giornaliero del future sul BTP (LIFFE) a 10 anni. L'articolo, oltre che di una breve presentazione delle *RN* , si articola in due sezioni. La prima, a carattere metodologico, ha come obiettivo la presentazione degli aspetti metodologici connessi con l'applicazione delle *RN* all'analisi delle serie storiche. La seconda si incentra sull'applicazione delle *RN* alla previsione del prezzo del future sul btp decennale. Riguardo agli aspetti metodologici delle *RN* la nostra attenzione è rivolta in particolare alle tecniche di stima (algoritmi supervisionati) ed ai criteri per la selezione del modello. L'analisi confronta il comportamento di modelli lineari e nonlineari (univariati e multivariati). La costruzione del modello neurale si basa su una tecnica di pre-whitening che consiste nell'utilizzare il miglior modello univariato come filtro per determinare la significativitá in delle variabili esogene. Il modello univariato é un ARIMA (5,1,5). Le variabili esogene prese in considerazione sono prezzo del future sul bund, capitalizzazione del mercato, cambio lira-dollaro, indice mibtel, volume degli scambi, differenziale tassi italiani-tedeschi. A causa della piccola dimensione della serie storica si é fatto ricorso al metodo del *bootstrap* per addestrare la *RN* . La tecnica di stima si basa su un algoritmo del secondo ordine che risulta essere molto più veloce del classico algoritmo back propagation. Il modello neurale che risulta avere la migliore performance in fase previsiva è costituito da uno strato nascosto con tre o cinque neuroni. Le variabili di ingresso sono

il prezzo del future sul BTP e sul BUND uno e due passi prima. Il modello neurale ha una performance superiore rispetto a quella del modello lineare come dimostrato nell'analisi dei risultati.

1 Introduzione

da fare

2 Reti Neurali: Cenni Introduttivi

All'interno della letteratura econometrica le *RN* possono essere classificate come modelli nonlineari non correttamente specificati. Quest'ultima caratteristica discende dalla loro proprietà di essere approssimatori di funzioni e quindi di non rappresentare per loro natura il "vero" modello generatore dei dati.

Le architetture neurali sono costituite da unità computazionali elementari dette *neuroni*. Le *RN* possono avere strutture diverse a seconda dell'applicazione; in questa sede, per l'analisi delle serie storiche, faremo riferimento ad architetture in cui i neuroni sono raggruppati in *strati* ordinati: strato di ingresso, strati intermedi o nascosti ed uno strato di uscita. I neuroni dello strato di ingresso hanno il compito di rappresentare le variabili esplicative, gli altri strati sono invece composti da unità computazionali caratterizzate da una *funzione di attivazione*. La funzione di attivazione del neurone j-esimo $f_j : \Re \to \Re$ è in generale di tipo nonlineare; nel caso in cui la funzione di attivazione di ogni neurone sia lineare la *RN* si riconduce ad un modello lineare. I neuroni nello strato di uscita sono solitamente caratterizzati da una funzione di attivazione lineare.

Ogni neurone riceve segnali di ingresso $\{x_1, \ldots, x_n\} = \mathbf{x} \in \Re^n$ sia da altri neuroni che dall'esterno della *RN* . L'argomento della funzione di attivazione del neurone è calcolato combinando linearmente le variabili di ingresso con i *pesi* associati al neurone e sommando a tale combinazione una *soglia* (l'insieme dei pesi e delle soglie costituiscono i parametri della *RN*). Il peso del neurone j-esimo associato alla variabile di ingresso i-esima è rappresentato da w_{ij}, la soglia da w_{0j}. La funzione computata dal neurone j può essere quindi espressa come segue

$$o_j = f_j \left(\sum_{i=1}^{n} x_i w_{ij} + w_{0j} \right) = f_j(a_j) \ ,$$

dove $a_j = \sum_{i=1}^{n} x_i w_{ij} + w_{0j}$ rappresenta l'*attivazione* del neurone j-esimo. Per semplificare la notazione non distingueremo, di qui in avanti, tra la soglia e gli altri pesi di un neurone j-esimo che saranno rappresentati sinteticamente tramite il vettore $\mathbf{w}_j \in \Re^{n+1}$; la variabile di ingresso associata alla soglia del neurone è una costante pari ad 1.

La funzione di attivazione può avere diverse espressioni analitiche: funzione di attivazione discontinua (a gradino $[0,1]$) oppure una funzione di attivazione continua quale la logistica $\sigma(a)$, la tangente iperbolica $\tanh(a)$, la gaussiana $\phi(a)$:

$$\sigma(a) = \frac{1}{1+e^{-a}},\ \tanh(a) = \left(1 - \frac{2}{1+e^{a}}\right),\ \phi(a) = e^{-a^2}.$$

Per soddisfare esigenze computazionali, le funzioni di attivazione devono avere come codominio l'intervallo $[0,1]$. Al fine di rappresentare valori non appartenenti a questo intervallo, come nel caso di neuroni dello strato di uscita, conviene inserire un operatore lineare nella *RN* .

Limitandoci all'utilizzo delle *RN* per l'analisi delle serie storiche rappresenteremo tramite $\{x_1(t),\ldots,x_n(t)\} = \mathbf{x}(t)$ le variabili esplicative o variabili di ingresso della *RN* al tempo t e tramite $\{y_1(t),\ldots,y_{n_o}(t)\} = \mathbf{y}(t)$ le variabili da rappresentare, la stima o uscita della *RN* sarà indicata tramite $\hat{\mathbf{y}}(t) = \mathcal{N}(\mathbf{x}(t))$. $\mathcal{N}(\cdot)$ rappresenta la funzione descritta dalla rete neurale.

I modelli di *RN* possono essere classificati in base alla loro struttura (architettura) ed al metodo di stima dei parametri. Seguendo il primo criterio le architetture sono distinte in *feedforward* (*statiche*) e *feedback* (*dinamiche*). Seguendo il secondo criterio è possibile individuare *RN supervisionate* e *non supervisionate.*

La distinzione tra reti statiche e dinamiche deriva dal tipo di connessioni tra i neuroni della *RN* . Se in una rete vi sono collegamenti di retroazione tra neuroni allora la rete è detta dinamica, altrimenti statica. Un collegamento di retroazione immette l'uscita computata da un neurone al tempo t come ingresso al tempo $t+1$ di un neurone appartenente allo stesso strato o ad uno precedente.

Nel caso della *RN* statica composta da n variabili di ingresso, n_h neuroni nello strato nascosto e da un neurone nello strato di uscita e quindi da una sola variabile di uscita, la rappresentazione analitica della *RN* risulta essere

$$\hat{y}(t) = f\left[\beta_0 + \sum_{j=1}^{n_h} \beta_j f_j \left(w_{0j} + \sum_{i=1}^{n} x_i(t) w_{ij}\right)\right] \tag{1}$$

dove l'unico neurone dello strato di uscita della rete è caratterizzato da una funzione di attivazione lineare con i parametri $\{\beta_0,\ldots,\beta_{n_h}\}$.

Nel caso in cui si considerino *RN* dinamiche allora la variabile di uscita della *RN* è non solo funzione delle variabili di ingresso contemporanee ma anche di quelle precedenti: $\hat{\mathbf{y}}(t)$ è funzione non solo di $\mathbf{x}(t)$ ma anche di $\mathbf{x}(t-1), \mathbf{y}(t-1)$ e $\hat{\mathbf{y}}(t-1)$; i collegamenti di retroazione introducono una componente di ritardo o di memoria nella struttura analitica della *RN* . Tipicamente un *RN* con queste caratteristiche, nota con il nome di Elmann, caratterizzata da uno strato nascosto, un'uscita e collegamenti di retroazione nello strato nascosto, è rappresentata dalla seguente funzione:

$$\hat{y}(t) = f\left[\beta_0 + \sum_{j=1}^{n_h} \beta_j f_j \left(w_{0j} + \sum_{i=1}^{n} x_i(t) w_{ij} + \sum_{k=1}^{n_h} \lambda_{kj} o_k(t-1)\right)\right] \tag{2}$$

dove

$$o_k(t) = f_k\left(w_{0k} + \sum_{i=1}^{n} x_i(t) w_{ik}\right) .$$

Le due architetture sopra introdotte possono essere generalizzate introducendo collegamenti diretti tra le variabili di ingresso e di uscita che rappresentano una componente lineare all'interno della architettura:

$$\hat{y}(t) = \alpha_0 + \sum_{l=1}^{n} \alpha_l x_l(t) + f\left[\beta_0 + \sum_{j=1}^{n_h} \beta_j f_j\left(w_{0j} + \sum_{i=1}^{n} x_i(t) w_{ij} + \sum_{k=1}^{n_h} \lambda_{kj} o_k(t-1)\right)\right] . \tag{3}$$

I modelli descritti in (1) sono "annidati" nella classe dei modelli descritti in (2) che a loro volta sono "annidati" nella classe descritta in (3).

