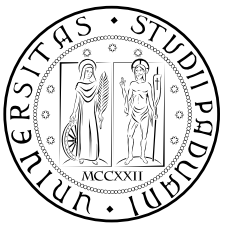


Università degli Studi di Padova

Dipartimento di Tecnica e Gestione dei Sistemi Industriali
Corso di Laurea Magistrale in Ingegneria Meccatronica

Tesi di laurea magistrale

Sviluppo di modelli dinamici black-box per convertitori DC-DC utilizzando reti neurali

Relatore: Prof. Mattavelli Paolo

Correlatore: Ing. Zilio Andrea

Studente: Bregantin Alberto
2038464

Anno accademico: 2023/2024

INDICE

0.1 *Introduzione*

Il lavoro svolto in questa tesi riguarda ciò che nella teoria dei sistemi è noto come *identificazione*. Per identificazione s'intende la determinazione di un modello sufficientemente accurato di un dato sistema a partire da un certo numero di informazioni, le quali possono essere dati ricavati a seguito di esperimenti effettuati sul sistema da identificare oppure risultati riportati nella letteratura scientifica come ad esempio le leggi della fisica. Vi sono tre tipi di identificazione a seconda del grado di conoscenza del sistema che si possiede.

- Identificazione white-box
- Identificazione gray-box
- Identificazione black-box

Nell'identificazione white-box si conosce totalmente il sistema, ovvero si conoscono interamente la struttura del modello e i parametri; ad esempio si conosce la struttura della funzione di trasferimento (grado numeratore e denominatore) e i suoi coefficienti. Pertanto l'identificazione white-box è banale dato che il modello è già noto. Nell'identificazione gray-box è nota la struttura del modello del sistema, ma non i parametri. Ad esempio è nota la struttura della funzione di trasferimento (grado del numeratore e del denominatore) ma non si ha alcuna informazione sui valori che i coefficienti possono assumere. Se nell'identificazione gray-box l'informazione che si possiede sul sistema è parziale, nell'identificazione black-box non si ha alcun tipo di informazione. Vale a dire che non si conosce né la struttura del modello né i parametri. L'unica cosa che è possibile analizzare è il comportamento ingresso-uscita.

Vi sono diversi motivi per i quali può rivelarsi necessaria l'identificazione di un dato sistema. Ad esempio le informazioni riportate sul datasheet potrebbero essere poco accurate o precise, nel caso peggiore potrebbero essere non sufficienti per poter caratterizzare il sistema. Inoltre disporre di un modello del sistema stesso è fondamentale per la progettazione del controllo, per lo studio della stabilità e per lo sviluppo di Digital Twin.

Questa tesi tratterà il problema dell'identificazione black-box di un convertitore boost sfruttando tecniche di deep-learning. Nello specifico si introdurranno alcuni concetti base del deep-learning stesso (rete neurale, addestramento, funzione di perdita, ecc …) e si discuteranno alcuni approcci noti in letteratura per la modellizzazione black-box di convertitori di potenza dc-dc. Infine verrà proposto un modello black-box per stimare la tensione di uscita di un convertitore boost mediante l'utilizzo di reti neurali feed-forward.

1

RETI NEURALI FEED-FORWARD

1.1 *Struttura di una rete neurale feed-forward*

Esistono diversi modi per modellizzare un dato sistema. In questa tesi si utilizzeranno in particolare delle tecniche di deep learning e pertanto in questo capitolo verranno introdotti i principali concetti sui quali si basano tali tecniche.

L'idea è quella di determinare delle relazioni o regole che legano determinati dati in ingresso a dei corrispondenti dati in uscita. Per chiarire meglio, si consideri ad esempio un qualunque algoritmo in esecuzione su un elaboratore. Normalmente il programmatore scrive il codice, ovvero le "regole" utilizzate per calcolare l'output, e l'utente inserisce l'input. Il problema che ora ci si pone è quello di anziché determinare l'output dagli input e dal codice, determinare il codice a partire dagli input e dai corrispondenti output. In questo modo si realizza una macchina che, oltre a fornire un output in corrispondenza di un dato input, riesce anche ad apprendere come gli stessi input sono legati ai corrispondenti output. In altri termini, la macchina non solo esegue una certa azione, ma impara anche il modo mediante il quale deve eseguirla. Il processo secondo il quale un sistema apprende le regole che legano gli output ai determinati input è noto come Machine Learning (ML).

Vi sono svariati problemi che possono essere risolti grazie al ML. Tipici esempi sono i problemi di classificazione, come associare una parola appartenente a un insieme finito di parole a un'immagine o ad una traccia audio. Si pensi in particolare a un problema in cui si deve stabilire il soggetto di un'immagine (una persona, un paesaggio o un animale). Altri tipi di problemi trattati con tecniche di ML sono quelli di regressione dove si deve determinare la funzione matematica che lega gli input agli output corrispondenti. L'identificazione black-box è di fatto un problema di regressione.

Prima di proseguire con l'illustrare come una macchina tramite un opportuno algoritmo possa "apprendere", è necessario compiere alcune osservazioni. In primo luogo è essenziale disporre di un insieme di dati, detto data set, composto da coppie *(input, output, atteso)* in modo tale da fornire alla macchina l'informazione sulla quale basarsi per apprendere. Ad esempio se si deve affrontare un problema di classificazioni di immagini, si dovrà creare un data set formato da coppie immagine-parola. Se invece il problema è di regressione, in particolare si deve determinare una funzione f che lega una variabile indipendente $x \in \mathbb{R}$ a una variabile dipendente $y \in \mathbb{R}$, $y = f(x)$, il data set sarà composto da coppie

$$x_i,\ y_i = f(x_i)$$

con $i = 1, 2, \dots N$; dove N è il numero di coppie presenti nel data set.

In secondo luogo è necessaria una metrica per quantificare se la macchina ha effettivamente appreso oppure no. A tale scopo viene definita una funzione di perdita che viene valutata su un sottoinsieme del data set detto *batch*. Per fissare le idee, si consideri

il precedente esempio in cui il problema è stimare la funzione f che lega x a y. Si supponga inoltre di fornire in ingresso alla macchina, che ha appreso in qualche modo come associare a x un corrispondente $\tilde{y}$, $n \leq N$ valori di $x = x_i$ (con $i = 1, 2, \ldots, n$). Tali valori sono presi a caso dal data set insieme ai corrispondenti y_i. n costituisce pertanto la dimensione del *batch*. Una valida funzione di perdita L adatta a quantificare quanto le stime $\tilde{y}_i$, ciascuna relativa all'i-esimo x_i, siano corrette può essere data dalla radice quadrata dell'errore quadratico medio RMSE:

$$L = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2} \tag{1.1}$$

dove $y_i = f(x_i)$ è il valore corretto che deve essere stimato presente nel data set. Pertanto se $L = 0$ significa che la macchina è in grado di prevedere in modo corretto il valore assunto da $y = f(x)$.

In generale le tecniche di ML si basano proprio su quanto appena detto.

- Si crea un data set che contenga le informazioni necessarie a capire qual è l'output corretto in corrispondenza di un determinato input.
- Si realizza "l'apprendimento" tramite un algoritmo di ottimizzazione volto a minimizzare una funzione di perdita L a partire dai dati presenti nel data set.

Ovviamente sia la creazione del data set che la scelta della funzione di perdita devono essere effettuate tenendo conto del problema che si desidera risolvere.

Per quanto riguarda l'algoritmo di ottimizzazione su cui si basa "l'apprendimento", esso deve agire su qualcosa modificando il comportamento della macchina. Quindi si deve prima definire come quest'ultima prevede l'output e poi definire l'algoritmo di ottimizzazione. Tra le diverse strutture utilizzate nel ML utilizzate per predire l'output corrispondente a determinati input vi è la *rete neurale feed-forward*.

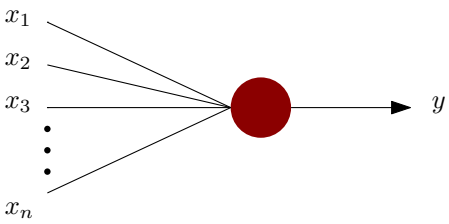
Prima di spiegare come è costruita una rete neurale feed-forward è necessario descrivere la struttura degli elementi di base che la compongono: i neuroni. Un neurone con n ingressi è costituito da una funzione di attivazione $act : \mathbb{R} \to \mathbb{R}$ non lineare, da n valori $w_i \in \mathbb{R}$ con $i = 1, 2, \ldots, n$ e da un valore di offset $b \in \mathbb{R}$. w_i e b sono generalmente indicati con il nome di pesi del neurone. In figura 1.1 è riportato il relativo schema a blocchi e la tipica rappresentazione. Come si può osservare, l'uscita y viene calcolata applicando la funzione di attivazione a una combinazione lineare degli n ingressi x_i e degli n pesi w_i sommata al peso di offset b; in formule

$$y = act\left(\sum_{i=1}^{i=n}(w_i\, x_i) + b\right)$$

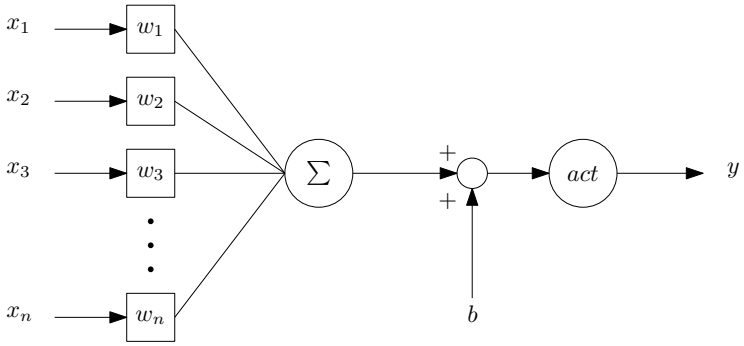
Tipiche funzioni di attivazione utilizzate sono la tangente iperbolica (tanh), la funzione sigmoide σ e la funzione *relu* definite di seguito.

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \qquad \sigma(x) = \frac{1}{1 + e^{-x}} \qquad relu(x) = max(0, x)$$

Una rete neurale in generale è costituita da un numero anche elevato di neuroni, il cui comportamento è dunque determinato dalla funzione di attivazione, dal valore dei pesi



(a)



(b)

Figura 1.1. (a) Tipica rappresentazione di un neurone. (b) Schema a blocchi di un neurone.

ad esso associati. Ovviamente il comportamento globale della rete dipende anche da come i neuroni sono tra di loro interconnessi. In figura 1.2 è riportato un semplice esempio di una rete formata da tre neuroni, da due ingressi e da tre uscite. Si può osservare come gli ingressi vengano elaborati attraverso tre neuroni le cui uscite coincidono con quelle della rete stessa. In particolare denotando con w_{ji} il peso relativo all'i-esimo ingresso del j-esimo neurone n_j e con b_j il corrispondente valore costante, le uscite y_j risultano pari a

$$y_j = act(w_{j1}x_1 + w_{j2}x_2 + b_j)$$

Per compattezza si è solito scrivere il tutto in forma matriciale supponendo che i neuroni possiedano tutti la medesima funzione di attivazione act.

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = act\left(\begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}\right) = act\left([W]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [\boldsymbol{b}]\right)$$

dove W è la matrice dei pesi w_{ji} e $\boldsymbol{b}$ è il vettore che contiene gli offset b_j dei neuroni. Inoltre nella formula la funzione $act(x)$ è intesa come applicata elemento per elemento se x è un vettore. Siccome gli ingressi vengono processati attraverso un solo stadio formato

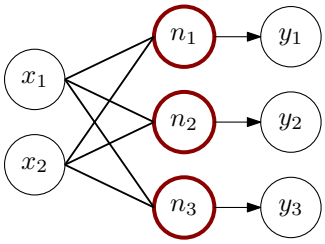


Figura 1.2. Esempio di rete neurale formata da 3 neuroni con 2 ingressi e 3 uscite.

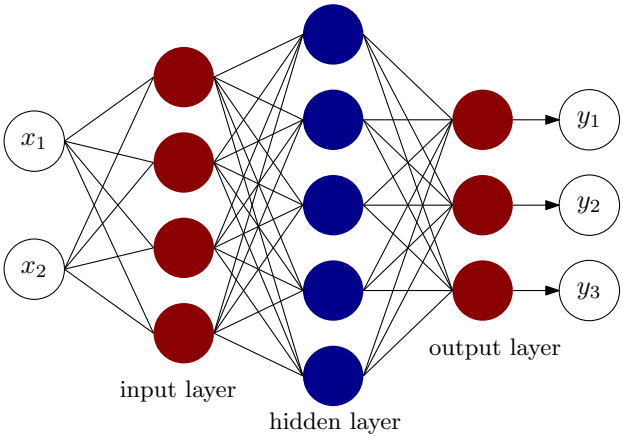


Figura 1.3. Rete neurale costituita da tre layer.

da tre neuroni, si dice che la rete è formata da un singolo layer. Naturalmente è possibile considerare configurazioni formate da più di un layer come la rete neurale in figura 1.3. In questo caso gli ingressi x_1 e x_2 vengono prima processati dal primo layer formato da 4 neuroni, poi dal layer più interno denominato *hidden layer* formato da 5 neuroni e, infine, dall'ultimo layer costituito da 3 neuroni le cui uscite coincidono con quelle della rete. In ciascun layer gli ingressi alla rete o le uscite del layer precedente vengono prima pesati attraverso la relativa matrice dei pesi, sommati al vettore degli offset $\boldsymbol{b}$ e successivamente al risultato è applicata la funzione di attivazione (si precisa che i neuroni dello stesso layer sono caratterizzati dalla medesima funzione di attivazione). Una rete neurale feed-forward è una rete come quella in figura 1.3 che ha una struttura sviluppata su uno o più layer.

Si è provato che la struttura multi-layer appena descritta delle reti feed-forward consente di rappresentare, per un opportuno numero di layer e di neuroni in ciascun layer e per un adeguato valore dei pesi e degli offset, una qualsiasi funzione che lega degli ingressi $x_1, x_2, \ldots, x_n$ a delle uscite $y_1, y_2, \ldots, y_m$. In particolare una rete feed-forward è in grado di rappresentare funzioni non lineari e questo è reso possibile dalla presenza di funzioni di attivazioni non lineari. Se infatti ogni singolo neurone fosse dotato di una funzione di attivazione lineare, allora l'intero comportamento ingresso-uscita della rete

neurale sarà di fatto lineare e conseguentemente essa potrà essere utilizzata solo per rappresentare funzioni lineari.

Oltre alle reti neurali feed-forward, esistono altre strutture più complesse come le reti neurali ricorsive (*recursive neural network*) e le reti neurli convoluzionali (*convolutional neural network*) giusto per citarne alcune. In questo lavoro di tesi verranno impiegate solamente reti neurali feed-forward.

Il sottoinsieme del machine learning che utilizza modelli formati da reti neurali è detto deep learning. Il nome deriva dal fatto che l'input viene elaborato attraverso un certo numero di layer posti in cascata l'uno dopo l'altro. Maggiore è il numero di layer, maggiore sarà la complessità della rete stessa e il numero di elaborazioni a cui l'input sarà sottoposto al fine di ottenere l'output. Il numero di layer viene detto appunto profondità della rete e costituisce, insieme al numero di neuroni, al tipo di funzioni di attivazione utilizzate e alla funzione di perdita, i cosiddetti *iper-parametri* della rete. Una volta che questi ultimi sono stati fissati, la struttura della rete neurale risulta definita. Restano però da determinare i pesi dei singoli neuroni che nel linguaggio del deep learning vengono denominati *training parameters*. Questi ultimi costituiscono di fatto i gradi di libertà sui quali l'algoritmo di ottimizzazione potrà agire al fine di minimizzare la funzione di perdita.