Tra le architetture neurali utilizzate in letteratura vi sono anche quelle con più di uno strato nascosto. Una *RN* statica con due strati nascosti ed una variabile di uscita è descritta dalla funzione

$$\hat{y}(t) = f\left\{v_0 + \sum_{k=1}^{n_{h_1}} v_k f_k\left[\beta_{0k} + \sum_{j=1}^{n_h} \beta_{jk} f_j\left(w_{0j} + \sum_{i=1}^{n} x_i(t) w_{ij}\right)\right]\right\} .$$

Si noti come i modelli con due strati nascosti non siano "annidati" nei modelli con uno strato nascosto.

2.1 Proprietà delle Reti Neurali

Le *RN* hanno attirato l'interesse dei ricercatori per le loro proprietà di approssimazione universale, cioè per la loro capacità di ricostruire la legge che descrive un dato fenomeno.

In questa Sezione discuteremo le proprietà delle *RN* che risultano essere interessanti per il loro impiego nell'analisi delle serie storiche. In questa prospettiva la capacità di approssimazione e la capacità di generalizzazione di una *RN* devono essere discusse. Nel primo caso si tratta della capacità di una *RN* di approssimare una data funzione, la performance si riferisce a prestazioni valutate sull'insieme di osservazioni utilizzato per la stima dei parametri del modello neurale. La capacità di generalizzazione delle *RN* si riferisce invece alle prestazioni della *RN* su un insieme di osservazioni diverso da quello utilizzato per la stima. Una buona capacità di approssimazione non implica buone prestazioni in fase previsiva, infatti anche le *RN* sono soggette a fenomeni quali *over fitting* e *sovra parametrizzazione* che consentono modesti errori di approssimazione ma non garantiscono bassi errori in fase previsiva.

2.1.1 Capacità di Approssimazione

La letteratura sulle proprietà di approssimazione delle *RN* riguarda soprattutto le reti statiche; per le reti dinamiche, a causa della loro complessa struttura, i risultati a disposizione sono meno soddisfacenti.

Un primo insieme di risultati riguarda le capacità di approssimazione di funzioni continue definite su un insieme compatto appartenente ad uno spazio con dimensione finita, e.g., $f(\mathbf{x}) : K \subset \Re^n \rightarrow \Re^q$. È stato dimostrato che una *RN* feedforward con tre strati, funzione di attivazione sigmoide (una funzione $f(a) : \Re \rightarrow \Re$ tale che $\lim_{a\rightarrow-\infty} f(a) = 0$, $\lim_{a\rightarrow\infty} f(a) = 1$) nello strato nascosto ed un numero sufficientemente elevato di neuroni può approssimare funzioni continue definite su un insieme compatto di $\Re^n$, [11, 22]. Altre funzioni di attivazione possono essere impiegate nello strato nascosto della *RN* al fine di ottenere le proprietà di approssimazione: funzione limitata e continua [22], funzione continua, non polinomiale e limitata all'infinito [30], funzione limitata, non polinomiale e continua a tratti [27]. Si noti come una delle caratteristiche fondamentali delle funzioni di attivazione al fine di stabilire il risultato sia la presenza di un parametro soglia.

Per quanto riguarda la definizione del numero di strati nascosti della *RN* , i risultati teorici garantiscono che uno strato nascosto è sufficiente a garantire la proprietà di approssimazione; non c'è quindi differenza tra le capacità di approssimazione di *RN* con uno o più strati nascosti. Le differenze riguardano altri aspetti quali, ad esempio, le capacità di generalizzazione. Studi comparativi, si veda [12], hanno mostrato che reti con uno o due strati nascosti aventi lo stesso numero di parametri hanno in media prestazioni equiparabili anche se le reti con due strati nascosti sono più difficili da stimare. Questi risultati possono essere estesi al caso di un funzionale reale continuo definito su un insieme compatto di uno spazio lineare normato. La proprietà è stata stabilita in [10] richiedendo che la funzione di attivazione sia di tipo Tauber-Wiener (TW)[1] che, nel caso delle funzioni continue, implica la sua non polinomialità, si veda [10]. Analoghi risultati sono stati ottenuti per operatori continui e nonlineari, si veda [10].

Definite le caratteristiche delle funzioni di attivazione che garantiscono le proprietà di approssimazione, è interessante valutare il livello di accuratezza che può essere raggiunto dalle *RN* . Per *RN* con funzione di attivazione sigmoide, è stato dimostrato che una rete con tre strati, n variabili di ingresso e n_h neuroni nello strato nascosto raggiunge valori dell'errore quadratico medio dell'ordine di $O(1/n_h)$, si veda [5]; per una espansione in serie con n_h termini, l'errore quadratico medio è dell'ordine di $O(1/n_h^{2/n})$. Il livello di accuratezza raggiungibile dalle *RN* non dipende quindi dalla dimensione del vettore delle variabili di ingresso ed è superiore per vettori delle variabili di ingresso di dimensione maggiore di 2. Modelli più tradizionali, quali le espansioni trigonometriche, polinomiali, spline, raggiungono lo stesso livello di errore

[1] **Definizione**: Se una funzione $f(a) : \Re \rightarrow \Re$ è tale per cui tutte le combinazioni lineari

$$\sum_{i=1}^{N} w_i f(\lambda_i x + \theta_i)$$

con $\lambda_i \in \Re$, $\theta_i \in \Re$, $w_i \in \Re$, $i = 1, \ldots, N$ sono dense in ogni $C([a_1, a_2])$ allora la funzione $f(a)$ è detta di *Tauber-Wiener.*

quadratico medio con un numero più elevato di parametri. Questi risultati sono stati estesi a *RN* con funzione di attivazione non sigmoide in [21]; in questo caso è stato dimostrato che l'errore quadratico medio decresce al crescere del numero dei neuroni nello strato nascosto nell'ordine di $O(1/\sqrt{n_h})$.

2.1.2 Capacità di Generalizzazione

Oltre alle capacità di approssimazione di una *RN* è interessante valutare la sua capacità di generalizzazione cioè il suo comportamento su un insieme di osservazioni non utilizzate in fase di stima. In questa Sezione viene discusso come l'errore di generalizzazione dipenda in parte dalle insufficiente capacità di approssimazione della *RN* (errore di approssimazione) ed in parte dalla limitatezza dell'insieme delle osservazioni (errore di stima), si veda in proposito [35].

Si consideri il problema della stima dei parametri di una *RN* sulla base di un insieme di osservazioni $D_T \equiv \{\mathbf{x}(t), \mathbf{y}(t)\}$, $t = 1, \ldots, T$, ottenute campionando una funzione di distribuzione $P(\mathbf{x}, \mathbf{y})$ incognita. Se indichiamo con $\hat{y} = \mathcal{N}(\mathbf{x})$ la funzione realizzata dalla *RN* , la stima del modello $\mathcal{N}(\mathbf{x})$ si riconduce alla minimizzazione di una funzione di costo che misura l'errore atteso, nel caso della norma L_2 si ha:

$$E\left[(\mathbf{y} - \mathcal{N}(\mathbf{x}))^2\right] = \int\int P(\mathbf{x}, \mathbf{y}) \, (\mathbf{y} - \mathcal{N}(\mathbf{x}))^2 \, d\mathbf{x}d\mathbf{y} \; .$$

Poichè $P(\mathbf{x}, \mathbf{y})$ è incognita, la stima della *RN* si riconduce alla minimizzazione dell'errore quadratico medio campionario:

$$C(\mathcal{N}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{2} \sum_{i=1}^{n_o} (y_i(t) - \hat{y}_i(t))^2$$

che converge per $T \to \infty$ a $E\left[(\mathbf{y} - \mathcal{N}(\mathbf{x}))^2\right]$.

Un modello neurale $\mathcal{N}(\mathbf{x})$ appartenente ad una classe di funzioni $\mathcal{F}$ viene scelto sulla base dell'insieme delle osservazioni D_T minimizzando $C(\mathcal{N})$:

$$\mathcal{N}^* = \arg \min_{\mathcal{N} \in \mathcal{F}} C(\mathcal{N}) \; .$$

Si noti che nel caso di modelli non correttamente specificati la classe di funzioni $\mathcal{F}$ non include la funzione che rappresenta la vera distribuzione di probabilità. Il problema di minimizzazione così formulato è mal posto in quanto, in base alla scelta di $\mathcal{F}$, si possono avere infinite soluzioni.

Se restringiamo la classe di funzioni $\mathcal{F}$ alla famiglia delle *RN* feedforward con uno strato nascosto e n_h neuroni, si ottiene $\mathcal{N}^*_{T,n_h}$, dove $\mathcal{N}_{T,n_h}$ sta ad indicare che $\mathcal{N}$ è una *RN* con n_h neuroni ed è stimata con T osservazioni. Seguendo l'analisi in [35], è possibile fornire un limite superiore all'errore di generalizzazione di una *RN* tramite due componenti:

$$E\left[(\mathbf{y} - \mathcal{N}^*_{T,n_h}(\mathbf{x}))^2\right] = \int (\mathbf{y} - \mathcal{N}^*_{T,n_h}(\mathbf{x}))^2 P(\mathbf{x}) d\mathbf{x} \leq \underbrace{\epsilon(n_h)}_{\text{err. approx.}} + \underbrace{\nu(T, n_h)}_{\text{err. stima}} \; .$$

Il limite superiore per l'errore di generalizzazione è dato dalla somma dell'errore di approssimazione $\epsilon(n_h)$ e dell'errore di stima $\nu(T, n_h)$. L'errore di approssimazione non dipende dal numero di osservazioni T, dipende solo dalle capacità di approssimazione della *RN* e gode della proprietà $\lim_{n_h \to \infty} \epsilon(n_h) = 0$, l'errore di stima $\nu(T, n_h)$ è invece una funzione crescente di n_h e decrescente di T.