Nel prossimo paragrafo si illustreranno più nel dettaglio le modalità secondo le quali i pesi e gli offset della rete devono essere variati al fine ridurre progressivamente il valore di una generica funzione di perdita ottenendo una rete neurale addestrata.

1.2 *Discesa a gradiente*

In generale in qualsiasi problema di minimizzazione di una certa funzione f che dipende da un certo numero di variabili indipendenti, assume un significato rilevante il suo gradiente denotato con ∇f. A livello intuitivo esso indica localmente la "direzione" in cui la funzione stessa aumenta. Si pensi nel caso semplice di una funzione a una sola variabile $g(z)$. Se la derivata in un punto $z = z_o$ è positiva, allora significa che valutando g nel punto $z = z_o + \epsilon$ (con ϵ sufficientemente piccolo) si ha che $g(z_o + \epsilon) > g(z_0)$. Se invece la derivata in $z = z_o$ è negativa, allora si ha $g(z_o + \epsilon) < g(z_0)$. Dunque a seconda del segno della derivata (che è equivalente alla direzione del gradiente per funzioni di una variabile) la funzione $g(z)$ aumenta (derivata positiva) o diminuisce (derivata negativa) a seguito di un piccolo incremento ϵ. La figura 1.4 fornisce una rappresentazione grafica di quanto detto. Il ragionamento può essere naturalmente applicato anche alle funzioni di perdita di una generica rete neurale. A livello più formale, denotando con $\boldsymbol{w}$ il vettore che contiene tutti i pesi (w_{ji} e b_j) dei neuroni della rete, la funzione di perdita per $\boldsymbol{w}$ tendente a un certo $\boldsymbol{w_o}$ è approssimabile al primo ordine nel modo seguente

$$L(\boldsymbol{x}, \boldsymbol{w}) \simeq L(\boldsymbol{x}, \boldsymbol{w_o}) + \langle \nabla L(\boldsymbol{x}, \boldsymbol{w_o}) \ , \ \boldsymbol{w} - \boldsymbol{w_o} \rangle \tag{1.2}$$

dove $\nabla L(\boldsymbol{x}, \boldsymbol{w_o})$ è il gradiente della funzione di perdita valutato per il vettore degli ingressi alla rete $\boldsymbol{x}$ e per il vettore dei pesi $\boldsymbol{w_o}$. L'operazione $\langle \cdot \ , \ \cdot \rangle$ denota il prodotto scalare tra vettori. Se il gradiente non è nullo, si può osservare dalla (1.2) che il segno del secondo termine determina l'incremento o la diminuzione della funzione di perdita essendo quest'ultima sempre positiva. In particolare ponendo $\boldsymbol{w} = \boldsymbol{w_o} - \lambda \nabla L(\boldsymbol{x}, \boldsymbol{w_o})$, dove $\lambda \in \mathbb{R}$

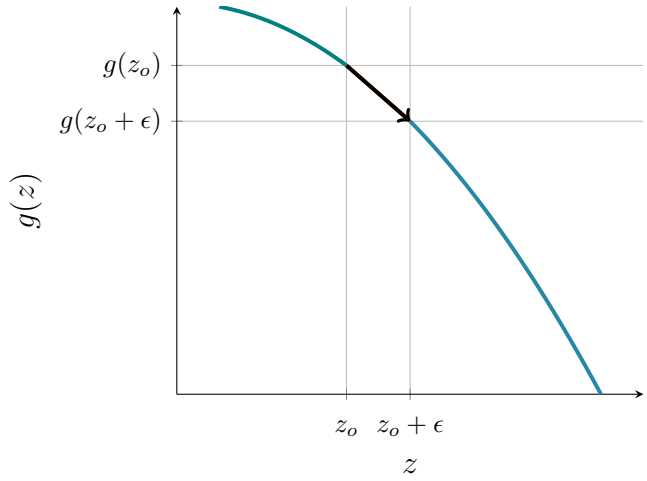


Figura 1.4. Interpretazione grafica della discesa a gradiente. La funzione $g(z)$ ha derivata negativa in z_o e pertanto, incrementando il valore della variabile indipendente da z_o a $z_o+\epsilon$, la funzione viene ridotta. La freccia in nero rappresenta "lo spostamento" che si ottiene a seguito dell'incremento di quantità ϵ.

è un parametro detto learning rate, si ottiene sostituendo nella (1.2)

$$L(\boldsymbol{x}, \boldsymbol{w}) \simeq= L(\boldsymbol{x}, \boldsymbol{w_o}) - \lambda \|\nabla L(\boldsymbol{x}, \boldsymbol{w_o})\|^2 \tag{1.3}$$

La formula traduce in linguaggio matematico il ragionamento descritto in precedenza: se la rete ha inizialmente un vettore dei pesi pari a w_o, allora è possibile diminuire la funzione di perdita aggiornando i pesi stessi nella "direzione" opposta del gradiente (il segno $-$ nella 1.3 ha appunto tale significato).

Il metodo di minimizzazione appena illustrata prende il nome di discesa a gradiente e costituisce la base degli algoritmi di ottimizzazione utilizzati nell'addestramento delle reti neurali. Si sottolinea inoltre che il learning rate λ è un *iper-parametro* delle rete neurale e deve essere scelto con attenzione. Se il valore di λ è troppo piccolo, allora l'algoritmo di minimizzazione sarà lento nel convergere. Nel caso peggiore l'algoritmo può convergere in un punto di minimo locale (diverso dal minimo globale) e di conseguenza, essendo nullo il gradiente, l'aggiornamento dei pesi non è più possibile. Graficamente si può pensare alla funzione $g(z)$ della figura 1.4 come se fosse "piatta" in un certo intervallo; in quel caso la derivata sarebbe nulla e non si avrebbe alcuna informazione sulla direzione in cui muoversi per minimizzare la funzione di perdita. Se invece il valore di λ è grande, allora l'algoritmo di minimizzazione convergerà più velocemente e sarà meno propenso a bloccarsi su punti di minimo locale; tuttavia valori alti del learning rate comportano incrementi molto grandi dei pesi e ciò può portare a un comportamento divergente.

1.3 *Algoritmo di Back-propagation*

É opportuno osservare che affinché la discesa a gradiente sia applicabile è necessario che il gradiente ∇L rispetto a un qualsiasi peso o offset esista. Innanzitutto si osserva che

nel calcolo dell'uscita di una rete neurale sono coinvolte esclusivamente operazioni di somma, operazioni di moltiplicazione e le funzioni di attivazione dei neuroni. Dunque il gradiente ∇L esiste se le funzioni di attivazione sono derivabili (essendo già addizioni e moltiplicazioni derivabili). In effetti quest'ultima affermazione risulta vera per le funzioni di attivazione introdotte in precedenza. Ad esempio per la funzione sigmoide e la tangente iperbolica si ha

$$\frac{d\ \tanh(x)}{d\,x} = 1 - \tanh(x)^2 \qquad \frac{d\,\sigma(x)}{d\,x} = \sigma(x)(1-\sigma(x))$$

Invece la funzione $relu(x)$ risulta non derivabile solamente in per $x = 0$. Nonostante ciò si pone nulla la derivata per definizione in $x = 0$ ottenendo

$$\frac{d\,relu(x)}{d\,x} = \begin{cases} 1 & \text{se } x > 0 \\ 0 & \text{altrimenti} \end{cases}$$

Si fa notare che porre nulla la derivata in $x = 0$ non crea alcun problema nella fase di addestramento. Infatti se durante l'addestramento stesso è richiesto di valutare la derivata della funzione *relu* in $x = 0$, allora essendo per definizione nulla si otterrà che $\nabla L = 0$ e quindi non si avrà alcun aggiornamento dei pesi.

Sapendo dunque che il gradiente ∇L risulta ben definito (almeno per le funzioni di attivazione fino ad ora introdotte) è necessario predisporre di un procedimento iterativo in grado di calcolare ∇L indipendentemente dalla struttura della rete. In questo modo sarà possibile implementare a livello di codice sorgente la discesa a gradiente secondo la formula (1.3). L'algoritmo di Back-propagation svolge proprio questa funzione. Si consideri

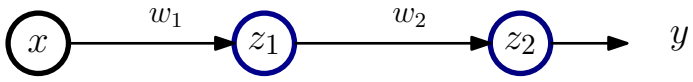


Figura 1.5. Esempio di applicazione dell'algoritmo di Back-propagation.

una semplice rete neurale feed-forward come quella riportata in figura 1.5 in cui i neuroni z_1 e z_2 sono caratterizzati dalla stessa funzione di attivazione *act*. Si supponga di voler calcolare il gradiente della funzione di perdita L rispetto al peso w_2 relativo al neurone z_2. Chiaramente L dipende dal valore dell'uscita della rete y che a sua volta dipende dal valore di w_2. Si può quindi applicare la regola della catena ottenendo

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial y}\,\frac{\partial y}{\partial w_2}$$

le due derivate a secondo membro sono immediatamente calcolabili una volta nota l'espressione della funzione di perdita e quella della funzione di attivazione *act*. Il procedimento è analogo se si vuole calcolare il gradiente rispetto al peso w_1 del neurone z_1

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial y}\,\frac{\partial y}{\partial z_{1\,out}}\,\frac{\partial z_{1\,out}}{\partial w_1}$$

con $z_{1\,out}$ uscita del neurone z_1. Come si può dedurre dall'esempio precedentemente illustrato, l'algoritmo di back-propagation calcola il gradiente partendo dall'uscita y e procedendo all'indietro (da cui il nome back-propagation) moltiplicando di volta in volta le varie derivate secondo la regola della catena.

1.4 *Fase di addestramento e scelta degli iper-parametri*

Avendo introdotto nei paragrafi precedenti la discesa a gradienti e l'algoritmo di back-propagation, è possibile ora descrivere in modo esaustivo i vari passaggi che compongono la fase di addestramento di una rete neurale. All'inizio i vari pesi di ogni singolo neurone della rete vengono inizializzati in modo casuale. Poi si prosegue iterando per un certo numero di volte la seguente procedura

1. Si prende a caso un *batch* del data set di dimensione fissata formato quindi da coppie $(\boldsymbol{x}\ \boldsymbol{y})$. $\boldsymbol{x}$ è il vettore degli ingressi, mentre $\boldsymbol{y}$ è il corrispondente vettore delle uscite attese.
2. Si valuta l'uscita della rete per gli ingressi $\boldsymbol{x}$ del *batch* ottenendo le corrispondenti previsioni dell'uscita $\tilde{\boldsymbol{y}}$.
3. Si calcola il valore della funzione di perdita e del gradiente per gli ingressi $\boldsymbol{x}$, le uscite $\boldsymbol{y}$ e le previsioni $\tilde{\boldsymbol{y}}$ relativi al *batch*. Il gradiente è calcolato mediante l'algoritmo di back-propagation.
4. Si aggiornano i pesi secondo la discesa a gradiente riducendo il valore della funzione di perdita sul *batch*.

Pertanto ad ogni iterazione la funzione di perdita valutata su *batch* diversi viene ridotta. La dimensione del batch e il numero di iterazioni, detto numero di epoche, sono anche essi iper-parametri della rete. L'intera procedura di addestramento appena illustrata prende il nome di *discesa a gradiente stocastica su mini-batch* (*mini-batch stochastic gradient descend*) dato che la discesa a gradiente è eseguita di volta in volta su *batch* del data set presi a caso. La dimensione del *batch* determina il tempo richiesto per l'addestramento della rete. Per batch molto piccoli si ha minor tempo per calcolare il gradiente e la funzione di perdita e di conseguenza l'addestramento della rete sarà più rapido. Viceversa se la dimensione del batch è molto grande, allora il carico computazionale richiesto nel calcolo del gradiente sarà più elevato. Ne segue un addestramento molto più lento della rete, ma in compenso la discesa a gradiente sarà più efficiente e accurata nel minimizzare la funzione di perdita dato che il gradiente stesso è calcolato su un vasto sottoinsieme del data set. Usare *batch* né troppo grandi né troppo piccoli assicura un buon compromesso tra efficienza e carico computazionale nell'addestramento della rete neurale. Osservare inoltre che la dimensione del data set incide notevolmente sull'addestramento dato che data set più grandi consentono di avere *batch* più grandi e, pertanto, miglior accurtezza da parte della rete nel predire l'uscita una volta addestrata.

Si fa notare che nell'esecuzione della discesa a gradiente stocastica il learning-rate λ deve essere inizialmente fissato e rimane costante per l'intera fase di addestramento. Esistono però delle varianti della discesa a gradiente stocastica note con il nome di ottimizzatori che modificano ad ogni epoca il valore del learning rate adattandolo in modo tale da evitare punti di minimo locale e minimizzare con un numero minore di

epoche la funzione di perdita. *RMSProp* (*Root Mean Square Propagation*) e *Adam* (*Adaptive Moment Estimation*) sono due degli ottimizzatori solitamente utilizzati.

Si vuole ora descrivere le modalità con cui è necessario scegliere gli iper-parametri (numero di neuroni, numero di layer, funzioni di attivazione, ecc …) al fine di ottenere una rete in grado di predire in modo corretto l'output. Per cominciare il layer di uscita della rete neurale deve essere costruito in base al problema da risolvere. Si supponga ad esempio di voler risolvere un semplice problema di classificazione in cui si deve assegnare a un dato numerico fornito in input un giudizio negativo o positivo. Allora il layer di uscita sarà costituito da un unico neurone avente come funzione di attivazione la funzione sigmoide. In questo modo la rete fornirà in uscita un valore di probabilità compreso tra 0 e 1 (valori che la funzione sigmoide può assumere). Dunque se in corrispondenza di un valore x in ingresso, la rete fornisce in uscita un valore di 0.8; allora significa che secondo la rete il valore x ha un giudizio positivo con probabilità del 80%. Si consideri ora un problema di regressione in cui a un valore x si vuole associare un altro valore $y = f(x)$ secondo una funzione f ignota. In questo caso l'ultimo layer della rete sarà costituito da un solo neurone avente come funzione di attivazione l'identità $act(x) = x$. Il motivo per cui si utilizza la funzione identità è quello di consentire alla rete di stimare funzioni f lineari; ciò sarebbe difficile se si impiegassero solo funzioni di attivazione non lineari. Inoltre funzioni di attivazione come la sigmoide limiterebbero l'output a un intervallo di valori indesiderato e non è detto che il valore y da predire rientri in tale intervallo (ad esempio tra 0 e 1 nel caso della sigmoide). Un'alternativa alla funzione identità è la $relu(x)$ che è lineare solo nel semipiano positivo ed è adatta se si desidera che la rete fornisca in output valori y positivi. Lo stesso ragionamento viene fatto nella scelta della funzione di perdita che, come detto nel primo paragrafo, avviene secondo il problema che si desidera risolvere mediante la rete neurale.

Inoltre è buona pratica normalizzare i valori in ingresso e in uscita a una rete neurale e il data set, ad esempio tra 0 e 1. Non è di fatto sicuro fornire in ingresso in fase di addestramento alla rete valori contenuti in un intervallo molto ampio dato che, data set costituiti da valori numerici molto diversi tra loro in termini di ordini di grandezza, possono comportare ampie variazioni dei pesi nella discesa a gradiente compromettendone la convergenza.

Per quanto riguarda il numero di neuroni e il numero di layer, questi vengono determinati a seconda di come si comporta la rete quando, una volta conclusa la fase di addestramento, essa viene sottoposta a calcolare le uscite in corrispondenza di ingressi che non appartengono al data set con cui la rete è stata addestrata. In particolare il data set viene suddiviso nelle seguenti tre parti.

- Data set di addestramento (*training data set*): è quella parte di data set che viene utilizzata esclusivamente per l'addestramento.
- Data set di validazione (*validation data set*): è la parte di data set che contiene dati diversi rispetto a quelli del data set di addestramento e che viene utilizzata per determinare gli iper-parametri della rete.
- Data set di test (*test data set*): è la parte di data set che viene impiegata per valutare le prestazioni della rete neurale una volta che gli iper-parametri sono stati fissati. Il data set di test è composto da dati differenti rispetto al data-set di addestramento e al data-set di validazione

Tipicamente la divisione del data set tra addestramento, validazione e test è 80%, 10%, 10% rispettivamente. Al fine di comprendere meglio il criterio con cui gli iper-parametri vengono scelti, è opportuno introdurre due importanti concetti: il fenomeno dell'*over-fitting* e quello dell'*under-fitting*. Il fenomeno dell'*under-fitting* si osserva quando il valore della funzione di perdita, alla fine della fase di addestramento, non risulta ben minimizzato. Ciò significa che la rete neurale non è in grado di prevedere correttamente l'uscita in corrispondenza di un ingresso appartenente al data set di addestramento. L'*over-fitting* si osserva invece quando la funzione di perdita assume valore piccoli se valutata sul data set di addestramento, mentre assume valori grandi se valutata sul data set di validazione. Tipicamente ciò accade quando il data set di addestramento contiene un numero non sufficientemente grande di dati e la rete neurale è formata da layer costituiti da un alto numero neuroni. Di fatto il termine *overfitting* deriva dal fatto che reti molto grandi in termini di numero di neuroni tendono ad *adattarsi* troppo al training data set e di conseguenza stimano con accuratezza minore l'uscita in corrispondenza di ingressi che non appartengono al training data set stesso.

Riassumendo, reti costituite da un basso numero di layer e di neuroni tendono ad andare in *underfitting*; viceversa reti costituite da un elevato numero di layer e neuroni rischiano l'*overfitting*. Lo stesso si verifica con il numero di epoche. Eseguire un gran numero di epoche nell'addestramento di una rete conduce all'*overfitting*, mentre eseguirne un numero ridotto comporta l'*underfitting*.

Sulla base di quanto detto si adottano i seguenti passaggi per la scelta degli iper-parametri per una generica rete.

- Si fissa la struttura della rete in termini di numero di layer e di neuroni, la funzione di perdita e le funzioni di attivazione dei neuroni. Per convenienza è consigliabile partire con reti abbastanza semplici (uno o due layer con una decina di neuroni).
- Si addestra la rete per un numero fissato di epoche e per una certa dimensione del *batch*.
- Si osserva l'andamento della funzione di perdita sul data set di validazione ad ogni epoca. In particolare si interrompe l'addestramento quando la funzione di perdita sui dati di validazione comincia ad aumentare (segno che la rete sta andando in *overfitting*). Se invece non viene osservato alcun aumento della funzione di perdita, si può provare ad aumentare il numero di epoche in modo da migliorare l'addestramento. Si ricorda che il numero di epoche non deve essere molto basso altrimenti si rischia l'*underfitting*.
- Al termine dell'addestramento si salva il valore della funzione di perdita valutata sul data di validazione.
- Si cambiano gli iper-parametri aumentando ad esempio il numero di neuroni, il numero di layer e provando funzioni di attivazione diverse. Si riesegue quindi l'addestramento ottenendo un nuovo valore della funzione di perdita sul data set di validazione.
- A questo punto si può costruire una tabella che riporta, per diverse scelte degli iper-parametri, diversi valori della funzione di perdita valutata sul data set di validazione. Si sceglie quindi la rete che ha il minor valore della funzione di perdita.
- Si valutano le prestazioni della rete scelta utilizzando il data set di test.

Si sottolinea che la scelta degli iper-parametri deve essere effettuata esclusivamente sul data set di validazione e usare quello di test solo per valutare le prestazioni della rete finale. Utilizzare infatti il data set di validazione per valutare il comportamento della rete non avrebbe senso dato che la rete stessa è stata manipolata (modificando opportunamente gli iper-parametri) in modo tale da aumentarne le prestazioni proprio sul data set di validazione stesso. Infine se alla fine di tale procedura le prestazioni sul data set di test della rete non sono soddisfacenti, si dovrà aumentare la dimensione del data set complessivo (che naturalmente andrà poi diviso nelle tre parti di validazione, test e addestramento) e rieseguire l'addestramento ottenendo dunque una rete molto più accurata nella predizione dell'uscita. Con quest'ultima affermazione si conclude l'introduzione ai concetti base che riguardano le reti neurali.

In questo capitolo si vogliono presentare i principali modelli impiegati per descrivere il comportamento di convertitori di potenza dc-dc.

Sistemi come i convertitori di potenza sono utilizzati per interfacciare tra loro altri sistemi come batterie e pannelli fotovoltaici, consentendone anche il collegamento con la rete elettrica. La conoscenza del comportamento di un convertitore di potenza è dunque molto importante al fine di realizzare tale interfacciamento e implementare sistemi di controllo per il medesimo convertitore. Esistono diversi modelli matematici per i convertitori dc-dc; alcuni sono molto semplici, ma poco accurati; altri invece garantiscono un livello molto alto di accuratezza in cambio però di una maggiore complessità. In questo

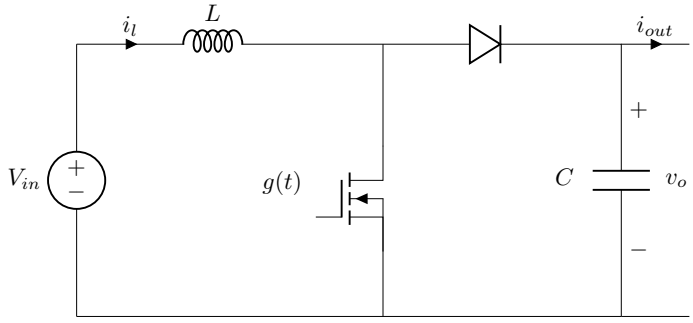


Figura 2.1. Circuito del convertitore boost ideale (privo di elementi parassiti).

capitolo si prenderà come riferimento il convertitore boost in catena aperta riportato in figura 2.1. Il circuito è composto da elementi ideali (privi di elementi parassiti). Gli ingressi del convertitore sono la tensione di ingresso V_{in}, la corrente di uscita i_{out} e il comando di gate $g(t)$ del mosfet, il quale sarà considerato uno switch ideale. Pertanto il segnale di gate $g(t)$ potrà assumere solo due valori: 1 nel caso in cui lo switch sia acceso e 0 nel caso in cui sia spento. Le uscite del convertitore sono la corrente in ingresso sull'induttore i_l e la tensione di uscita v_o. Normalmente il convertitore funzione applicando al gate un segnale PWM con un valore di duty cycle $\hat{\delta}$ scelto a seconda della tensione di uscita v_o che si vuole ottenere. Data la presenza di elementi non lineari come il diodo e lo switch, il comportamento del convertitore sarà di fatto non lineare.

2.1 *Modello medio di un convertitore*

Il modo più accurato per descrivere il comportamento di un convertitore è attraverso un sistema di equazioni differenziali non lineari. La struttura delle equazioni inoltre

dipende dalla modalità di funzionamento nella quale il convertitore si trova. Ad esempio il convertitore boost ammette due modalità di funzionamento.

- Continuous conduction mode (CCM): la corrente i_l è sempre positiva ed è quindi tale da mantenere acceso il diodo negli intervalli in cui lo switch è spento.
- Discontinuous conduction mode (DCM): la corrente sull'induttore si annulla negli intervalli in cui lo switch è spento comportando l'interdizione del diodo.

Nel caso in cui il convertitore boost sia in funzionamento CCM, le equazioni che ne descrivono il comportamento sono le seguenti

$$\begin{cases} L\,\dfrac{di_l(t)}{dt} = V_{in}(t) - (1 - g(t))\, v_o(t) \\ \\ C\,\dfrac{dv_o(t)}{dt} = (1 - g(t))\, i_l(t) - i_{out}(t) \end{cases} \tag{2.1}$$

come si può notare le equazioni sono non lineari data la presenza di operazioni di moltiplicazione. Le (2.1) consentono di rappresentare in maniera molto accurata il comportamento del convertitore includendo sia il ripple della corrente sull'induttore che quello sul condensatore. Tuttavia molte volte risulta comodo utilizzare dei modelli che coinvolgano direttamente il duty cycle δ impiegato nella modulazione PWM e non il segnale di comando $g(t)$, soprattutto se si desidera progettare un sistema di controllo per il convertitore. Tali modelli si ottengono a partire dalle equazioni del convertitore applicando una media mobile nel tempo, ricavando quindi delle equazioni che non legano più tra loro le grandezze istantanee ma quelle medie. In generale se $x(t)$ è il valore di una grandezza (tensione o corrente ad esempio) istantaneo, il valore medio è dato da

$$\overline{x} = \frac{1}{T_{sw}} \int_{t-T_{sw}}^{t} x(\tau)\, d\tau \tag{2.2}$$

dove T_{sw} denota il periodo di modulazione della PWM. Applicando ad esempio la (2.2) alla prima equazione in (2.1) si ottiene

$$L\frac{d\,\bar{i}_l(t)}{dt} = \overline{V}_{in}(t) - \frac{1}{T_{sw}} \int_{t-T_{sw}}^{t} (1 - g(\tau))\, v_o(\tau)\, d\tau$$

Supponendo che il ripple sulla tensione di uscita sia piccolo, è possibile considerare la tensione di uscita v_o pressoché costante e pari al valore medio $\overline{v}_o(t)$ nel periodo di modulazione ottenendo quindi

$$\frac{1}{T_{sw}} \int_{t-T_{sw}}^{t} (1 - g(\tau))\, v_o(\tau)\, d\tau = \overline{v}_o(t)\ \frac{1}{T_{sw}} \int_{t-T_{sw}}^{t} 1 - g(\tau)\, d\tau = \overline{v}_o(t)\,(1 - \delta(t))$$

dove l'ultima uguaglianza segue dalla definizione di duty cycle

$$\delta(t) = \frac{1}{T_{sw}} \int_{t-T_{sw}}^{t} g(\tau)\, d\tau$$

Eseguendo lo stesso ragionamento anche per la seconda equazione in (2.1) (nell'ipotesi che il ripple sulla corrente dell'induttore i_l sia trascurabile nel periodo di modulazione) si ottiene il modello medio del convertitore boost in CCM

$$\begin{cases} L\dfrac{d\,\bar{i}_l(t)}{dt} = \overline{V}_{in}(t) - \overline{v}_o(t)\,(1-\delta(t)) \\ C\dfrac{d\,\overline{v}_o(t)}{dt} = (1-\delta(t))\,\bar{i}_l(t) - \bar{i}_{out}(t) \end{cases} \tag{2.3}$$

Come accennato in precedenza, nel modello medio compare direttamente il duty cycle e non il segnale di comando $g(t)$ dello switch. Inoltre il modello medio di un qualsiasi convertitore consente di rappresentarne in buona approssimazione il comportamento fintanto che il ripple sulle variabili di stato (tensione sui condensatori e correnti sugli induttori) è piccolo. Si ricorda infine che il modello medio fornisce solo informazioni sull'andamento nel tempo delle grandezze medie e non di quelle istantanee. Ciò significa che se ad esempio nel convertitore boost si desidera sapere l'andamento del ripple sulla tensione di uscita v_o, si dovranno utilizzare le (2.1) e non il modello medio dato dalle (2.3).

Nei prossimi paragrafi si descriveranno alcuni metodi black-box presenti in letteratura per stimare il modello medio di un dato convertitore DC-DC.

2.2 *Modello lineare:* G-parameters model

In generale risulta complesso gestire modelli medi di un convertitore basati su equazioni differenziali non lineari. Pertanto di norma vengono utilizzati dei modelli linearizzati attorno a un determinato punto di lavoro noti in letteratura con il nome di *G-parameters models*. In questi modelli ciascuna grandezza $x(t)$ del sistema viene scomposta in due componenti: una componente continua X, la quale identifica il punto di lavoro, e una componente di perturbazione (o componente di piccolo segnale) $\hat{x}(t)$. La grandezza complessiva è data dalla somma delle due componenti $x(t) = X + \hat{x}(t)$. Nel caso specifico del convertitore boost alimentato da una tensione di ingresso V_{in} costante, è possibile operare una linearizzazione del modello medio nell'intorno di un certo valore di duty cycle D e di un certo valore di corrente di uscita I_{out}. Il punto di lavoro è pertanto dato dalla coppia di valori $(D,\ I_{out})$. Il modello lineare che si ottiene in termini di funzioni di trasferimento è rappresentato schematicamente in figura 2.2. Come si può notare ciascuna grandezza è stata scomposta nelle rispettive componenti (componente continua e componente di piccolo segnale)

$$\begin{aligned} \delta(s) &= D + \hat{\delta}(s) \\ i_{out}(s) &= I_{out} + \hat{i}_{out}(s) \\ v_o(s) &= V_o + \hat{v}_o(s) \\ i_l(s) &= I_l + \hat{i}_l(s) \end{aligned}$$

Si precisa che tutte le grandezze di tensione e corrente sopra riportate e in figura 2.2 sono intese come grandezze medie e non istantanee. Ad esempio con v_o si intende il valore medio della tensione di uscita del convertitore e non il valore istantaneo. Le funzioni

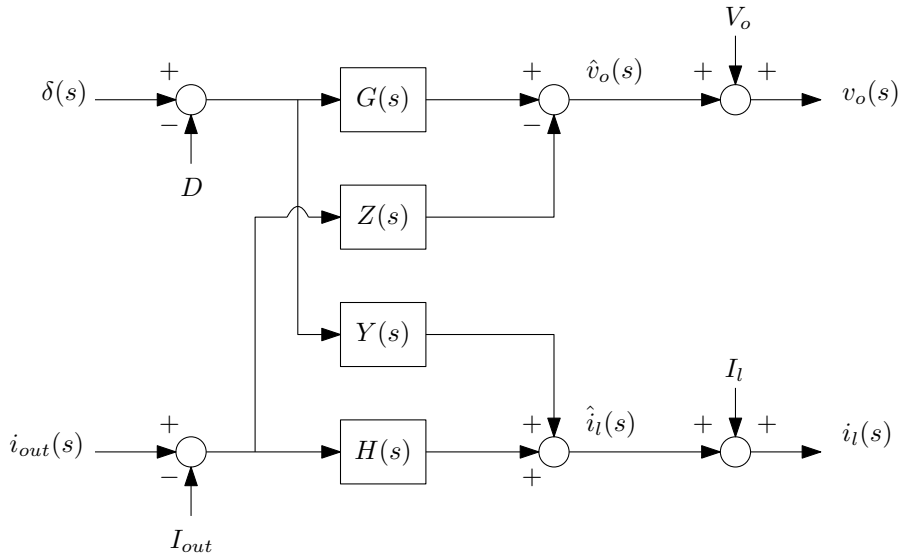


Figura 2.2. Modello *G-parametrico* del convertitore boost alimentato da una tensione di ingresso V_{in} costante. Il modello lega il duty cycle $\hat{\delta}$ e la corrente in uscita i_{out} alla tensione di uscita v_o e alla corrente in ingresso i_l..

di trasferimento $G(s)$, $Z(s)$, $Y(s)$ e $H(s)$ nel diagramma a blocchi prendono appunto il nome di *G-parameters* e ciascuna di esse assume un preciso significato fisico.