Si può notare come riguardo alla capacità di generalizzazione esista un trade-off tra il numero delle osservazioni T ed il numero di neuroni n_h della *RN* . Fissato T, se n_h aumenta, l'errore di approssimazione $\epsilon(n_h)$ diminuisce ma l'errore di stima $\nu(T, n_h)$ aumenta. In generale, architetture semplici hanno alti errori di approssimazione e bassi errori di stima, per modelli complessi si verifica il contrario.

2.2 Tecniche di Stima

Gli algoritmi di apprendimento, ovvero i metodi di stima dei parametri delle *RN* , possono essere *supervisionati* o *non supervisionati*. Ipotizziamo di utilizzare una *RN* per lo studio di un fenomeno descritto da una generica funzione $\mathbf{y} = f(\mathbf{x})$ con $\mathbf{y} \in \Re^{n_o}$ e $\mathbf{x} \in X \subset \Re^n$. Gli algoritmi per la stima dei parametri della *RN* sono detti *supervisionati* se l'insieme delle osservazioni disponibili è composto da coppie $\{\mathbf{x}(t), \mathbf{y}(t)\}$, $t = 1, \ldots, T$, cioè se sono disponibili osservazioni sia per le variabili di ingresso che di uscita. In questo caso è possibile definire una funzione di costo $C(\mathbf{w})$, funzione dei parametri della rete, che viene minimizzata tramite algoritmi stocastici. Ad esempio, nel caso della norma L_2 la funzione di costo è data dall'errore quadratico medio

$$C(\mathbf{w}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{2} \sum_{i=1}^{n_o} (y_i(t) - \hat{y}_i(t))^2 = \frac{1}{T} \sum_{t=1}^{T} C(\mathbf{w}, t) \ , \tag{4}$$

dove $C(\mathbf{w}, t)$ è l'errore quadratico al tempo t. Gli algoritmi che utilizzano una misura dell'errore che è compiuto dalla *RN* nell'approssimare la funzione $f(\mathbf{x})$ sono detti supervisionati perchè le osservazioni $\{\mathbf{y}(t)\}$ supervisionano il comportamento della *RN* penalizzando $C(\mathbf{w})$ quando l'errore quadratico è elevato.

Una *RN* statica o dinamica è una funzione definita da un insieme di parametri (pesi) che devono essere stimati su un insieme di osservazioni. Le difficoltà che si incontrano nella stima dei parametri di una *RN* , rispetto a modelli econometrici più classici, sono date dall'ampiezza dello spazio dei parametri e dalla nonlinearità della funzione da minimizzare. A causa di questa caratteristica, il processo di minimizzazione risulta essere particolarmente sensibile ai valori iniziali assegnati ai parametri della *RN* . Diversi metodi di inizializzazione possono essere adottati: l'inizializzazione con valori casuali uniformemente distribuiti in un intorno dello zero è la più semplice, tecniche più elaborate si basano sulla linearizzazione delle funzioni di attivazione della rete neurale e sull'inizializzazione dei parametri con i valori stimati del modello linearizzato, in proposito si veda [26].

L'algoritmo di stima introdotto per primo in letteratura per le reti statiche è quello di *Back Propagation*, si veda [42]. L'algoritmo è del tipo gradiente discendente. Ogni

parametro della RN è modificato proporzionalmente alla derivata parziale dell'errore quadratico istantaneo $C(\mathbf{w}, t)$ rispetto al parametro:

$$w_{ij}(t) = w_{ij}(t-1) - \eta_t \frac{\partial C(\mathbf{w}, t)}{\partial w_{ij}}$$

dove

$$\frac{\partial C(\mathbf{w}, t)}{\partial w_{ij}} = -\delta_j(t) o_j(t)$$

e

$$\delta_j(t) = \begin{cases} f_j'(a_j(t)) \cdot (y_j(t) - \hat{y}_j(t)) & \text{se } j \text{ è un neurone dello strato di uscita} \\ f_j'(a_j(t)) \cdot \sum_{k=0}^{n_h} \delta_k(t) w_{jk} & \text{se } j \text{ è un neurone degli strati intermedi} \end{cases}$$

η_t è il passo di apprendimento che può essere o costante o decrescente in t; il passo di apprendimento può essere anche funzione dell'errore osservato. Dalla descrizione dell'algoritmo si intuisce il motivo del nome Back Propagation: l'errore allo strato di uscita della RN è pari a $y_j(t) - \hat{y}_j(t)$, per gli strati intermedi l'errore in uscita è propagato all'indietro moltiplicato per i parametri dello strato precedente. Questo metodo di stima consente di stimare i parametri di una RN calcolando $\delta_k(t)$ per ogni strato senza ricorrere all'errore in uscita della RN . L'algoritmo appartiene alla famiglia degli algoritmi di Robbins-Monro. Metodi di stima più classici, quali il metodo dei minimi quadrati nonlineari, possono essere utilizzati per la stima dei parametri di una RN . In alcune applicazioni il metodo dei minimi quadrati nonlineari è impiegato insieme all'algoritmo di Back Propagation: l'algoritmo Back Propagtion fornisce una inizializzazione per uno *smoothing* tramite i minimi quadrati nonlineari, si veda in proposito [23].

Alcune varianti dell'algoritmo di Back Propagation sono possibili introducendo un termine di momento nella legge che aggiorna i parametri della RN :

$$\Delta w_{ij}(t) = -\eta_t \frac{\partial C(\mathbf{w}, t)}{\partial w_{ij}} + \alpha \Delta w_{ij}(t-1)$$

oppure calcolando il gradiente non sulla base dell'errore istantaneo ma sulla base dell'errore totale $C(\mathbf{w})$ (*batch* Back Propagation).

In questo lavoro è stato utilizzato l'algoritmo di ottimizzazione di Levenberg-Marquardt che fornisce un metodo di ottimizzazione più sofisticato rispetto alla discesa lungo il gradiente. Nonostante che il processo di ottimizzazione sia molto più pesante da un punto di vista computazionale, questo metodo è da preferire al più utilizzato metodo di Back-propagation in quanto i tempi necessari per l'apprendimento da parte della rete sono notevolemente inferiori e pertanto giustificano l'utilizzo di un metodo più complesso. Il metodo di Levenberg-Marquardt aumenta la velocità di convergenza del processo di apprendimento in modo significativo. L'aggiornamento dei parametri in questo caso avviene secondo la seguente formula:

$$w_{ij}(t) = w_{ij}(t-1) + (\mathcal{C}(\mathbf{w}, t)\mathcal{C}(\mathbf{w}, t) + \mu I)^{-1} \mathcal{C}(\mathbf{w}, t) C(\mathbf{w}, t) \qquad (5)$$

dove C è lo Jacobiano del funzionale di costo C e μ è uno scalare. Se μ è elevato, l'espressione (5) approssima la discesa lungo il gradiente altrimenti diventa il metodo di minimizzazione di Gauss-Newton.

Altri tipi di algoritmi sono stati introdotti in letteratura: *Up-start* [15], *Gradiente Coniugato* [32], *Expectation Maximization* (EM) [2]; uno studio comparativo è fornito in [4].

Nel caso della stima dei parametri di una *RN* dinamica la minimizzazione della funzione costo è più complessa in quanto il valore di un parametro al tempo t si ripercuote sulla funzione calcolata dalla *RN* in ogni istante di tempo successivo. Gli algoritmi per la stima dei parametri di *RN* dinamiche possono essere ricondotti a due classi principali: algoritmi *locali in tempo* e *locali in spazio*. I primi utilizzano per il calcolo del gradiente della funzione di costo solo le variabili riferite ad istanti di tempo contigui, si veda [55]; i secondi calcolano le componenti del gradiente utilizzando variabili spazialmente contigue ovvero il calcolo della derivata parziale $\frac{\partial C(\mathbf{w},t)}{\partial w_{ij}}$ viene eseguito utilizzando le grandezze relative ai neuroni i e j posti ai due estremi della connessione w_{ij}, si veda [37]. Non esiste un algoritmo per il calcolo del gradiente che abbia entrambe le proprietà.

3 Reti Neurali per l'Analisi delle Serie Storiche: Selezione del Modello e Applicazioni

Come abbiamo già posto in evidenza nelle precedenti Sezioni, la *RN* deve essere opportunamente progettata a seconda dell'applicazione. L'utilizzo delle *RN* richiede quindi una attenta analisi delle variabili esplicative del fenomeno e dell'architettura.

Come mostrato nella Sezione 2.1.2, la selezione del modello è particolarmente rilevante al fine di valutare la capacità di generalizzazione delle *RN* . Questa proprietà non è affatto assicurata dai risultati di approssimazione universale ottenuti in letteratura: secondo questi studi buone prestazioni in fase di stima sono ottenute aumentando il numero dei parametri della *RN* , così facendo si può verificare il problema dell'*over-fitting* a causa della sovra parametrizzazione dell'architettura con una conseguente modesta prestazione in fase previsiva *out of sample*. Le tecniche di selezione che sono presentate in questa Sezione mirano ad individuare architetture che raggiungono un buon livello di generalizzazione. Ci sofferemeremo in particolare sulla tecnica di prewhitening che sarà utilizzata nella nostra applicazione.

Come evidenziato in [24], la selezione del modello di *RN* non può essere effettuata tramite i metodi classici dell'inferenza statistica in quanto questi richiedono la corretta specificazione del modello. Due classi di metodi per la selezione del modello neurale possono essere individuate in letteratura: metodi biologici e metodi data-based. Nel primo caso si segue una visione "evoluzionista" e la *RN* più appropriata per l'applicazione è quella che sopravvive ad una serie di prove effettuate durante la stima della *RN* : la selezione della *RN* e la stima dei suoi parametri vengono effet-

tuate contemporaneamente. Esempi di queste tecniche sono quelle basate sui metodi di pruning e sugli algoritmi genetici. Nel secondo caso si fa riferimento alla valutazione di alcuni indici che misurano le prestazioni della *RN*. Le tecniche di selezione del primo tipo hanno un fondamento teorico modesto, la loro popolarità è in gran parte dovuta alla euristica biologica e al fatto che non richiedono l'intervento dell'esperto nella definizione del modello; queste tecniche hanno comunque mostrato buone prestazioni a costo di un forte impegno computazionale. I metodi data-based hanno invece un fondamento teorico più solido.

Nella prima classe rientrano i metodi di *pruning* che prevedono la definizione di un'architettura sovra dimensionata sia in termini di variabili di ingresso che di neuroni e connessioni e l'eliminazione progressiva, durante la fase di stima, delle unità non rilevanti, si veda in proposito [51, 38]. Le tecniche di pruning più utilizzate sono quelle che si basano su una analisi della sensitività della *RN* rispetto ai singoli parametri e quelle che si basano su una funzione di costo da minimizzare con un termine di penalità connesso alla complessità della *RN*. Nel primo caso, in linea con la stima ricorsiva dei parametri della *RN*, la derivata della funzione espressa dalla *RN* rispetto ai parametri oppure la sensitività della funzione di costo rispetto alla eliminazione di un parametro vengono computate, nel caso in cui la derivata o la sensitività del costo rispetto al peso siano piccole in valore assoluto si provvede alla eliminazione del parametro. Nel caso della funzione di costo con penalità, una penalità viene assegnata ai neuroni in funzione della loro significatività all'interno della funzione computata dalla *RN*; con questo accorgimento i parametri non significativi vengono "forzati" a zero all'interno della procedura di stima. Nessun risultato sulla capacità di queste tecniche di selezionare l'architettura più efficiente è dato in letteratura.

Tra i metodi di selezione biologici troviamo quelli che si basano sugli *Algoritmi Genetici*, [20]. La logica è simile a quella che è alla base della tecnica di pruning: la *RN* viene selezionata e i suoi parametri vengono stimati a partire da una popolazione di *RN*. A questo scopo occorre che le architetture neurali siano descritte o meglio codificate: ogni architettura neurale della popolazione con le sue variabili di ingresso e connessioni viene codificata tramite una stringa binaria. Questa procedura fa sì che, anche nel caso di applicazioni molto semplici, la popolazione di partenza sia molto numerosa con un costo computazionale notevole. La selezione della *RN* e la sua stima sono fatte iterativamente valutando la prestazioni delle architetture nell'insieme di apprendimento combinando due procedure dal vago sapore evolutivo: seguendo un classico principio darwiniano, le *RN* che descrivono meglio il fenomeno sopravvivono, queste sono poi "mutate" tramite una tecnica di *crossover* secondo il principio che i figli sono meglio dei genitori. Per queste tecniche di selezione e di stima delle *RN* non si conoscono risultati di convergenza e di consistenza; è oramai un risultato consolidato in letteratura che tecniche di stima ricorsive basate sul gradiente della funzione di costo hanno prestazioni superiori rispetto alle tecniche di stima basate sugli Algoritmi Genetici, si veda in proposito [3].

La tecnica di *cross-validation* prevede di selezionare il modello neurale dividendo

l'insieme delle osservazioni in tre parti: l'insieme di apprendimento, l'insieme di cross-validation e l'insieme di test, si veda in proposito [29, 48]. Secondo tale metodo, una volta stimata la *RN* sul primo insieme di dati, il *prediction risk* (errore quadratico medio sull'insieme di test), è stimato calcolando l'errore quadratico medio sul secondo insieme di dati. L'architettura con l'errore quadratico medio meno elevato viene scelta a fini previsivi sul terzo insieme di dati; una volta scelta la *RN* , l'insieme di cross-validation viene inserito nell'insieme di apprendimento e la *RN* viene stimata di nuovo.

Per la selezione dell'architettura neurale è possibile adottare un approccio di tipo Bayesiano, si veda in proposito [9, 36]. Esso prevede la selezione del modello neurale tramite la massimizzazione della probabilità a posteriori che è possibile esprimere come prodotto di due termini: (1) la funzione di verosimiglianza $l(\mathbf{y}/\mathbf{x}, \mathbf{w})$ che rappresenta la distribuzione di probabilità delle variabili di uscita $\mathbf{y}$ condizionata dalle variabili di ingresso $\mathbf{x}$ e dal modello neurale rappresentato dai pesi $\mathbf{w}$, (2) la distribuzione di probabilità a priori dei pesi $p(\mathbf{w})$ che rappresenta un a priori sulle caratteristiche della *RN* come funzione analitica. Sulla base di queste due componenti è possibile costruire una funzione di costo da minimizzare

$$K(\mathbf{w}) = -\underbrace{\log l(\mathbf{y}/\mathbf{x}, \mathbf{w})}_{(1)} - \underbrace{\log p(\mathbf{w})}_{(2)} .$$

La funzione di verosimiglianza è usualmente costruita seguendo l'ipotesi che gli errori siano distribuiti secondo una distribuzione di probabilità gaussiana. Come nel caso dei modelli lineari, l'a priori sui pesi introduce una distorsione rispetto allo stimatore di massima verosimiglianza. Gli a priori sui pesi della *RN* possono avere funzioni diverse, generalmente essi riguardano l'accuratezza delle previsioni o la forma funzionale della *RN* ; nel primo caso gli a priori sono sulla accuratezza delle stime sfruttando le informazioni sul fenomeno, comunemente a priori entropici sono utilizzati (lo stimatore di massima entropia di Zellner può essere ottenuto anche per le *RN*). Nel secondo caso, la distribuzione a priori premia architetture che computano una funzione dal comportamento regolare; la funzione di costo è in questi casi funzione della curvatura della *RN* cioè della sua derivata seconda.

Alcuni dei criteri informativi utilizzati per selezionare modelli parametrici classici sono stati estesi alle *RN* , si veda in proposito [33, 1]. In questo caso la selezione del modello neurale si riconduce al confronto di indici che sono calcolati sull'insieme delle osservazioni su cui viene effettuata la stima dei parametri della *RN* , gli indici rappresentano una stima dell'errore di generalizzazione *out of sample*. Gli indici AIC (Akaike's Information Criterion)

$$AIC = \log \mathrm{MSE} + 2k \cdot \frac{1}{T}$$

e SIC (Schwartz's Information Criterion)

$$\mathrm{SIC} = \log \mathrm{MSE} + k \cdot \frac{\log T}{T}$$

possono essere calcolati per le RN , dove k è il numero totale di parametri della RN e MSE (*Mean Square Error*) è l'errore quadratico medio; per il loro utilizzo nelle applicazioni si veda [46, 16]. I criteri informativi sono stati estesi in [34], sotto forma del *Network Information Criterion* (NIC), a RN stimate tramite il metodo del gradiente discendente. Il NIC approssima il costo out of sample associato ad una RN tramite il costo medio calcolato sull'insieme di apprendimento ed un termine proporzionale al gradiente medio della funzione di costo:

$$NIC = C + \frac{1}{T}\mathrm{Tr}(G^* Q^{*-1})$$

dove C è il costo medio calcolato sull'insieme di apprendimento con T osservazioni, Q è la varianza del gradiente dell'errore istantaneo sull'insieme di apprendimento e G è il valore medio dell'Hessiano della funzione di costo valutato sull'insieme di apprendimento. L'applicazione del criterio NIC è computazionalmente dispendiosa. Sull'utilizzo dei criteri informativi nel caso di modelli non correttamente specificati si veda [43].