- $G(s)$ è la funzione di trasferimento tra la perturbazione sul duty cycle $\hat{\delta}$ e la perturbazione sulla tensione di uscita $\hat{v}_o$.
- $Z(s)$ rappresenta l'impedenza di uscita del convertitore.
- $Y(s)$ è la funzione di trasferimento tra la perturbazione sul duty cycle $\hat{\delta}$ e la perturbazione sulla corrente di uscita $\hat{i}_{out}$.
- $H(s)$ rappresenta il guadagno di corrente inverso e determina la perturbazione sulla corrente in ingresso al convertitore $\hat{i}_l$ a fronte di una perturbazione sulla corrente di uscita $\hat{i}_{out}$.

In definitiva le perturbazioni sulla tensione di uscita $\hat{v}_o$ e sulla corrente dell'induttore $\hat{i}_l$ possono essere determinate a partire dalle perturbazioni sul duty cycle $\hat{\delta}$ e sulla corrente di uscita $\hat{i}_{out}$ nel modo seguente

$$\begin{bmatrix} \hat{v}_o(s) \\ \hat{i}_l(s) \end{bmatrix} = \begin{bmatrix} G(s) & -Z(s) \\ Y(s) & H(s) \end{bmatrix} \begin{bmatrix} \hat{\delta}(s) \\ \hat{i}_{out}(s) \end{bmatrix} \tag{2.4}$$

Per quanto riguarda le componenti continue della tensione di uscita e della corrente sull'induttore V_o e I_l, queste ultime possono essere determinate a partire dal punto di lavoro (D, I_{out}).

É importante sottolineare che il modello *G-parametrico* è valido fintanto che le perturbazioni assumono valori contenuti. Se ad esempio nel convertitore boost il duty cycle

applicato o la corrente di uscita si discostano molto dai rispettivi valori in continua D e I_{out}, allora il modello *G-parametrico* ottenuto tramite la linearizzazione nel punto di lavoro $(D,\ I_{out})$ non sarà più corretto. In altre parole il modello *G-parametrico* fornisce solo il comportamento locale del convertitore in un determinato punto di lavoro. Se questo cambia, allora sarà necessario operare una linearizzazione nel nuovo punto di lavoro ottenendo delle funzioni di trasferimento diverse e pertanto un modello *G-parametrico* differente.

Quanto appena detto giustifica la seguente procedura black-box per determinare il modello *G-parametrico* di un generico convertitore.

- Si identificano le variabili di ingresso del sistema. Prendendo come riferimento il convertitore boost, le variabili di ingresso sono la tensione in ingresso V_{in}, il duty cycle δ e la corrente di uscita i_{out}.
- Si fissa un certo valore delle variabili di ingresso identificando in questo modo un determinato punto di lavoro sul quali eseguire la linearizzazione. Nel caso del boost precedentemente descritto per semplicità si è assunta costante la tensione di ingresso V_{in} e quindi, al fine di identificare il punto di lavoro, è sufficiente fissare un valore per il duty cycle D e un valore per la corrente di uscita I_{out}. Se si desidera studiare il comportamento del convertitore anche al variare della tensione di ingresso, sarà necessario considerare anche un determinato valore di V_{in} e pertanto il punto di lavoro sarà dato dalla terna $(D,\ I_{out},\ V_{in})$.
- Si determina il modello linearizzato nel punto di lavoro identificando le funzioni di trasferimento che legano le perturbazioni sugli ingressi alle perturbazioni sulle uscite.
- Si determina la componente continua di ciascuna uscita nel punto di lavoro misurandone il valore a regime.
- Per ciascuna uscita si sommano la relativa componente continua e la relativa perturbazione ottenendone in questo modo l'andamento nel tempo complessivo.

Chiaramente la parte più complessa è l'identificazione delle varie funzioni di trasferimento. In generale l'approccio solitamente utilizzato consiste nel sommare al punto di lavoro una perturbazione sinusoidale a una certa frequenza f in ingresso e misurare la corrispondente perturbazione sull'uscita. La funzione di trasferimento valutata alla frequenza f è dunque data dal rapporto tra la trasformata di Fourier della perturbazione sull'uscita e la trasformata di Fourier di quella iniettata in ingresso.

Una volta nota la funzione di trasferimento per diversi valori della frequenza perturbante f, è possibile eseguire un *best-fitting* al fine di determinarne numericamente i coefficienti del numeratore e quelli del denominatore. Ciò ovviamente è valido per sistemi formati da un solo ingresso e una sola uscita. Nel caso di sistemi costituiti da un certo numero di ingressi e uscite sarà necessario perturbare uno alla volta ciascun ingresso, sfruttando la sovrapposizione degli effetti, e misurare le corrispondenti perturbazioni sulle uscite.

Un altro aspetto importante nell'identificazione delle funzioni di trasferimento riguarda l'ampiezza della sinusoide perturbante. Se l'ampiezza è troppo piccola, vi è il rischio che anche l'ampiezza della perturbazione sull'uscita sia piccola e si confonda con il rumore presente nel sistema reale. Viceversa se l'ampiezza della sinusoide in ingresso è troppo grande, allora cambierà il punto di lavoro del sistema e non sarà possibile stimare

correttamente la funzione di trasferimento. Di conseguenza l'ampiezza della perturbazione in ingresso dev'essere scelta né troppo grande né troppo piccola al fine di evitare che il punto di lavoro del sistema cambi e che la perturbazione in uscita non si sovrapponga al rumore. Oltre a segnali perturbanti di tipo sinusoidale che eccitano il sistema a una sola frequenza, si possono utilizzare anche segnali in grado di eccitare il sistema su più frequenze come il rumore bianco, segnali multi-tone o segnali chirp (segnali sinusoidali che variano nel tempo la frequenza).

2.3 *Modello di Wiener-Hammerstein*

Il modello *G-parametrico* consente di approssimare il comportamento di un convertitore in un punto di lavoro attraverso un modello lineare. Tuttavia un generico convertitore presente delle non linearità specialmente in steady-state. Al fine di tenere conto di tali non linearità, in letteratura sono stati proposti dei modelli noti come *modelli di Wiener-Hammerstein*.

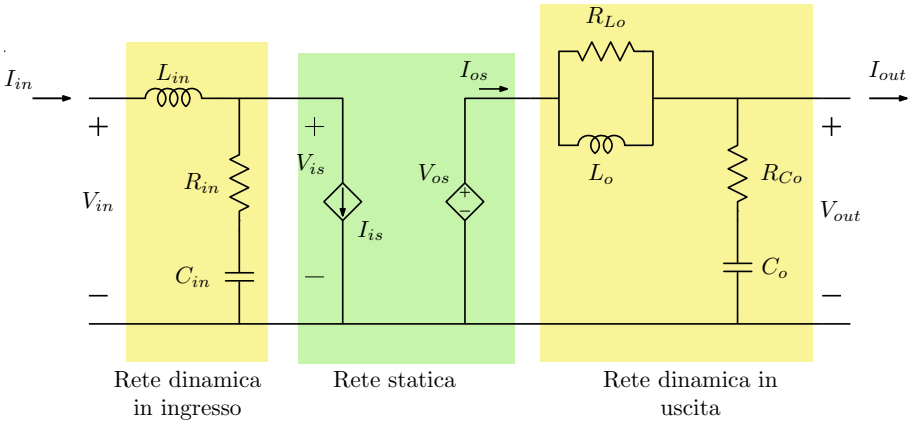


Figura 2.3. Esempio di modello di Wiener-Hammerstein per un convertitore dc-dc.

In figura 2.3 è riportato un esempio di modello di *Wiener-Hammerstein* ([7]) utilizzato per rappresentare il comportamento di un convertitore DC-DC in termini delle tensioni e correnti in ingresso e in uscita, supponendo fissato il duty cycle nella modulazione PWM. Si ricorda ancora una volta che tutte le grandezze di tensione e corrente riportate sono intese come grandezze medie valutate nel periodo di modulazione della PWM stessa.

Come si può notare il modello è composto da una rete statica utilizzata per descrivere il comportamento in steady-state del convertitore e da due reti dinamiche utilizzate per modellarne il comportamento dinamico.

La rete statica è costituita da un generatore variabile di corrente I_{is} e da un generatore di tensione V_{os}. I valori V_{os} e I_{is} variano in maniera non lineare a seconda della tensione e corrente in ingresso (V_{in} e I_{in}) e della tensione e corrente in uscita (V_{out} e I_{out}) le quali identificano il punto di lavoro del sistema.

Le due reti dinamiche sono invece composte da circuiti lineari costituiti da resistori, induttori e condensatori. La rete dinamica in ingresso descrive la dinamica tra la tensione

V_{in} e la corrente I_{in}, mentre quella in uscita modella la dinamica tra la tensione V_{out} e la corrente I_{out}.

La determinazione di un generico modello di *Wiener-Hammerstein* avviene identificando una alla volta le diverse reti che lo compongono. La rete statica viene determinata analizzando il comportamento a regime del convertitore variando il punto di lavoro. Per quanto riguarda le reti dinamiche, queste vengono determinate a partire dalla risposta al gradino del convertitore. In particolare per identificare la rete dinamica in ingresso si applicano dei gradini sulla tensione d'ingresso V_{in} e si misura la corrispondente corrente media in ingresso I_{in}. I valori delle resistenze, degli induttori e dei condensatori della rete vengono quindi determinati attraverso una minimizzazione ai minimi quadrati (*Least Square method*) sulla base dell'andamento della risposta a gradino ottenuta sulla corrente in ingresso I_{in}. L'identificazione della rete dinamica in uscita avviene in modo analogo applicando dei gradini sulla corrente di uscita I_{out} (tipicamente si esegue una variazione a gradino del carico in uscita) e acquisendo la tensione in uscita V_{out}.

Naturalmente per un dato convertitore è possibile ricavare diversi modelli di *Wiener-Hammerstein* variando il numero di condensatori, resistori e induttori che compongono le reti dinamiche al fine di ottenere approssimazioni migliori del comportamento dinamico del convertitore.

2.4 *Modello politopico*

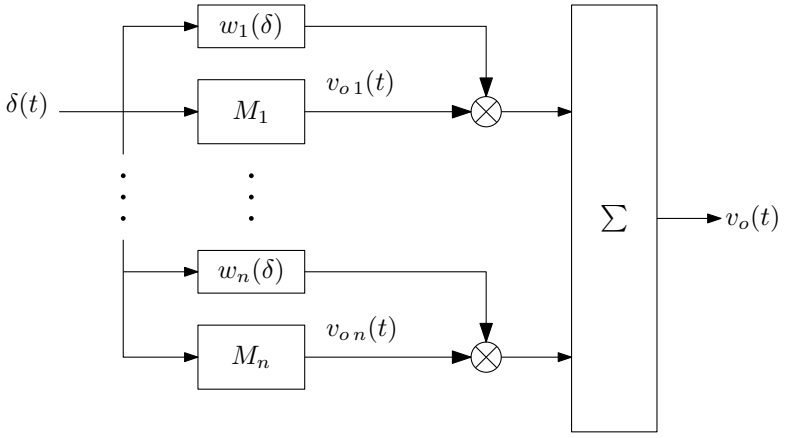


Figura 2.4. Modello politopico utilizzato per determinare l'andamento nel tempo della tensione di uscita $v_o(t)$ del convertitore in funzione del duty cycle.

Il modello di *Wiener-Hammerstein* permette di tener conto delle non linearità relative alla risposta in steady-state trascurando tuttavia possibili non linearità presenti nel comportamento dinamico. In generale la risposta di un convertitore di potenza DC-DC è non lineare sia dal punto di vista dinamico che dal punto di vista statico. I modelli politopici risolvono il problema considerando il comportamento dinamico non lineare che caratterizza il convertitore.

L'idea alla base dei modelli politopici consiste nel considerare un certo numero di modelli locali, ricavati in diversi punti di lavoro, e di combinarli attraverso una somma pesata delle relative uscite. La figura 2.4 riporta un esempio di modello politopico impiegato per descrivere il comportamento ingresso-uscita tra la tensione media di uscita v_o e il duty cycle δ applicato. Come si può notare dal diagramma, ciascuna uscita degli n modelli locali M_i $i = 1, 2, \ldots$ viene moltiplicata per la relativa funzione di peso w_i la quale dipende dal valore assunto dal duty cycle. L'uscita del modello politopico è dunque ottenuta sommando le varie uscite pesate; in simboli

$$v_o(t) = \sum_{i=1}^{n} w_i(\delta)\, v_{o\,i}(t)$$

La struttura di ciascun modello locale è riportata in figura 2.5. Come nel modello

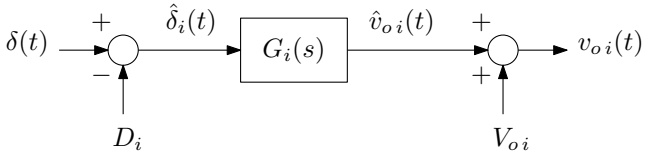


Figura 2.5. Struttura del modello locale i-esimo utilizzato nel modello politopico.

G-parametrico, l'uscita $v_{o\,i}(t)$ è calcolata sommando la componente statica V_{o_i} (calcolata a partire dal punto di lavoro dato dal valore di duty cycle D_i) alla componente di perturbazione $\hat{v}_{o\,i}(t)$. Quest'ultima è ottenuta dalla perturbazione sul duty cycle $\hat{\delta}_i(t)$ e dalla funzione di trasferimento $G_i(s)$ la quale costituisce il modello linearizzato nel punto di lavoro D_i. L'identificazione del modello locale in un punto di lavoro avviene nello stesso modo dei modelli *G-parametrici*: si fissa il punto di lavoro, si determina la funzione di trasferimento perturbando l'ingresso e misurando la perturbazione sull'uscita e infine si determina la componente statica dell'uscita stesse misurandone il valore assunto a regime.

Una volta ricavati i vari modelli locali, non resta che fissare le funzioni di peso w_i. Esse devono soddisfare le seguenti tre proprietà.

- Ciascuna funzione di peso può assumere solo valori tra 0 e 1.
- Il valore di ogni funzione di peso w_i è tanto più elevato quanto il punto di lavoro nel quale si trova il convertitore è *"vicino"* al punto di lavoro in corrispondenza del quale l'i-esimo modello locale è stato ottenuto. A livello più formale, se il punto di lavoro è individuato dalle variabili di ingresso al sistema $x_1, x_2, \ldots, x_n$, allora ciascuna funzione di peso soddisfa per un generico punto di lavoro $(X_1, X_2, \ldots, X_n)$

$$\begin{cases} w_i(x_1, x_2 \ldots, x_n) \simeq 1 & \text{se } (x_1, x_2, \ldots, x_n) \simeq (X_1, X_2, \ldots, X_n) \\ w_i(x_1, x_2 \ldots, x_n) \simeq 0 & \text{altrimenti} \end{cases} \tag{2.5}$$

  In questo modo l'uscita del modello locale ottenuto nel punto di lavoro $(X_1, X_2, \ldots, X_n)$ viene considerata nell'uscita complessiva del modello politopico solamente quando il convertitore si trova in tale punto di lavoro.

- La somma di tutte le funzioni di peso w_i impiegate nel modello politopico dev'essere pari a 1

$$\sum_{i=1}^{n} w_i(x_1,\, x_2, \ldots,\, x_n) = 1 \quad \text{per ogni } (x_1,\, x_2, \ldots,\, x_n) \tag{2.6}$$

Funzioni di peso solitamente utilizzate sono le funzioni a doppia sigmoide la cui espressione matematica, nel caso il punto di lavoro dipenda da una sola variabile di ingresso x, è di seguito riportata

$$w(x) = \sigma(m_l\,(\delta - c_l)) - \sigma(m_r\,(\delta - c_r)) = \frac{1}{1 + e^{-m_l\,(x - c_l)}} - \frac{1}{1 + e^{-m_r\,(x - c_r)}}$$

m_l e c_l denotano rispettivamente la pendenza e il centro dell'estremo in salita della doppia sigmoide, mentre m_r e c_r denotano invece la pendenza e il centro di quello in discesa.

Al fine di soddisfare le condizioni (2.5) e (2.6) è necessario scegliere opportunamente i valori delle pendenze m_l e m_r e dei centri m_r e c_r di ciascuna funzione di peso. A titolo di esempio si consideri il modello politopico in figura 2.4. Si ricorda che ognuno dei modelli locali M_i rappresenta il modello linearizzato del convertitore nel punto di lavoro D_i. Supponendo di aver scelto gli n punti di lavoro D_i uniformemente distribuiti nel intervallo $[D_1;\ D_n]$, si determinano i centri e le pendenze di ciascuna funzione di peso w_i come segue.

- Per gli indici $i = 2,\, 3,\, \ldots,\, n$ si considera il corrispondente punto di lavoro D_i e il precedente D_{i-1} e si prende il valore medio $c_i = (D_{i-1} + D_i)/2$.
- Si pongono pari a c_i il centro c_r della funzione di peso w_{i-1} e il centro c_l della funzione di peso w_i.
- Dato che i vari punti di lavoro sono uniformemente distribuiti nel intervallo $[D_1;\ D_n]$, si ha per ogni $i = 2,\, 3,\, \ldots,\, n$

$$D_i = D_{i-1} + \Delta \quad \text{con } \Delta > 0$$

Inoltre avendo posto c_i pari al valore medio nell'intervallo $[D_{i-1};\ D_i]$ si ha

$$D_i - c_i = c_{i+1} - D_i = \frac{\Delta}{2}$$

Si pone quindi la pendenza m_l e m_r di ciascuna funzione di peso w_i pari a

$$m = \frac{2\,K}{\Delta}$$

dove K è un valore costante. Così facendo per ogni $i = 2,\, 3,\, \ldots,\, n-1$ si avrà

$$
\begin{aligned}
w_i(D_i) &= \frac{1}{1+e^{-m\,(D_i-c_i)}} - \frac{1}{1+e^{-m\,(D_i-c_{i+1})}} \\
&= \frac{1}{1+e^{-2\,K\,(D_i-c_i)/\Delta}} - \frac{1}{1+e^{-2\,K\,(D_i-c_{i+1})/\Delta}} \\
&= \frac{1}{1+e^{-K}} - \frac{1}{1+e^{K}} \\
&= \sigma(K) - \sigma(-K)
\end{aligned}
$$

La funzione sigmoide $\sigma(x)$ è praticamente unitaria per $x \geq 5$ e pressoché nulla per $x \leq -5$. Pertanto scegliendo $K \geq 5$ risulta

$$w_i(D_i) = \sigma(K) - \sigma(-K) \simeq 1$$

Invece nel caso in cui la funzione di peso $w_i(\delta)$ $(i = 2,\ 3,\ \ldots,\ n-1)$ venga valutata per valori di δ molto diversi da D_i, più precisamente per $\delta << c_i$ o per $\delta >> c_{i+1}$, si ha

$$
\begin{cases}
\sigma(m\,(\delta - c_i)) \simeq \sigma(m\,(\delta - c_{i+1})) \simeq 0 & \text{per } \delta << c_i \\
\sigma(m\,(\delta - c_i)) \simeq \sigma(m\,(\delta - c_{i+1})) \simeq 1 & \text{per } \delta >> c_{i+1}
\end{cases}
$$

ne segue

$$
\begin{cases}
w_i(\delta) = \sigma(m\,(\delta - c_i)) - \sigma(m\,(\delta - c_{i+1})) \simeq 0 & \text{per } \delta << c_i \\
w_i(\delta) = \sigma(m\,(\delta - c_i)) - \sigma(m\,(\delta - c_{i+1})) \simeq 1 - 1 = 0 & \text{per } \delta >> c_{i+1}
\end{cases}
$$

Pertanto le funzioni di peso $w_2,\ w_3,\ \ldots,\ w_{n-1}$ soddisfano le condizioni in (2.5) per i punti di lavoro $D_1,\ D_2,\ \ldots,\ D_n$.

- Infine si determinano i centri c_l e c_r rispettivamente delle funzioni di peso w_1 e w_n ponendo

$$
\begin{aligned}
c_l &= D_1 - \frac{\Delta}{2} \\
c_r &= D_n + \frac{\Delta}{2}
\end{aligned}
$$

In questo modo le condizioni (2.5) saranno soddisfatte anche per w_1 e w_n.

La figura 2.6 riporta un possibile andamento qualitativo di alcune funzioni di peso w_i del modello politopico riportato in figura 2.4 ottenute secondo la procedura descritta. Come si può notare la forma della funzione di peso w_i è una sorta di *campana* centrata nel relativo punto di lavoro D_i e in corrispondenza del quale assume il valore massimo. Inoltre si può dimostrare che la somma punto per punto delle varie funzioni di peso nell'intervallo $[D_1;\ D_n]$ è pari a uno, ovvero anche la condizione 2.6 è soddisfatta per ogni $\delta \in [D_1;\ D_n]$. Intuitivamente questo accade perché appena una funzione di peso comincia a diminuire al variare del duty cycle, quella adiacente inizia ad aumentare come si può osservare graficamente in figura 2.6.

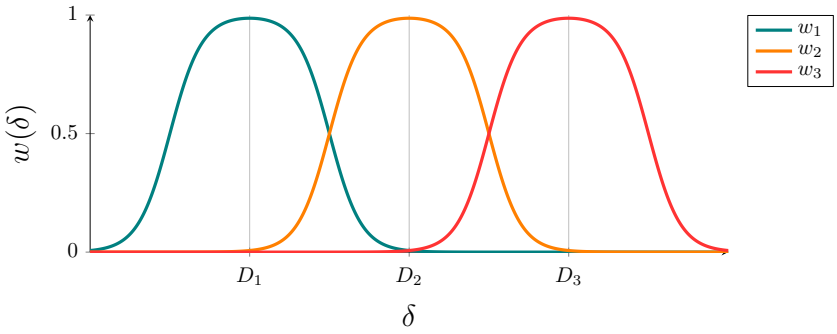


Figura 2.6. Funzioni di peso a doppia sigmoide del modello politopico.

Ad esempio se si suppone di aumentare gradualmente il duty cycle δ da D_1 a D_2, la funzione di peso w_1 comincia a diminuire pesando sempre di meno l'uscita del modello locale M_1. In contemporanea la funzione di peso w_2 inizia ad aumentare pesando sempre di più l'uscita del modello M_2. Il risultato è una transizione *"morbida"* dal punto di lavoro D_1 al punto di lavoro D_2. In definitiva, attraverso questo meccanismo basato sulle funzioni di peso, il modello politopico consente di tener conto delle non linearità sia nel comportamento dinamico che in quello statico di un convertitore, pesando maggiormente di volta in volta il modello locale ottenuto nel punto di lavoro in cui il convertitore stesso si trova attualmente.

In [3] si è provato come il modello politopico riesca a prevedere il comportamento di un convertitore boost controllato sulla tensione di uscita con una maggiore accuratezza in confronto al modello *G-parametrico* e al modello di *Wiener-Hammerstein*.

Tuttavia la complessità tende a crescere all'aumentare del numero di punti di lavoro. Questo perché se i punti di lavoro sono in numero elevato, allora anche il numero di modelli sarà elevato. Ciò si traduce in un maggior tempo impiegato per identificare tutti i modelli locali che dovranno essere combinati e in un maggior carico computazionale dato che per calcolare l'uscita del modello politopico si dovrà prima calcolare l'uscita di ogni singolo modello locale. Inoltre la cosa peggiora se si considerano modelli politopici costituiti da più variabili di ingresso. Infatti il numero di punti di lavoro aumenta in maniera esponenziale all'aumentare del numero di ingressi e in aggiunta le funzioni di peso dipendenti da più di un singolo ingresso avranno un'espressione diversa e più complicata rispetto alle w_i descritte in precedenza. In [1] è riportato lo sviluppo di un modello politopico a due ingressi per un convertitore DC-DC.

2.5 *Modello con reti neurali NARX*

L'ultimo metodo di identificazione black-box presente in letteratura che si vuole descrivere è quello basato su reti neurali NARX (*Nonlinear Autoregressive Model with Exogenous Input*). In tale metodo si assume un modello generale a tempo discreto del sistema ignoto dato da

$$\boldsymbol{y}(k) = f(\boldsymbol{u}(k), \boldsymbol{u}(k-1), \ldots, \boldsymbol{u}(k-m), \boldsymbol{y}(k-1), \ldots, \boldsymbol{y}(k-n)) \tag{2.7}$$

dove $\boldsymbol{y}(k)$ e $\boldsymbol{u}(k)$ sono rispettivamente il vettore delle uscite e il vettore degli ingressi del sistema valutati all'istante di campionamento k. La funzione f lega le uscite correnti $\boldsymbol{y}(k)$ agli ingressi correnti $\boldsymbol{u}(k)$ e ai valori precedentemente assunti sia dalle uscite che dagli ingressi stessi. m e n sono due numeri interi positivi e indicano il massimo ritardo da considerare sull'ingresso e sull'uscita rispettivamente. Identificare un sistema il cui comportamento è descritto dall'equazione (2.7) significa dunque determinare la funzione f.

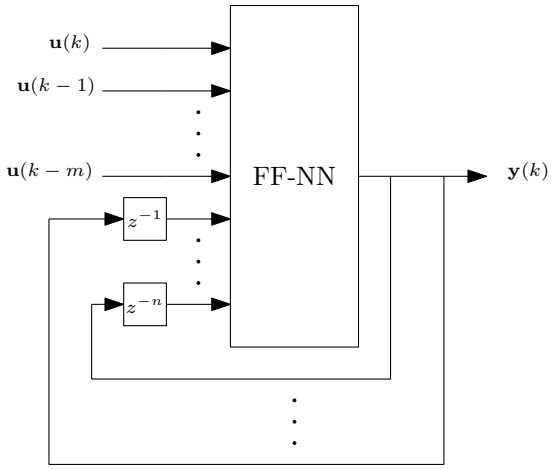


Figura 2.7. Struttura della rete neurale NARX.

La figura 2.7 riporta la struttura della rete neurale NARX. Come si può notare dallo schema, la rete NARX non è altro che una rete feed-forward (indicata in figura con FF-NN) le cui uscite sono retroazionate in modo da riprodurre la forma dell'equazione (2.7). La stessa rete feed-forward viene di fatto utilizzata per stimare la funzione f riproducendo in tal modo il comportamento complessivo, sia statico che dinamico, del sistema ignoto. Esempi in letteratura di implementazione di modelli black-box basati sulla NARX (per sistemi anche diversi dai convertitori DC-DC) si possono trovare in [6], [2] e [8].

La parte più delicata nello sviluppo di tali modelli è la generazione del data set necessario per l'addestramento della rete neurale. L'idea di base è quella di perturbare gli ingressi del sistema mediante dei segnali aventi le seguenti specifiche.

- I segnali in ingresso al sistema devono essere in grado di eccitare il sistema su un intervallo di frequenze molto ampio al fine di caratterizzare al meglio il comportamento dinamico del sistema stesso.
- L'ampiezza di ciascun segnale dev'essere abbastanza grande in modo da poter osservare il comportamento non lineare del sistema in diversi punti di lavoro.

In [8] ad esempio si sono perturbati il duty cycle e la corrente di uscita di un convertitore DC-DC impiegando dei segnali costituiti da variazioni a gradino di ampiezza variabile come riportato in figura 2.8. Scegliendo un'ampiezza dei gradini abbastanza elevata è possibile eccitare il sistema in diversi punti di lavoro; tuttavia, come si può notare

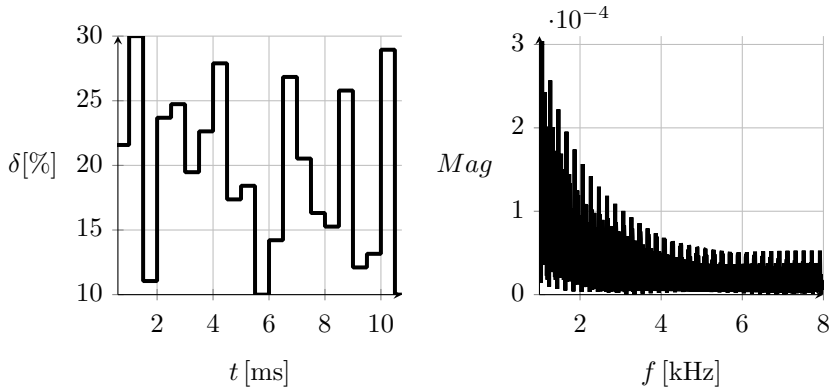


Figura 2.8. Perturbazione sul duty cycle δ mediante variazioni a gradino di ampiezza diversa nell'intervallo $[10\%,\ 30\%]$. Nel grafico di sinistra è riportato l'andamento nel tempo della perturbazione mentre nel grafico di destra è riportato il relativo contenuto in frequenza.

sempre dalla figura 2.8, lo spettro di un segnale perturbante siffatto tende a diminuire all'aumentare della frequenza. Pertanto se si vuole caratterizzare il comportamento del sistema a frequenze più elevate, sarà necessario sovrapporre alla perturbazione a gradino altri segnali con contenuto in alta frequenza come rumore bianco o segnali chirp.

In [6] sono stati invece utilizzati dei segnali composti da un'onda quadra ad ampiezza e frequenza variabili in maniera casuale nel tempo. In questo modo, variando nel tempo anche la frequenza oltre che l'ampiezza, è possibile eccitare il sistema su uno spettro più ampio.

Una volta scelto il segnale con il quale perturbare gli ingressi $\boldsymbol{u}$ del sistema, si procede con la creazione del data set osservando l'effetto che le perturbazioni producono sulle uscite $\boldsymbol{y}$. Nello specifico il data set sarà composto dai valori assunti dai segnali di perturbazione in ingresso $\boldsymbol{u}(k)$ e dai corrispondenti valori assunti dalle uscite $\boldsymbol{y}(k)$ nei diversi istanti di campionamento k.

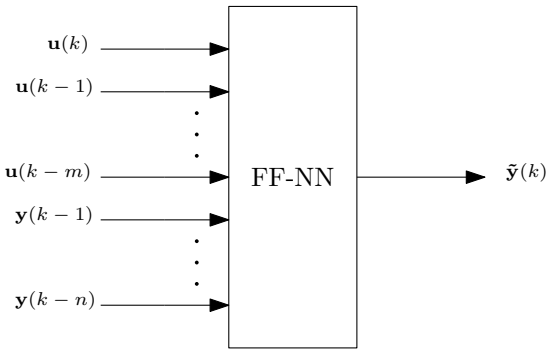


Figura 2.9. Configurazione a catena aperta della rete neurale NARX..