L'utilizzo del criterio *Minimum Description Length* (MDL) per la selezione del modello neurale permette di effettuare in contemporanea la stima e la selezione del modello.

Il criterio MDL si basa sul principio di Ockham in base al quale il modello più semplice che "spiega" l'insieme delle osservazioni è da preferire. A questo fine è necessario definire una misura della complessità del modello neurale. Una misura comunemente utilizzata per valutare la complessità di un modello è il logaritmo della funzione densità di probabilità che lo rappresenta e che genera i dati $\mathbf{y}$, ($\log p(\mathbf{y}/\mathbf{x}, \mathcal{N})$ *log-verosimiglianza*). Se un modello viene valutato sulla base di questa sola grandezza, il criterio di selezione prevede la massimizzazione di $p(\mathbf{y}/\mathbf{x}, \mathcal{N})$ rispetto ai parametri $\mathbf{w}$ che caratterizzano $\mathcal{N}$. Questa procedura è la classica stima di *Massima Verosimiglianza* (MV) che soffre dell'inconveniente di privilegiare modelli sovra parametrizzati che bene riproducono l'insieme delle osservazioni utilizzate in fase di stima, si veda in proposito [40]. Al fine di evitare questo inconveniente, il principio MDL prevede di valutare un modello sulla base anche di una seconda componente che misura la complessità del modello stesso.

Secondo il principio MDL la scelta del modello avviene definendo una funzione di costo il cui minimo globale corrisponde al modello "migliore". La funzione di costo adottata è

$$M(\mathcal{N}) = L(\mathbf{y}/\mathbf{x}, \mathcal{N}) + L(\mathcal{N}) \ , \tag{6}$$

dove il termine non negativo $L(\mathcal{N})$, la lunghezza di codice associata a $\mathcal{N}$, misura la complessità di $\mathcal{N}$ ed è tale da soddisfare la disuguaglianza di Kraft $\sum_{\mathcal{N}} 2^{-L(\mathcal{N})} \leq 1$. Se l'insieme dei modelli neurali presi in esame avesse cardinalità finita o numerabile e a tale insieme fosse associato un a priori $p(\mathcal{N})$ allora si otterrebbe

$$L(\mathcal{N}) = -\log p(\mathcal{N}) \ , \quad L(\mathbf{y}/\mathbf{x}, \mathcal{N}) = -\log p(\mathbf{y}/\mathbf{x}, \mathcal{N}) \ .$$

La minimizzazione di (6) è equivalente in questo caso al metodo di stima *Maximum A Posteriori* (MAP). Se non viene definito un a priori allora il criterio MDL si riconduce alla stima di MV.

Quando si può scrivere esplicitamente la funzione $M(\mathcal{N})$ il metodo MDL si dice *non predittivo*. Per modelli con k parametri e se i residui sono distribuiti come una variabile casuale normale, allora al tendere di $T \to \infty$, il valore di $M(\mathcal{N})$ coincide con il SIC:

$$M(\mathcal{N}) = \mathrm{SIC}(\mathcal{N}) = \log \mathrm{MSE} + k \cdot \frac{\log T}{T} \ .$$

Per la maggior parte dei modelli parametrici è possibile scrivere esplicitamente le componenti della funzione di costo in (6); nel caso di modelli non parametrici il compito può essere molto difficile (per un applicazione alla selezione delle *RN* si veda [17]). In questo caso è possibile utilizzare l'algoritmo MDL *predittivo* che computa $M(\mathcal{N})$ in linea con la stima della *RN* , si veda [41]. Per una sua implementazione e applicazione alla selezione delle *RN* nell'ambito dell'analisi delle serie storiche si veda [8].

I motivi che fanno preferire questo metodo di selezione del modello neurale rispetto ad altri metodi sono molteplici, si veda [6]. Il criterio MDL è consistente e, se il modello è correttamente specificato, $f \in \mathcal{F}$, si verifica che

$$\lim_{T \to \infty} \mathcal{N}_T = f \text{ con probabilità } 1 \ .$$

Il criterio MDL è consistente anche se il modello non è correttamente specificato a condizione che la funzione f sia approssimabile da una sequenza di funzioni della famiglia $\mathcal{F}$.

Per concludere la rassegna delle metodologie che possono essere seguite nella definizione di architetture neurali occorre ricordare i metodi classici che si basano sulla *verifica delle ipotesi*. La principale limitazione di questo approccio riguarda la *potenza* della tecnica. In tale ambito sono state proposte alcune procedure per identificare quali siano i neuroni ridondanti dello strato nascosto, si veda [25]. Il problema principale che si incontra nell'applicare questo approccio è dato dalla presenza di *nuisance parameters* cioè di parametri del modello che sono specificati solo sotto l'ipotesi alternativa. Per modelli neurali ciò si verifica in due casi. Se si vuole verificare la corretta specificazione di un modello lineare si usa confrontarlo con una *RN* del tipo in (3). In questo caso l'ipotesi nulla è definita da $\{\beta_j, j = 1, \ldots n_h\} = 0$ e l'alternativa da $\beta_j \neq 0$; i nuisance parameters sono w_{ij}. Un altro caso in cui si incontra questo problema è quello in cui si vuole valutare il numero di unità nascoste della *RN* . In particolare, si consideri una rete con n_h unità nascoste, $\mathbf{w}_j \neq 0$, $j = 1, \ldots, n_h$, ed una rete con n_{h_1} unità nascoste in più cioè $\mathbf{w}_j \neq 0$, $j = 1, \ldots, n_h + n_{h_1}$. L'ipotesi nulla in questo caso è $H_0 : \mathbf{w}_j = 0$, $j = n_h + 1, \ldots, n_{h_1}$ e l'alternativa è $H_a : \mathbf{w}_j \neq 0$, $j = n_h + 1, \ldots, n_{h_1}$; i nuisance parameters sono $\mathbf{w}_j$, $j = n_h + 1, \ldots, n_{h_1}$. Per ovviare a questo inconveniente diversi accorgimenti possono essere seguiti, si veda in proposito [47, 44].

3.1 Applicazioni all'analisi delle serie storiche

Nell'ambito econometrico e dell'analisi delle serie storiche abbiamo sia studi di natura applicativa che di natura teorica: le *RN* sono considerate "ipermodelli" nonlineari capaci di racchiudere al loro interno modelli particolari. Soprattutto grazie al contributo di H. White [52, 53, 24] si sono avuti studi metodologici che hanno introdotto nell'ambito delle *RN* problematiche proprie della statistica inferenziale. Come abbiamo già posto in evidenza, le *RN* sono per loro natura modelli non correttamente specificati e quindi il framework più appropriato per la loro analisi risulta essere quello proposto in [54]; sul fronte dei test delle ipotesi e della selezione del modello i risultati classici dell'inferenza statistica sulla significatività dei parametri non possono essere estesi. Risultati positivi sono stati invece ottenuti riguardo agli algoritmi utilizzati per la stima dei parametri della *RN*, in [24] è stato dimostrato che algoritmi del tipo di Newton, gradiente discendente, Back Propagation sono consistenti e sono asintoticamente equivalenti ad uno stimatore del tipo *Nonlinear Least Squares*.

Lo studio delle *RN* ha dato contributi anche all'analisi econometrica; in proposito vale la pena di ricordare il test di nonlinearità tramite *RN* proposto in [25, 47]: le *RN* con collegamento diretto ingresso/uscita vengono stimate, la presenza di nonlinearità nella serie storica è testata valutando la capacità dei neuroni dello strato nascosto di modellare i residui ottenuti dal modello lineare.

Le applicazioni sono soprattutto alle serie storiche finanziarie con osservazioni giornaliere/infragiornaliere, per un quadro delle principali aree di applicazione si veda [39, 50]. Data la quantità di lavori pubblicati in questo campo, una rassegna delle applicazioni non avrebbe alcun significato. La maggior parte delle applicazioni lascia abbastanza perplessi: scarsa attenzione è diretta alla selezione del modello neurale, spesso la strategia di selezione è del tipo *trial & error* o della migliore prestazione previsiva, si veda ad esempio [46], molta attenzione è invece dedicata all'aspetto previsivo ma i risultati spesso non sono convincenti, solo raramente infatti una analisi comparata con le tecniche tradizionali è fornita. L'analisi dei residui e della capacità della *RN* di processare informazioni ha avuto infine uno spazio molto limitato in letteratura. Nella maggior parte degli studi l'approccio è del tipo serie storiche: l'insieme informativo per la descrizione del fenomeno è fornito dalla serie storica, solo raramente in un contesto multivariato; questo aspetto fa sì che le *RN* siano spesso confrontate in ambito finanziario con *trading rules* più tradizionali in una ottica puramente previsiva generando l'impressione che le *RN* siano l'ultimo ritrovato dell'analisi tecnica finanziaria. Qui di seguito ci soffermiamo su alcune delle applicazioni più interessanti.

Le *RN*, selezionate in base all'indice SIC, sono impiegate in [16] per la previsione giornaliera dei rendimenti associati all'indice Dow Jones sulla base dei rendimenti osservati; le loro prestazioni sono confrontate con trading rules lineari, del tipo *moving average* e GARCH-M. L'analisi mostra che le *RN* hanno una migliore prestazione previsiva rispetto ai modelli lineari e GARCH-M, l'accuratezza è superiore di circa il 13% e del 32% qualora i segnali di vendita e di acquisto siano inseriti nell'insieme

informativo.

Una analisi multivariata sulle serie storiche dei cambi è fornita in [23]. In questo caso una tecnica ricorsiva di stima basata sull'algoritmo di Newton è utilizzata per inizializzare la stima della *RN* che viene effettuata tramite una procedura di tipo Nonlinear Least Squares; la *RN* è selezionata tramite il criterio MDL predittivo. Il confronto con modelli ARMA multivariati mostra che le *RN* hanno prestazioni superiori sia riguardo alla previsione in valore assoluto che riguardo alla previsione della direzione. Un'analisi multivariata tramite *RN* di indicatori macroeconomici su base trimestrale è fornita in [45]; i risultati mostrano che le *RN* non migliorano in modo sostanziale le prestazioni dei modelli lineari.

Uno degli aspetti che lascia più perplessi nelle applicazioni prese in esame è da ricercare nella assenza di una spiegazione dei risultati ottenuti utilizzando le *RN* : il successo o il fallimento delle *RN* viene solitamente ascritto alla presenza o meno di nonlinearità nella serie storica. Nei tre casi presi in esame l'analisi si riconduce ad un esercizio molto costoso in termini computazionali con un confronto tra un numero molto elevato di modelli neurali e lineari. Le finalità investigative della analisi non sono molto chiare.

In [7] abbiamo utilizzato le *RN* per predire il tasso di medio di aggiudicazione dei BOT emessi in asta dal Governo Italiano. L'analisi è sviluppata sia in un contesto univariato tramite modelli autoregressivi che nel caso di un insieme informativo esteso comprendente variabili di politica monetaria e del mercato monetario. L'analisi è particolarmente interessante perchè la serie storica mostra, a causa della uscita della lira dallo SME nel settembre 1992, un comportamento differenziato: prima del settembre 1992 la serie storica mostra un andamento abbastanza lineare, dopo il settembre 1992 la serie storica mostra la presenza di forti nonlinearità. Queste caratteristiche della serie storica hanno permesso di valutare la prestazione delle *RN* e dei modelli lineari a seconda che la stima sia effettuata sul periodo caratterizzato da nonlinearità o meno. L'analisi è stata effettuata considerando due insiemi informativi, uno con più dinamica dell'altro (più variabili ritardate). L'analisi mostra che le *RN* hanno prestazioni superiori rispetto ai modelli lineari riguardo alla capacità di processare informazioni e di fare previsioni qualora i modelli siano stimati includendo nell'insieme di stima della *RN* il periodo caratterizzato da nonlinearità e soprattutto quando l'insieme informativo con minore dinamica è preso in esame.

4 Analisi delle serie storica del BTP future

La metodologia per l'analisi delle serie storiche tramite *RN* proposta nelle Sezioni precedenti è stata applicata all'analisi della serie storica del prezzo del BTP future 10 anni quotato al LIFFE di Londra. La serie è composta dai prezzi alla chiusura dal 9 Giugno 1998 al 30 Ottobre 1998, vedi Fig. 1.

Poichè gli algoritmi di stima delle *RN* ed i metodi per la loro selezione prevedono che la serie storica sia stazionaria abbiamo scelto di analizzare la serie storica delle

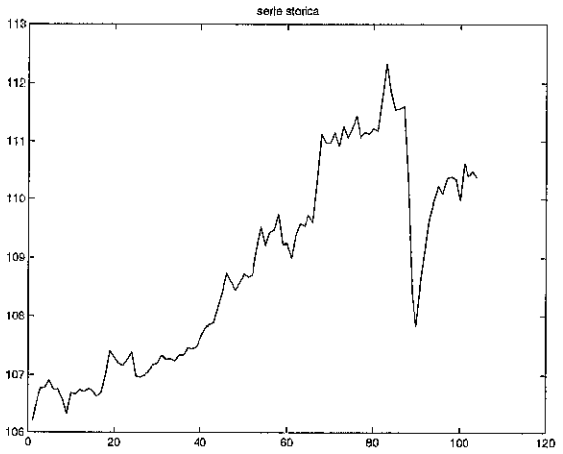

Figure 1: Serie storica BTP-future dal 9 Giugno 1998 al 30 Ottobre 1998

differenze prime, vedi Fig. 2, la serie storica dei livelli non è infatti stazionaria.

La stazionarietà della serie alle differenze è stata verificata valutando che i coefficenti della sua funzione di autocorrelazione SACF siano nell'intervallo $[-0.05, 0.05]$.

La scelta di utilizzare le differenze dei livelli della serie originaria e non dei logaritmi, come usualmente viene fatto in ambito finanziario, è dettata dal fatto che nel secondo caso gli algoritmi di stima del tipo Back Propagation sembrano funzionare in modo più efficiente, si veda in proposito [7].

Nell'utilizzare le *RN* per l'analisi della serie storica occorre trattare i dati in modo opportuno. I neuroni sono infatti caratterizzati da funzioni di attivazione sensibili in un intorno del parametro soglia; per questo motivo le variabili di ingresso della *RN* sono state scalate nell'intervallo $[-1, 1]$.

Nella selezione del modello neurale abbiamo utilizzato una metodologia che si basa sulle tecniche di bootstrap e di prewhitebning. Illustriamo brevemente queste due tecniche in due sottosezioni.

4.1 Bootstrap

Quando è disponibile solo un numero limitato di dati può risultare conveniente ricorrere ad un algoritmo che genera l'insieme di dati da utilizzare per le predizioni. A tal fine il metodo del *bootstrap*, introdotto da Efron in [14] si è rivelato appropriato producendo risultati molto interessanti. Per uno studio approfondito di tale metodo si veda [19, 13, 18]. Il metodo del *bootstrap* fornisce informazioni sull'incertezza presente nella stima di parametri basata su variabili indipendenti ed identicamente distribuite (i.i.d.).

Il problema può essere formulato nel seguente modo. Siano $\{x_1, \ldots, x_n\}$ n osservazioni indipendenti ottenute da una distribuzione $F(x)$. Un parametro $\theta = g(F())$

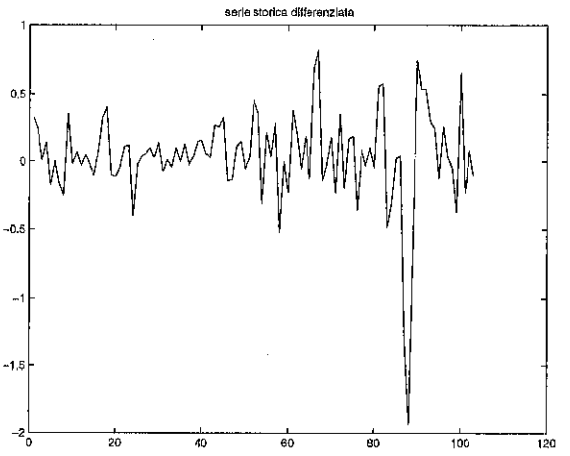

Figure 2: Serie storica alle differenze

viene stimato sulla base delle n osservazioni $\hat{\theta} = \hat{\theta}(\{x_1, \ldots, x_n\})$ ma durante la stima di tale parametro, la funzione distribuzione F non è nota. L'idea principale del metodo del *bootstrap* è di sostituire la funzione distribuzione $F(x)$ con una funzione empirica $F_n(x)$. Possiamo così riassumere il metodo nei seguenti due passi:

- il parametro θ viene sostituito con il parametro $\tilde{\theta}$ che dipende dalla funzione empirica $F_n(x)$

$$\begin{array}{l} F_n(x) = \frac{\#\{x_i \leq x\}}{n} \\ \tilde{\theta} = g(F_n(\cdot)) \end{array} \tag{7}$$

  Si noti che i dati in (7) sono noti poichè $x_1, \ldots, x_n$ sono osservazioni.

- Dalla funzione distribuzione $F_n(x)$ si simula lo stesso numero di osservazioni indipendenti ottenendo così gli elementi

$$x_1^*, \ldots, x_n^*. \tag{8}$$

  Questi elementi sono scelti in modo casuale dall'insieme originale: alcuni elementi possono essere presenti più volte ed altri possono non essere presenti affatto. Adesso è possibile eseguire la stima dei parametri

$$\hat{\theta}^* = \hat{\theta}(x_1^*, \ldots, x_n^*) \tag{9}$$

  utilizzando lo stesso stimatore del problema reale.