Creato il data set, si prosegue con la fase di addestramento nella quale si considererà

una configurazione a catena aperta della NARX eliminando l'anello di retroazione come mostrato in figura 2.9. Come si può notare, in tale configurazione l'addestramento della rete NARX è del tutto analogo all'addestramento di una rete feed-forward: si prelevano dal data set diversi valori degli ingressi $\boldsymbol{u}(k)$, $\boldsymbol{u}(k-1)$, $\boldsymbol{u}(k-m)$ e $\boldsymbol{y}(k-1)$, ..., $\boldsymbol{y}(k-n)$ e delle uscite precedenti $\boldsymbol{y}(k-1)$, ..., $\boldsymbol{y}(k-n)$, si calcola l'uscita corrispondente della rete $\tilde{\boldsymbol{y}}(k)$ e infine si aggiornano i pesi della rete stessa confrontando l'uscita predetta $\tilde{\boldsymbol{y}}(k)$ con l'uscita reale $\boldsymbol{y}(k)$ presente nel data set. Naturalmente prima di poter eseguire l'addestramento è necessario aver fissato i massimi ritardi m e n da considerare rispettivamente sugli ingressi $\boldsymbol{u}$ e sulle uscite $\boldsymbol{y}$. A tal proposito il metodo più semplice che solitamente viene utilizzato è quello di considerare m e n come iper-parametri della rete NARX. Di conseguenza si procede inizialmente addestrando la rete stessa per piccoli valori di m e n per poi effettuare successivamente ulteriori addestramenti aumentando di volta in volta i valori di m e n fintanto che non si raggiungono valori abbastanza bassi della funzione di perdita valutata sui dati di validazione.

Una volta completata la fase di addestramento e una volta che gli iper-parametri della rete NARX (compresi i valori dei ritardi m e n) sono stati fissati, si ripristina la retroazione sulle uscite ottenendo il modello del sistema descritto dall'equazione (2.7).

Oltre alla NARX, sono stati proposti in letteratura metodi basati su altre tipologie di reti neurali addestrate a partire dal comportamento nel tempo del sistema in risposta a una perturbazione degli ingressi. Ad esempio in [5] e in [4] si sono utilizzate delle reti neurali ricorsive nell'identificazione black-box di convertitori di potenza DC-DC. Ciononostante tali reti risultano essere meno preferibili rispetto alla rete NARX a causa della maggiore complessità nella struttura e del lungo tempo richiesto per il loro l'addestramento.

3

IDENTIFICAZIONE BLACK-BOX DI UN CONVERTITORE DC-DC BOOST

In questo capitolo si proporrà un metodo alternativo di identificazione black-box diverso da quelli visti in precedenza. Inoltre tale metodo verrà impiegato nella stima del modello medio di un convertitore boost in funzionamento CCM.

3.1 *Descrizione del modello black-box*

L'idea sulla quale si basa il metodo di identificazione che si intende sviluppare prende spunto dai modelli *G-parametrici* introdotti nel capitolo precedente. Per semplicità si consideri un generico sistema costituito da un solo ingresso $x(t)$ e una sola uscita $y(t)$. In un modello *G-parametrico* l'ingresso viene scomposto nella sua componente statica (o continua) X e nella sua perturbazione $\hat{x}$. L'uscita $y(t)$ di conseguenza viene stimata sommando alla componente statica Y (calcolata a partire da X) la componente di perturbazione $\hat{y}(t)$ a sua volta determinata applicando al sistema linearizzato in X, rappresentato dalla funzione di trasferimento $G(s)$, la perturbazione sull'ingresso $\hat{x}$. Il problema è che se si applica al modello un ingresso caratterizzato da una componente continua diversa da X, il punto di lavoro del sistema cambierà. Pertanto l'uscita predetta dal modello nel nuovo punto di lavoro non sarà più corretta dato che Y e $G(s)$ rappresentano il sistema linearizzato attorno al punto di lavoro precedente X.

Una possibile soluzione è quindi quella di adattare al variare dell'ingresso $x(t)$ sia la funzione di trasferimento $G(s)$ che la componente statica Y. In particolare, supponendo una forma generale della funzione di trasferimento

$$\frac{\hat{y}(s)}{\hat{x}(s)} = G(s) = \frac{b_m,\, b_{m-1},\, \ldots,\, b_0}{a_n,\, a_{n-1},\, \ldots,\, a_0}$$

i coefficienti a numeratore $b_m,\, b_{m-1},\, \ldots,\, b_0$ e quelli a denominatore $a_n,\, a_{n-1},\, \ldots,\, a_0$ dovranno dipendere da $x(t)$. In questo modo, se il punto di lavoro cambia, cambieranno di conseguenza anche l'espressione numerica di $G(s)$ e il valore statico Y: ne seguirà che l'uscita predetta dal modello sarà corretta. Sulla base di ciò si realizza lo schema riportato in figura 3.1. Come si può osservare la componente statica dell'uscita Y e la funzione di trasferimento $G(s)$ rappresentativa del sistema linearizzato vengono determinate a partire dall'ingresso $x(t)$ da due reti neurali feed-forward. Una rete neurale denominata nello schema *statica* stima un guadagno K che moltiplicato per l'ingresso $x(t)$ stesso produce la componente continua Y, mentre un'altra rete denominata *dinamica* fornisce una stima dei coefficienti di $G(s)$.

Ovviamente il comportamento in frequenza di un dato sistema reale è di tipo passa basso e quindi l'uscita di un qualsiasi modello black-box atto a descrivere tale sistema

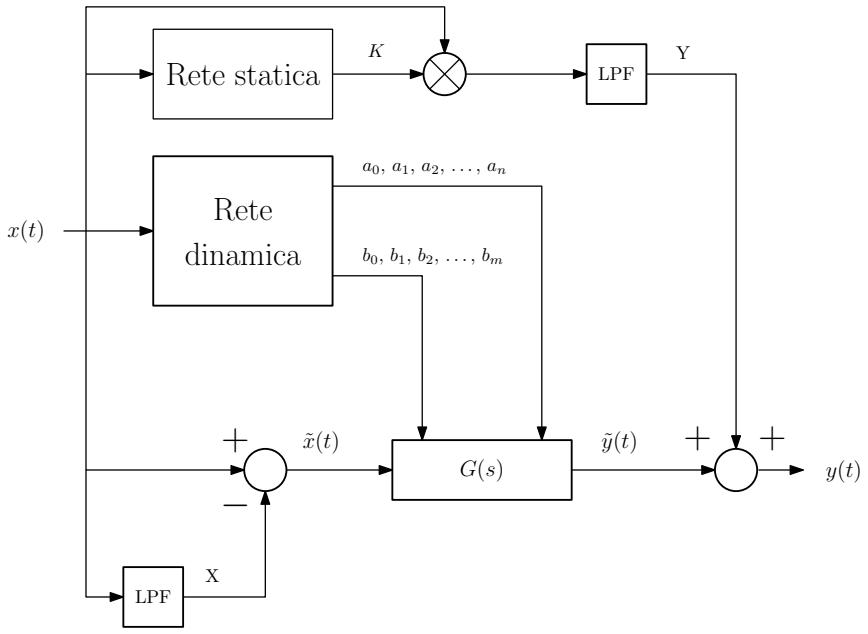


Figura 3.1. Schema del modello black-box basato sulla funzione di trasferimento $G(s)$ variabile con l'ingresso al sistema $x(t)$.

non potrà presentare variazioni a gradino. Pertanto, al fine di ottenere un andamento privo di variazioni troppo brusche sull'uscita $y(t)$, si predispone un filtro passa basso del primo ordine (LPF) a valle della rete statica come mostrato nello schema in figura 3.1. Infatti senza il filtro può accadere che, a fronte di una variazione a gradino dell'ingresso $x(t)$, anche la componente statica dell'uscita Y (e quindi anche l'uscita complessiva $y(t)$) vari a gradino. Di conseguenza si otterrebbe un modello in cui a una variazione a gradino dell'ingresso può corrispondere una variazione a gradino non voluta dell'uscita predetta $y(t)$.

Lo stesso filtro LPF è utilizzato nel calcolo della componente di perturbazione dell'ingresso $\hat{x}(t)$ che attraverso la funzione di trasferimento $G(s)$ permette di determinare la rispettiva perturbazione sull'uscita $\hat{y}(t)$. Nello specifico la perturbazione $\hat{x}(t)$ è ottenuta sottraendo all'ingresso $x(t)$ il punto di lavoro X a sua volta ottenuto filtrando l'ingresso $x(t)$ stesso. Equivalentemente $\hat{x}(t)$ è ottenuta filtrando l'ingresso $x(t)$ tramite il filtro passa alto dato da $1 - LPF(s)$, dove ovviamente $LPF(s)$ denota la funzione di trasferimento del filtro LPF.

Per quanto riguarda la frequenza di taglio di quest'ultimo, è importante che essa sia molto più bassa rispetto ai poli e agli zeri che la funzione di trasferimento $G(s)$ può avere al variare di $x(t)$. Detto in altri termini, per ogni valore dell'ingresso $x(t)$ il corrispondente polo o zero più lento di $G(s)$ deve trovarsi a frequenza molto più elevata rispetto alla frequenza di taglio del filtro LPF. In questo modo la componente statica Y varierà con una dinamica molto più lenta rispetto alla perturbazione $\hat{y}(t)$. Se la frequenza di taglio fosse a frequenze molto prossime a quelle dei poli e degli zeri di $G(s)$, allora si introdurrebbe

una dinamica dovuta al filtro non presente in quella del sistema reale descritta dalla stessa $G(s)$. Ne seguirebbe un andamento nel tempo dell'uscita $y(t)$ predetta dal modello diverso da quello dell'uscita reale.

3.2 *Creazione del data set per l'identificazione black box di un convertitore boost*

Nel paragrafo precedete si è proposto un modello black-box per l'identificazione di un generico sistema. Ora si desidera impiegare tale modello per identificare il comportamento di un convertitore boost in CCM. La figura 3.2 riporta uno schema del convertitore. Come

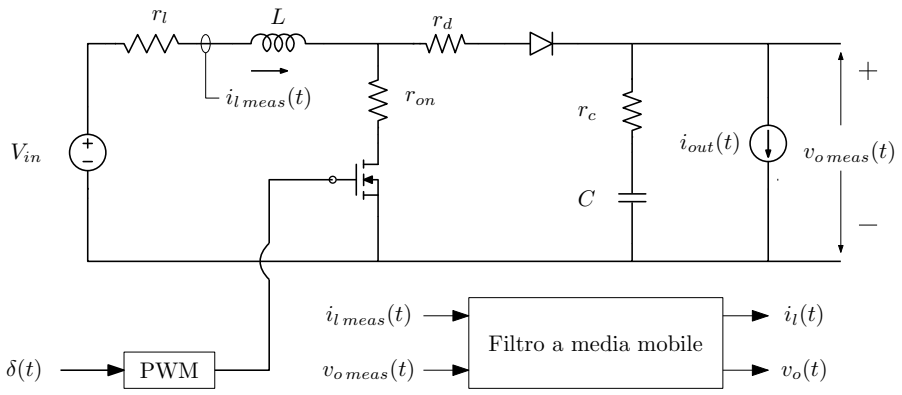


Figura 3.2. Modello del convertitore boost utilizzato in simulazione e da identificare secondo un approccio black-box. Il duty cycle δ e la corrente i_o costituiscono gli ingressi del sistema mentre le uscite sono la tensione media v_o di uscita del convertitore e la corrente media sull'induttore i_l.

si può osservare si è considerato un modello non ideale provvisto di elementi parassiti. In particolare si sono considerate le resistenze in serie all'induttore r_l, al condensatore r_c, al diodo r_d e al mosfet r_{on}. La tensione in ingresso V_{in} è fissata a un valore costante mentre il duty cycle δ e la corrente di uscita i_{out} vengono fatti variare. Nella figura si sono inoltre denotate rispettivamente con $v_{o\,meas}$ e con $i_{l\,meas}$ la tensione misurata in uscita dal convertitore e la corrente misurata sull'induttore, le quali vengono filtrate mediante un filtro a media mobile che fornisce in uscita i rispettivi valori medi v_o e i_l calcolati secondo la (2.2) con $T_{sw} = 1/f_{sw}$ (dove f_{sw} denota la frequenza di modulazione della PWM). La tabella 3.1 riporta i valori dei parametri del convertitore.

L'obiettivo che ora ci si pone è quello di determinare per diversi punti di lavoro in CCM del boost la funzione di trasferimento $G(s)$ che lega la perturbazione sul duty cycle $\hat{\delta}$ alla perturbazione sulla tensione media di uscita $\hat{v_o}$ in modo da poter creare il data set successivamente utilizzato per l'addestramento della rete neurale dinamica. Si procede dunque secondo i seguenti passi.

1. Si fissa un punto di lavoro del convertitore ponendo il duty cycle e la corrente di uscita a un valore costante D e I_{out} rispettivamente (si ricorda che la tensione di ingresso V_{in} è già fissata a un valore costante).

Valore capacità uscita	$C = 5.7\,\mu\text{F}$
Valore induttore	$L = 340\,\mu\text{H}$
Tensione di ingresso	$V_{in} = 150\,\text{V}$
Frequenza PWM	$f_{sw} = 20\,\text{kHz}$
Resistenza parassita induttore	$r_l = 0.5\,\Omega$
Resistenza parassita diodo	$r_d = 0.05\,\Omega$
Resistenza parassita capacità	$r_c = 0.1\,\Omega$
Resistenza r_{on} mosfet	$r_{on} = 0.05\,\Omega$

Tabella 3.1. Parametri del convertitore boost.

2. Come per i modelli *G-parametrici* introdotti nel capitolo precedente, si perturba il duty cycle sommando al valore in continua D un segnale sinusoidale $\hat{\delta}$ a frequenza f fissata.
3. Si esegue il rapporto tra la FFT (*Fast Fourier Transform*) della perturbazione osservata sulla tensione media di uscita v_o e la FFT della perturbazione iniettata in ingresso $\hat{\delta}$ al convertitore. IL rispultato è la funzione di trasferimento $G(s)$ valutata alla frequenza f.
4. Si ripete la procedura per diversi valori, in un certo intervallo, della frequenza f della perturbazione in ingresso in modo da valutare la funzione di trasferimento $G(s)$ per un certo numero di punti.
5. Si esegue un *fitting* di $G(s)$ a partire dai valori ottenuti a diverse frequenze ricavando quindi il valore dei coefficienti della funzione di trasferimento nel punto di lavoro $(D,\, I_{out})$.
6. Si riesegue l'intera procedura dal passo 1 per un diversi punti di lavoro in modo da creare un data set formato da un certo numero di elementi. Ogni elemento del data set comprenderà i valori dei coefficienti della funzione di trasferimento $G(s)$ e il corrispondente punto di lavoro $(D,\, I_{out})$.

Il procedimento appena descritto viene eseguito mediante simulazione del convertitore boost implementando lo schema in figura 3.2 in Matlab Simulink (il mosfet viene modellato con uno switch ideale). Nel dettaglio i punti di lavoro del convertitore sono stati scelti considerando 30 valori del duty cycle D equamente distribuiti nell'intervallo $[10\%;\ 30\%]$ e 30 valori della corrente di uscita I_{out}, anche essi equamente distribuiti, nell'intervallo $[2.5\text{A};\ 4.5\text{A}]$. In totale il numero di punti di lavoro risulta pari a 900.