Questo procedimento può essere rappresentato con il seguente diagramma:
INIZIALIZZAZIONE

$x_1, \ldots, x_n$ osservazioni i.i.d. di $F(x)$
$\hat{\theta} = \hat{\theta}(x_1, \ldots, x_n)$

$\tilde{\theta} = g(F_n(\cdot))$ da $\theta = g(F(x))$

BOOTSTRAP

for $i = 1, \ldots, NB$

CAMPIONAMENTO

for $j = 1, \ldots, n$

$k = \text{int}(1 + n \times \text{random}())$

$x_j^* = x_k$

end of j-loop

$\hat{\theta}^* = \hat{\theta}(x_1^*, \ldots, x_n^*)$

end of i-loop

RISULTATI

Il metodo del bootstrap può essere utilizzato per migliorare le capacità di predizione delle *RN* creando un insieme di addestramento di dimensioni maggiori. La rete viene così addestrata su un insieme contenente un numero maggiore di dati e pertanto è in grado di estrarre più informazioni. D'altra parte, avendo a disposizione un insieme di dimensioni maggiori è possibile utilizzare una rete con un'architettura più complessa ed un numero più elevato di variabili esogene in ingresso alla rete.

4.2 Pre-whitening

Nel caso in cui il problema di predizione sia caratterizzato da serie storiche contenenti pochi dati ma allo stesso tempo siano disponibili diverse variabili esogene, si pone il problema della scelta di quali variabili esogene utilizzare. Infatti, utilizzare tutte la variabili a disposizione comporterebbe un numero elevato di parametri da stimare e ciò non è possibile perchè ciascuna serie storica non contiene un numero sufficiente di dati per tal fine. Il metodo del *pre-whitening* fornisce uno strumento in grado di stabilire quale delle variabili esogene a disposizione è più informativa rispetto alla variabile endogena oggetto delle predizioni. Qusta tecnica si basa sull'analisi dei residui delle predizioni ottenuti utilizzando il miglior modello univariato: le variabili esogene che contengono una correlazione più elevata sono quelle a maggior carattere informativo [49, 28]

4.3 Analisi insieme informativo esteso

Al fine di rappresentare l'andamento del prezzo del BTP future si sono considerate le seguenti variabili valutate al passo precendete: prezzo future bund 10 anni, capitalizzazione del mercato telematico dei titoli di stato, tasso di cambio lira-dollaro, indice libor, indice MIBTEL, posizioni aperte sul mercato dei futures, valori degli scambi

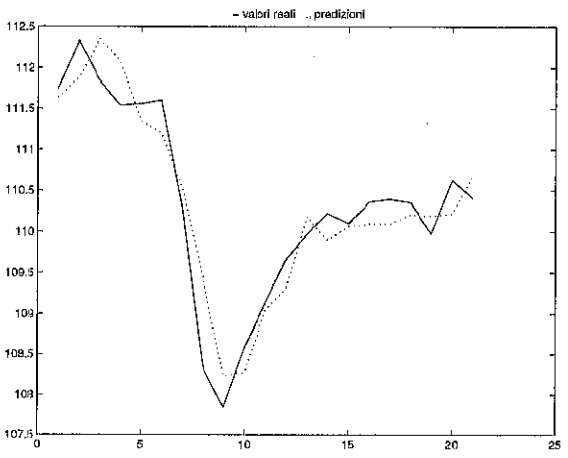

Figure 3: Predizioni del miglior modello lineare selezionato

sul mercato dei futures, valore scambi sul mercato dei titoli di stato, differenziale tassi italiani-tedeschi.

Avendo applicato il metodo *pre-whitening* a ciascuna variabile esogena a disposizione, è risultato che la serie storica relativa al **prezzo future sul bund** è quella a maggior carattere informativo e pertanto viene presa come ingresso esogeno ai modelli in esame.

4.4 Analisi con modelli multivariati

Nella selezione del miglior modello si è utilizzato l'errore quadratico medio E calcolato sull'insieme di validazione

$$E = \sqrt{\frac{1}{N_v}\sum_{i=1}^{N_v}[y(t_0+i) - \widehat{y}(t_0+i)]^2} \tag{10}$$

4.4.1 Modello lineare

Il miglior modello lineare scelto è di tipo ARMAX:

$$A(d)y(t) = B(d)u(t) + C(d)e(t) \tag{11}$$

con $n_a = 3, n_b = 5, n_c = 5$ che determina un errore quadratico medio (10) pari a

$$\boxed{\text{E=0.3914}} \tag{12}$$

La predizione del modello ARMAX é rappresentato in Fig. 3.

num. neuroni strato nascosto	E	NB
3	0.1999	2
5	0.1603	2
3	0.1577	3

Table 1: Alcuni esempi di RN usate in fase di selezione del modello.

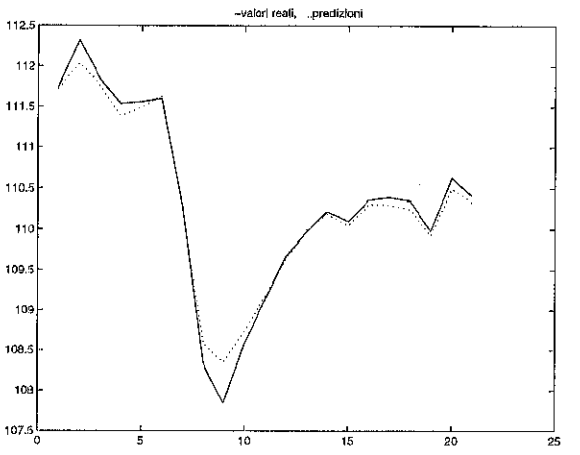

Figure 4: Predizioni rete neurale

4.4.2 Modello neurale

Per la selezione del modello neurale é stato applicato il metodo del *bootstrap* all'insieme di addestramento con diversi valori di NB e con diverse architetture di RN tutte comunque caratterizzate dal fatto di avere un unico strato nascosto. Gli errori quadratici medi di alcune delle RN analizzate in fase di selezione del modello sono riportate in Tab. 1.

Come mostrato in Tab. 1, il miglior modello neurale selezionato é stato stimato con il metodo del *bootstrap* con $NB = 3$ pertanto la rete viene addestrata su un insieme di dimensione 3 volte maggiore di quello di partenza. La RN selezionata ha 3 neuroni nello strato nascosto e 2 ingressi rappresentati dal prezzo del BTP alla valutazione precedente ed al prezzo del BUND alla valutazione precedente. L'errore di predizione (10) calcolato sull'insieme di validazione è pari a

$$\boxed{\text{E=0.1577}} \tag{13}$$

La predizione del modello neurale é rappresentato in Fig. 4.

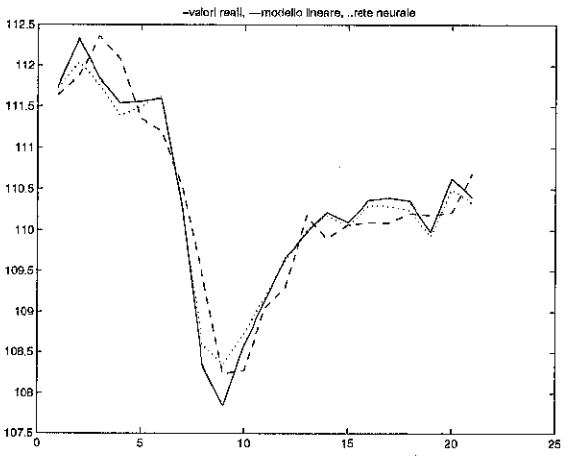

Figure 5: Predizioni modelli multivariati a confronto

4.4.3 Confronto dei modelli multivariati

In Fig. 5, sono messe a confronto le previsioni del miglior modello linerare e del miglior modello neurale ottenuti.

Gli indici che abbiamo utilizzato per la valutazione dei modelli sui dati di test sono di seguito elencati:

- *Mean Absolute Error* (MAE)

$$MAE = \frac{1}{N}\sum_{t=1}^{N}|y(t) - \hat{y}(t)|$$

- *Normalized Mean Square Error* (NMSE)

$$NMSE = \frac{\sum_{t=1}^{N}(y(t) - \hat{y}(t))^2}{\sum_{t=1}^{N}(y(t) - y(t-1))^2}$$

  l'indice NMSE confronta le prestazioni del modello con il *random walk*; se NMSE > 1 allora il modello ha prestazioni inferiori al random walk.

- *Forecast Mean Square Error* (FMSE):

$$FMSE = \frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t))^2$$

  e

$$R_{FMSE} = 1 - \frac{FMSE}{S_N^2}$$

  dove S_T^2 è la varianza della serie storica; si noti come R_{FMSE} possa essere negativo se il valore di FMSE supera quello di S_T^2.

Indici	Lineare	*RN*
MAE	0.3742	0.1086
NMSE	0.3696	0.0600
FMSE	0.1531	0.0248
R_{FMSE}	0.9484	0.9916
CF	0.4	0.05

Table 2: Coefficenti usati per valutare le prestazioni del modello lineare e neurale sui dati di test.

- *Confusion Matrix*

$$M = \text{Previsione} \overbrace{\begin{bmatrix} +,+ & +,- \\ -,+ & -,- \end{bmatrix}}^{\text{Stato del mondo}}$$

e l'indice

$$CF = \frac{\sum_{i \neq j} m_{ij}}{\sum_{i,j} m_{ij}} .$$

La confusion matrix fornisce una indicazione della capacità del modello di predire il segno della variazione della serie storica.

Per valutare se i diversi modelli hanno un comportamento previsivo diverso da un punto di vista statistico si può utilizzare il test proposto in [31].

Presentiamo adesso i test sulla previsione out of sample che abbiamo utilizzato nella nostra applicazione, vedi Tab. 2.

5 Conclusioni

LE SCRiVIAMO ALLA FINE

References

[1] S. Amari. Mathematical methods of neurocomputing. In O. E. Barndorff-Nielsen, J. L. Jensen, and W. S. Kendall, editors, *Networks and chaos – Statistical and probabilistic aspects*, pages 1–39, 1993.

[2] S. Amari. Information geometry of the EM and em algorithms for neural networks. Technical report, Department of Mathematical Engineering and Information Physics, Faculty of Engineering, University of Tokio, 1994.

[3] S. Baluja. An empirical comparison of seven iteratives and evolutionary function optimization heuristics. Technical Report CMU-CS-95-193, School of Computer Science, Carnegie Mellon University, 1995.

[4] E. Barnard. Optimizing for training neural nets. *IEEE Transactions on Neural Networks*, 3(2):232–240, 1992.

[5] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, May 1993.

[6] A. R. Barron and T. M. Cover. Minimum complexity density estimation. *IEEE Transactions on Information Theory*, 37(4):1034–1054, July 1991.

[7] E. Barucci, G. M. Gallo, and L. Landi. Linear versus Nonlinear Information Processing: a Look at Neural Networks. In M. Gilli, editor, *Computational Methods in Economics*, Advances in Computational Economics (AICE), pages 161–190. Kluwer Academic Publishers, 1995.

[8] E. Barucci and L. Landi. Reti neurali per l'analisi delle serie storiche: aspetti metodologici ed applicazioni. In *Sezione Metodologica, Congresso organizzato da Banca d'Italia, SADIBA*, November 1995.

[9] W. Buntine and A. S. Weigend. Bayesian Back-propagation. *Complex Systems*, 5:603–643, 1991.

[10] T. Chen and R. Chen. Universal approximation to non-linear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 4(6):910–918, 1993.

[11] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematical Control Signals Systems*, 2:303–314, 1989.

[12] J. de Villiers and E. Barnard. Backpropagation neural nets with one and two hidden layers. *IEEE Transactions on Neural Networks*, 4(1):136–141, 1992.

[13] T. J. Di Ciccio and J. P. Romano. A review of bootstrap confidence intervals. *Journal of the Royal Statistical Society, Ser. B*, 50:338–354, 1988.

[14] B. Efron. Bootstrap methods: Another look at the jackknife. In *The Annals of Statistics*, volume 7, pages 1–26. 1979.

[15] M. Frean. The Upstart algorithm: a method for constructing and training feed-forward neural networks. *Neural Computation*, 2:198–209, 1990.

[16] R. Gencay. The predictibility of security returns with simple technical trading rules. Preprint, 1994.

[17] C. W. L. Granger. Strategies for modelling nonlinear time-series relationships. *The Economic Record*, 69(206):233–238, 1993.

[18] D. Hinkley. Bootstrap methods. *Journal of the Royal Statistical Society, Ser. B*, 50:321–337, 1988.

[19] J.S.U. Hjorth. *Computer Intensive Statistical Methods. Validation model selection and bootstrap.* Chapman & Hall, 1994.

[20] J. H. Holland. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, MI, 1975.

[21] K. Hornik, M. Stinchcombe, H. White, and P. Auer. Degree of approximation results for feedforward networks approximating unknown mappings and their derivatives. *Neural Computation*, 6(6):1262, 1994.

[22] K. Hornik, M. Stinchombe, and H. White. Multi-layer feedforward networks are universal approximators. *Neural Networks*, 12:359–366, 1989.

[23] C.-M. Kuan and T. Liu. Forecasting exchange rates using feedforward and recurrent Neural Networks. Technical Report 92-0128, University of Illinois at Urbana-Champain, Bureau of Economic and Business Research, 1992.

[24] C.-M. Kuan and H. White. Artificial neural networks: an econometric perspective. *Econometric Rewievs*, 13:1–92, 1994.

[25] T.-H. Lee, H. White, and C. W.J. Granger. Testing for neglected non-linearity in time series models. *Journal of Econometrics*, 56:269–290, 1993.

[26] M. Lehtokangas, J. Saarinen, P. Huuhtanen, and K. Kaski. Initializing weights of a multi-layer perceptron network by using the Orthogonal Least Squares algorithm. Submitted to Neural Computation, 1994.

[27] M. Leshno, V. Ya. Lin, A. Pinkus, and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6:861–867, 1993.

[28] L. Ljung. *System identification - Theory for the User.* PTR Prentice Hall, 1999.

[29] M. Marron. A comparison of cross-validation techniques in density estimation. *The Annals of Statistics*, 15(1):152–162, 1987.

[30] H. N. Mhaskar and C. A. Micchelli. Approximation by superimposition of sigmoidal and radial basis functions. *Advances in Applied Mathematics*, 13:350–373, 1992.

[31] B. Mizrach. Forecast comparison in l_2. Technical report, Department of finance, Wharton school, 1991.

[32] M. Moller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4):525–533, 1993.

[33] J. E. Moody. The effective number of parameters: an analysis of generalization and regularization in non–linear learning systems. In *Advances in Neural and Information Processing Systems* 4. Morgan Kaufmann, 1992.

[34] N. Murata, S. Yoshizawa, and S. Amari. Network Information Criterion – determining the number of hidden units for an artificial neural network models. *IEEE Transactions on Neural Networks*, 4, 1993.

[35] P. Niyogi and F. Girosi. On the relationship between generalization error, hypothesis complexity, and sample complexity fror Radial Basis Functions. Technical Report AIM-1467, Center for Biological and Computational Learning, Department of Brain and Cognitive Scineces, Massachusetts Institute of Technology (MIT), 1994.

[36] S. J. Nowlan and G. E. Hinton. Simplifying Neural Networks by soft weight–sharing. *Neural Computation*, 4:473–493, 1992.

[37] B. A. Pearlmutter. Dynamic Recurrent Neural Networks. Technical Report CMU-CS-90-196, School of Computer Science, Carnegie Mellon University, Pittsburg, PA, 1990.

[38] R. Reed. Pruning algorithms – a survey. *IEEE Transactions on Neural Networks*, 4(5):740–747, 1993.

[39] Apostolos-Paul Refenes, editor. *Neural Networks in the capital markets.* Wiley, November 1994.

[40] J. Rissanen. Stochastic complexity. *Journal of Royal Statistical Society B*, 49(3):223–239, 1987.

[41] J. Rissanen. Information theory and neural nets. In P. Smolensky, M. Mozer, and D. Rumelhart, editors, *Mathematical Perspectives on Neural Networks.* Laurence Erlbaum Associates, 1994.

[42] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representation by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: explorations in the microstructure of cognition. Vol. 1: Foundations.* MIT, Press, 1986.

[43] C.-Y. Sin and H. White. Information Criteria for selecting possibly misspecified parametric models. Working paper, November 1992.

[44] M. B. Stinchcombe and H. White. Consistent specification testing with unidentified nuisance parameters using duality and Banach space limit theory. Technical report, Department of Economics, 0508, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA, 1993.

[45] N. R. Swanson and H. White. A model selection approach to real-time macro-economic forecasting using linear models and artificial neural networks. In *International Symposium of Forecastors*, Stockholm, Sweden, 1994.

[46] N. R. Swanson and H. White. A model selection approach to assessing the information in term structure using linear models and artificial neural networks. *Journal of Business & Economic Statistics*, 13:265–275, 1995.

[47] T. Teräsvirta, C.-F. Lin, and C. W. J. Granger. Power of the neural network linearity test. *Journal of Time Series Analysis*, 2:209–220, 1993.

[48] J. Utans and J. E. Moody. Selecting neural networks architectures via the prediction risk: application to corporate bond rating prediction. In *Proceedings of the First International Conference on Artificial Intelligence Applications on Wall Street*, 1991.

[49] W. Vandaele. *Applied Time Series and Box-Jenkins Models.* Academic Press, 1983.

[50] A. S. Weigend and N. A. Gershenfeld, editors. *Time Series Prediction: forecasting the future and understanding the past.* Addison-Wesley, 1992.

[51] A. S. Weigend, D. E. Rumelhart, and B. A. Huberman. Generalization by weight-elimination with application to forecasting. In D. S. Touretzky, editor, *Advances in Neural Information Processing*, pages 875–882. Morgan Kufmann Inc., 1991.

[52] H. White. Learning in artificial networks: a statistical perspective. *Neural Computation*, 1:425–464, 1989.

[53] H. White. *Artificial Neural Networks - Approximation & Learning algorithms.* Blackwell, Cambridge, Massachusetts – USA, 1992.

[54] H. White. *Estimation, Inference and specification analisys.* Cmabridge Univeristy Press, 1994.

[55] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280, 1989.