Per quanto riguarda l'intervallo di frequenze in cui valutare la funzione di trasferimento $G(s)$, si è considerata una frequenza minima di 1Hz e una frequenza massima pari a $f_{max} = f_{sw}/2 = 10\,\text{kHz}$ dato che a frequenze più elevate il modulo della funzione di trasferimento diminuisce progressivamente (ovvero viene superata la banda passante del sistema). L'intervallo di frequenze considerato è dunque $[1Hz;\ 10\text{kHz}]$. In tale intervallo si sono considerati 42 valori della frequenza f del segnale perturbante in ingresso $\hat{\delta}$; più precisamente si sono considerati 24 valori di f (uniformemente distribuiti secondo

la scala logaritmica in base 10) nell'intervallo $[1Hz;\ 10\text{kHz}]$ e 18 valori nell'intervallo $[1kHz;\ 10\text{kHz}]$ in modo da poter individuare eventuali risonanze in alta frequenza.

Infine l'ampiezza della perturbazione sinusoidale sul duty cycle viene fissata a 0.1%. Per tale valore si è osservato che il comportamento del sistema risulta lineare: data una perturbazione sinusoidale in ingresso alla frequenza f, si ottiene una variazione sinusoidale alla medesima frequenza sull'uscita. Chiaramente, analizzando il comportamento del convertitore in simulazione, non è presente alcun rumore sovrapposto alle grandezze misurate e quindi l'ampiezza della perturbazione $\hat{\delta}$ può essere scelta anche molto più piccola rispetto a un'analisi reale del convertitore. Naturalmente non è consigliabile perturbare il sistema utilizzando ampiezze molto piccole del segnale perturbante dato che nella simulazione di qualsiasi sistema potrebbero verificarsi errori dovuti ad approssimazioni nei calcoli.

Un altro aspetto importante che si vuole evidenziare riguarda il calcolo della FFT. Dato un segnale periodico osservato per un tempo T e campionato a una frequenza di campionamento F_s vale quanto segue.

- La risoluzione in frequenza è data dal rapporto F_s/n dove n è il numero di campioni del segnale sui quali viene calcolata la FFT. Pertanto se si vuole valutare il contenuto spettrale su un intervallo abbastanza fitto in frequenza, si dovrà aumentare il numero di campioni n oppure si dovrà campionare più lentamente il segnale con una frequenza F_s più bassa.
- Se il tempo T non coincide con un multiplo intero del periodo del segnale, il risultato della FFT sarà caratterizzato da *leakage* e inoltre lo spettro ottenuto sarà diverso da quello del segnale stesso.
- Lo spettro del segnale si ripete per frequenze (in modulo) superiori alla frequenza di Nyquist $F_s/2$. Quest'ultima rappresenta dunque la massima frequenza alla quale lo spettro può essere valutato.

Occorre quindi prestare una certa attenzione nel calcolo della FFT.

Dato che la frequenza f sia della perturbazione in ingresso che di quella osservata sull'uscita è nota, è necessario prendere una finestra di osservazione di ampiezza nel tempo pari a $1/f$, ovvero uguale al periodo dei segnali di cui si vuole calcolare la FFT, in modo da non avere leakage. La frequenza di campionamento viene invece scelta ponendo $F_s = f\,n$ per un numero di campioni n inizialmente fissato; così facendo la risoluzione spettrale risulterà pari a f. Con $n = 2$ la frequenza di Nyquist risulta pari a

$$\frac{F_s}{2} = \frac{f\,n}{2} = f$$

In questo modo il risultato della FFT, sia della perturbazione di ingresso che di quella in uscita, comprenderà una riga spettrale alla frequenza di interesse f. Indicando quindi con $\hat{v_o}(f)$ e con $\hat{\delta}(f)$ rispettivamente la riga spettrale ottenuta dalla FFT della tensione di uscita e quella ottenuta dalla FFT del duty cycle alla frequenza f, la funzione di traferimento $G(j2\pi f)$ valutata a tale frequenza è data da

$$G(j2\pi f) = \frac{\hat{v_o}(f)}{\hat{\delta}(f)}$$

Ovviamente variare la frequenza di campionamento è possibile in simulazione, ma in un caso reale si dovrà agire solo sul numero di campioni n per valutare la FFT alla frequenza f dato che non sarà sempre possibile variare la frequenza di campionamento F_s.

Una volta acquisiti diversi valori in frequenza della funzione di trasferimento, viene quindi esegito il fitting per ricaverne i coefficienti. La figura 3.3 riporta il fitting della

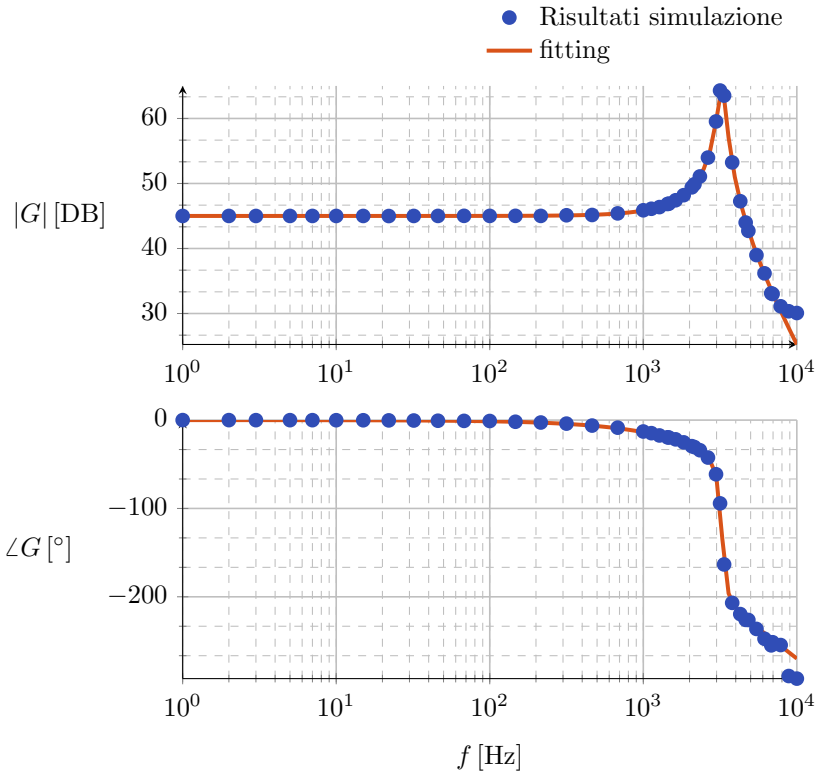


Figura 3.3. Fitting della funzione di trasferimento $G(s)$ ottenuto in corripondenza del punto di lavoro $D = 10\%$, $I_{out} = 2.8$ A.

funzione di trasferimento $G(s)$ ottenuto per un valore in continua del duty cycle e della corrente di uscita pari rispettivamente a $D = 10\%$ e a $I_{out} = 2.8$ A. Nella procedura di fitting si sono considerati diversi valori del grado del denominatore e del numeratore di $G(s)$. Per un grado pari a 3 per il denominatore e un grado pari a 2 per il numeratore si riesce ad ottenere un errore quadratico medio (RMSE) nel fitting adeguato. Nel caso specifico della figura 3.3 si sono ottenuti i seguenti errori

$$\text{errore RMSE modulo} = 0.9 \text{ dB}$$
$$\text{errore RMSE fase} = 5.7 \text{ deg}$$

La figura 3.4 riporta diversi fitting ottenuti in diversi punti di lavoro variando il valore in continua del duty cycle D e mantenendo costante quello della corrente di uscita I_{out}

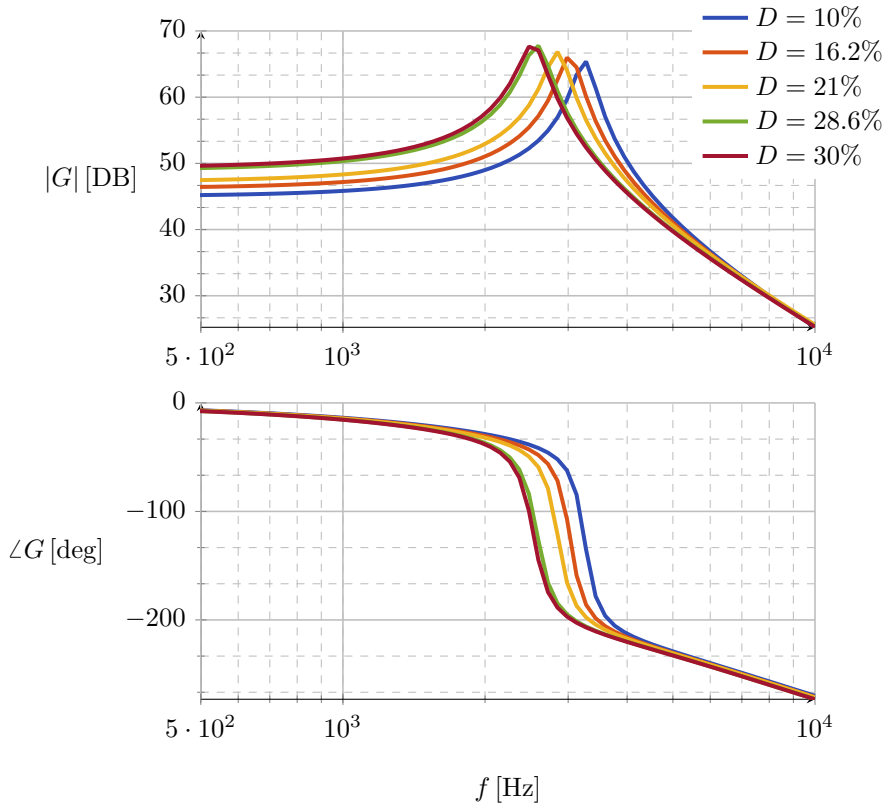


Figura 3.4. Diagramma di bode dei vari fitting ottenuti per diversi valori in continua del duty cycle D considerando una corrente di uscita I_{out} pari a 2.8 A.

pari a 2.8 A. Come si può notare dal diagramma del modulo la frequenza di risonanza diminuisce all'aumentare del duty cycle D. Inoltre il picco della risonanza aumenta leggermente all'aumentare di D, segno che per valori sempre più grandi di D la risposta del sistema sarà sempre meno smorzata. La tabella 3.2 riporta diversi della frequenza di

D [%]	freq. risonanza Hz
10	3275
16.2	3054
21	2848
28.6	2595
30	2535

Tabella 3.2. Valori della frequenza di risonanza al variare del valore in continua del duty cycle D.

risonanza stimata al variare di D.

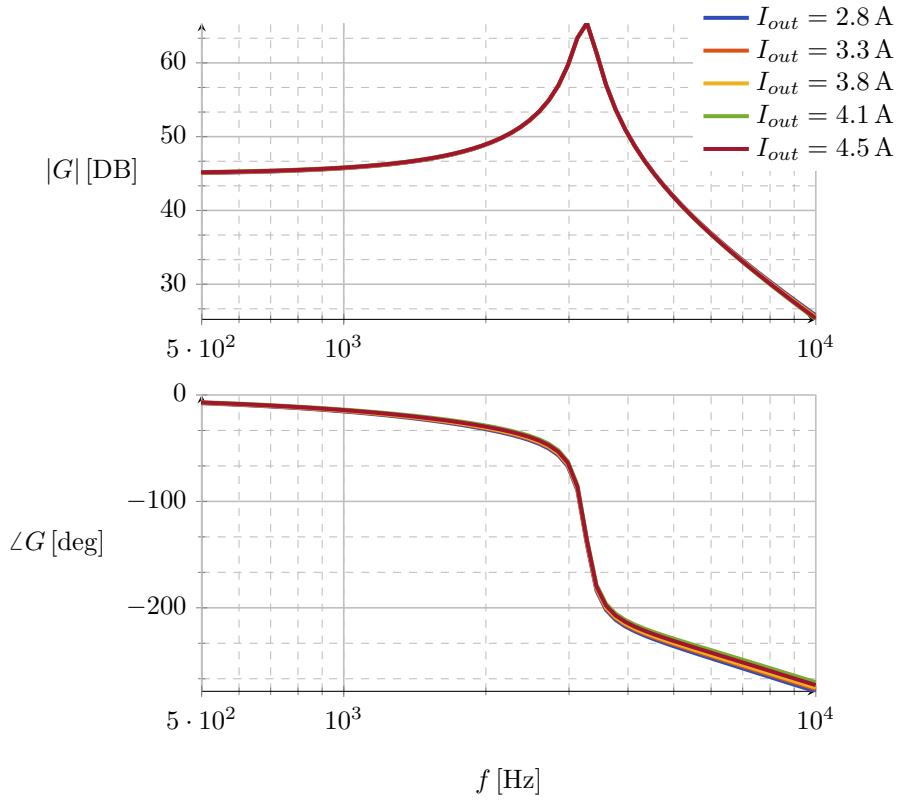


Figura 3.5. Diagramma di bode di alcuni fitting ottenuti per diversi valori in continua della corrente di I_{out}. Il duty cycle D è fissato a un valore pari a 10%.

La figura 3.5 invece riporta alcuni fitting ottenuti al variare della corrente I_{out} mantenendo il duty cycle D fisso al 10%. Come si evince dai grafici la funzione di trasferimento $G(s)$ non cambia al variare della corrente di uscita (si nota solo qualche variazione in alta frequenza). In particolare il valore della frequenza e del picco di risonanza (e quindi lo smorzamento) risultano indipendenti dalla corrente di uscita stessa; più nel dettaglio la frequenza di risonanza risulta pari a 3275 Hz (valore corrispondente a un duty cycle D del 10%).

I coefficienti della funzione di trasferimento ottenuti, mediante il fitting, nei diversi punti di lavoro del convertitore andranno a comporre il data set che sarà utilizzato nell'addestramento della rete neurale dinamica. Resta da illustrare la creazione del data set della rete neurale statica la quale, a partire dal punto di lavoro dato dalla coppia $(D;\ I_{out})$, fornirà in uscita il guadagno statico K. Quest'ultimo, moltiplicato per il valore del duty cycle $\delta(t)$ applicato in ingresso al convertitore, permetterà di ottenere la componente statica della tensione di uscita V_o.

In questo caso la procedura è molto più semplice della precedente: fissato il punto di lavoro si determina il guadagno K facendo il rapporto tra il valore della tensione di uscita V_o osservato a regime e il valore costante D del duty cycle applicato al convertitore (sostanzialmente il guadagno K è il rapporto tra la FFT della tensione di uscita v_o e la FFT del duty cycle δ valutate alla frequenza nulla).

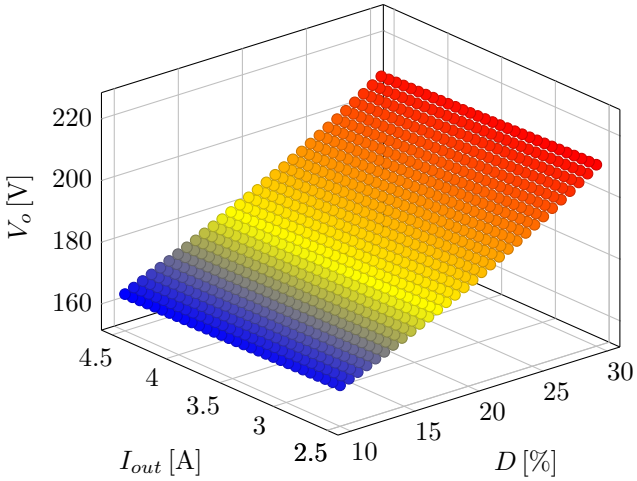


Figura 3.6. Tensione di uscita a regime V_o del convertitore boost ottenuta nei diversi punti di lavoro in termini del duty cycle D e della corrente di uscita I_{out}.

La figura 3.6 riporta in un grafico 3D i valori della componente continua della tensione di uscita $V_o = K \cdot D$ ottenuti nei punti di lavori considerati nella simulazione del convertitore boost. Dal grafico si evince che la tensione V_o aumenta all'aumentare del duty cycle D e ciò è coerente con il comportamento teorico del convertitore: la tensione di uscita V_o teorica è legata al duty cycle dall'equazione

$$V_o = \frac{V_{in}}{1 - D}$$

In realtà, anche se non si nota molto dal grafico, è comunque presente una leggera dipendenza della tensione V_o dalla corrente I_{out} e questo è dovuto alla presenza delle resistenze parassite considerate nel modello del convertitore (vedere la figura 3.2). I valori di K ottenuti nei diversi punti di lavoro andranno a comporre quindi il data set impiegato nell'addestramento della rete neurale statica.

3.3 *Risultati ottenuti nella fase di addestramento*

In questo paragrafo si vogliono presentare i vari risultati ottenuti nell'addestramento della rete neurale dinamica e della rete neurale statica.

La figura 3.7 riporta in forma tabellare la struttura del data set. Le prime due colonne costituiscono di fatto il punto di lavoro del convertitore in termini del duty cycle D e della corrente di uscita I_{out}, mentre le colonne denotate con b_1, b_0, a_2, a_1, a_0 riportano i

Punto di lavoro		Coeff. funzione di trasferimento				
D[%]	I_{out} [A]	b_1	b_0	...	a_0	K [V]
10	2.5	$-3.3 \cdot 10^{10}$	$4.3 \cdot 10^{15}$	...	$2.4 \cdot 10^{13}$	1658
10	2.57	$-4.6 \cdot 10^{10}$	$4.2 \cdot 10^{15}$	...	$2.4 \cdot 10^{13}$	1657
...	...	...		...		...

Figura 3.7. Struttura del data set riportato in forma di tabella. Ad ogni punto di lavoro sono associati rispettivamente i coefficienti della funzione di trasferimento $G(s)$ e il guadagno statico K.

valori dei coefficienti della funzione di trasferimento

$$G(s) = \frac{b_1\, s + b_0}{s^3 + a_2\, s^2 + a_1\, s + a_0}$$

ottenuti dal fitting nel punto di lavoro considerato. L'ultima colonna della tabella riporta i valori del guadagno statico K. Dato che la rete neurale dinamica dovrà stimare i coefficienti della funzione di trasferimento $G(s)$, essa sarà costituita da un layer di uscita composto da 5 neuroni. Invece il layer di uscita della rete neurale statica sarà composto da un solo neurone che fornirà in output il guadagno K.

Si procede dunque addestrando separatamente le due reti secondo la procedura descritta nel capitolo introduttivo sulle reti neurali. L'implementazione e l'addestramento delle reti è eseguito in ambiente Python mediante l'utilizzo della libreria *Tensor Flow*. Il data set, composto da un numero di punti di lavoro pari a 517, viene partizionato secondo una percentuale del 80% per l'addestramento, una percentuale pari al 10 per la validazione e una percentuale sempre pari al 10% per il test. Nella tabella 3.3 sono elencati alcuni

Ottimizzatore	ADAM
Funzione di attivazione	*relu*
Numero di epoche	1000
Dimensione del batch	25
Learning rate iniziale	$\lambda = 0.001$
Funzione di perdita	*mean squared error*

Tabella 3.3. Iper-parametri utilizzati nella fase di addestramento.

iper-parametri che sono stati fissati nell'addestramento di entrambe le reti. Si ricorda che l'ottimizzatore ADAM cambia il learning rate λ, inizialmente fissato, ad ogni epoca. Inoltre nella fase di addetramento si è operata la normalizzazione nell'intervallo $[0;\ 1]$ sia degli ingressi che delle uscite di ciascuna rete neurale sulla base dei valori massimi e minimi presenti nel data set. Ciò significa che se ad esempio si vogliono determinare i

coefficienti della funzione di trasferimento $G(s)$ nel punto di lavoro $(D;\ I_{out})$ si dovrà prima normalizzare il duty cycle D e la corrente I_{out}. I valori normalizzati di D e I_{out} saranno dunque forniti alla rete neurale dinamica che restituirà in uscita il valore normalizzato dei coefficienti b_1, b_0, a_2, a_1, a_0. Resta quindi da operare una normalizzazione di questi ultimi per ottenere l'espressione numerica di $G(s)$.

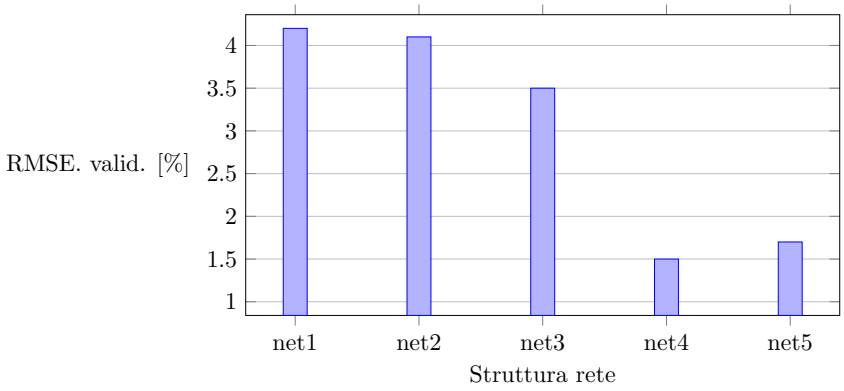


Figura 3.8. Errore di validazione per diverse configurazioni della rete neurale dinamica.

Per quanto riguarda la scelta del numero di neuroni e del numero di *hidden layer*, si sono effettuati diversi addestramenti valutando per le diverse configurazioni della rete addestrata l'errore RMSE sul data set di validazione.

Configurazione	Struttura	Numero di parametri	Numero di neuroni
*net*1	30 − 15 − 10 − 5	770	60
*net*2	50 − 30 − 5	1835	85
*net*3	50 − 40 − 20 − 5	3115	115
*net*4	60 − 50 − 30 − 5	4915	145
*net*5	80 − 60 − 50 − 5	8405	195

Tabella 3.4. Configurazioni considerate per la rete neurale dinamica.

In figura 3.8 è mostrato l'errore di validazione per le diverse strutture della rete neurale dinamica riportate in tabella 3.4. Nella stessa tabella è riportato inoltre il numero totale di pesi di cui è composta la rete per ciascuna delle configurazioni considerate. Ogni struttura della rete neurale utilizzata è indicata con una stringa del tipo

numero neuroni primo layer - numero neuroni secondo layer - ecc …

Ad esempio la struttura della configurazione *net*1 è costituita da quattro layer aventi rispettivamente 30, 15, 10 e 5 neuroni. Come detto in precedenza ogni configurazione della rete dinamica considerata prevede un layer di uscita composto da 5 neuroni, uno per ogni coefficiente della funzione di trasferiento $G(s)$.

Dall'istogramma in figura 3.8, si evince che la configurazione *net*1 costituita da un è quella che presenta l'errore RMSE di validazione in percentuale più alto pari al 4.2 %. Inoltre si osserva che all'aumentare del numero di neuroni fino alla configurazione *net*4 l'errore di validazione diminuisce. Al contrario quando si passa dalla configurazione *net*4 alla configurazione *net*5, l'errore di validazione aumenta leggermente. Il motivo è che la rete neurale nella configurazione *net*5 inizia a presentare segni di *overfitting*, ovvero il numero di neuroni è talmente elevato che la rete si adatta troppo al data set di training commettendo quindi errori sulla stima dei coefficienti in corrispondenza di punti di lavoro mai considerati nella fase di addestramento.

Alla luce dei risultati ottenuti, gli iper-parametri della rete neurale dinamica vengono scelti pari a quelli della configurazione *net*4 dato che tale configurazione consente di avere un errore RMSE di validazione molto basso pari al 1.5 %.

La scelta degli iper-parametri della rete statica avviene in modo del tutto analogo valutando l'errore di validazione conseguito sulla stima del guadagno K. Più precisamente si è scelta la struttura: $30 - 15 - 1$ con un errore di validazione RMSE pari al 0.22 %. Si ricorda che il layer di uscita della rete neurale statica è costituito da un solo neurone il quale fornisce in uscita il valore predetto del guadagno K.

A questo punto, una volta completato l'addestramento delle reti, si procede a valutarne le prestazione attraverso il data set di test.

La figura 3.9 riporta i grafici che mettono a confronto i coefficienti predetti dalla rete neurale dinamica e il guadagno predetto dalla rete neurale statica con i rispettivi valori reali prelevati dal data set di test. Tutti i valori sono stati normalizzati nell'intervallo $[0; 1]$.

Valore da stimare	Errore RMSE [%]
b_1	1.56
b_0	1.36
a_2	1.63
a_1	0.7
a_0	0.6
K	0.37

Tabella 3.5. Errore RMSE percentuale commesso dalle reti neurali nella predizione delle uscite (coefficienti della funzione di trasferimento e guadagno K).

La chiave di lettura secondo la quale interpretare ciascun grafico è la seguente. Ogni singolo punto in blu ha come coordinate nel grafico la coppia *(valore reale; valore predetto)*. Pertanto se i punti si dispongono lungo la bisettrice di equazione $y = x$ (rappresentata nei singoli grafici da una linea tratteggiata) allora significa che la rete neurale sarà in grado di stimare con una buona accuratezza il valore del coefficiente della funzione di trasferimento $G(s)$ o del guadagno K. Viceversa se i punti risultano molto dispersi, allora il valore predetto dalla rete sarà poco accurato.

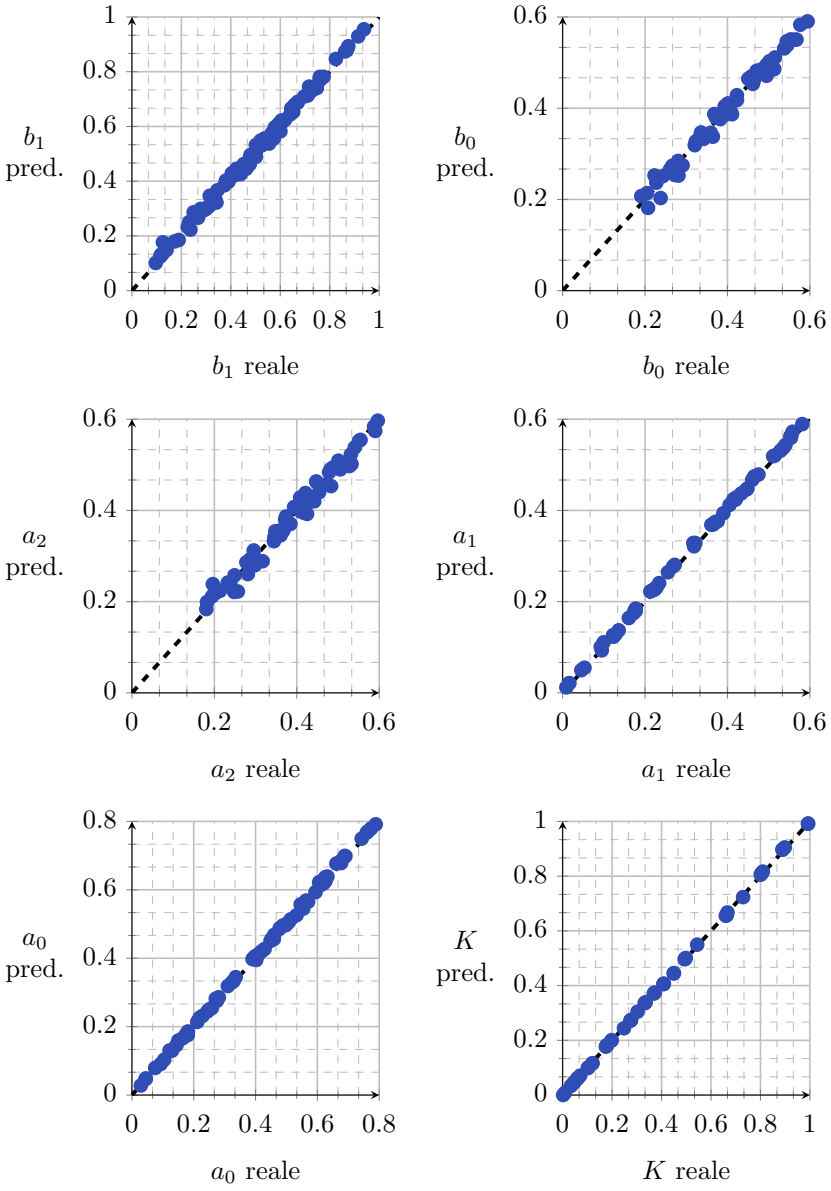


Figura 3.9. Prestazioni delle reti neurali addestrate. I grafici riportano sull'asse delle ordinate i valori predetti dalle reti neurali e sull'asse delle ascisse i corrispondenti valori reali presenti nel data set di test.

Osservando ad esempio i grafici relativi alla stima dei coefficienti, si può notare che qualche punto non risulta effettivamente allineato rispetto agli altri, segno che la rete neurale dinamica commetterà comunque un certo errore nel predire qualche valore dei

coefficienti stessi. Osservando invece il grafico relativo al guadagno statico K (l'ultimo in basso a destra) ci si può accorgere che i punti risultano ben allineati. Ne segue che la rete neurale statica è in grado di stimare con una buona accuratezza il guadagno K stesso.

Nella tabella 3.5 sono elencati gli errori RMSE percentuali commessi dalle reti. Nello specifico si è ottenuto un errore non superiore al 2.5 % per quanto riguarda la stima dei coefficienti, mentre per il guadagno statico K l'errore commesso risulta pari 0.37 %.

4

ANALISI RISULTATI OTTENUTI IN SIMULAZIONE

In questo ultimo capitolo della tesi si utilizzeranno alcuni modelli descritti nei capitoli precedenti per per predire, al variare del duty cycle δ, l'andamento del tempo della tensione media di uscita v_o del convertitore boost simulato (schema in figura 3.2). In particolare vengono implementati in ambiente Matlab Simulink i seguenti modelli black-box.

- Modello basato su reti neurali feed-forward descritto al capitolo precedente (vedere figura 3.1). L'ingresso del modello $x(t)$ è dato dal duty cycle δ, mentre l'uscita $y(t)$ costituisce la tensione di uscita $v_o(t)$ del convertitore. La frequenza del filtro passa basso LPF è fissata a 100 Hz. D'ora in avanti si denoterà tale modello con "Modello NN".
- Modello lineare (*G-parameters model*) ottenuto nel punto di lavoro $D = 10\%$, $I_{out} = 2.9$ A.
- Modello politopico (schema in figura 2.4) composto da 11 modelli locali ottenuti nei punti di lavoro corrispondenti alla corrente $I_{out} = 2.9$ A e ai valori D di duty cycle [10%; 10.7%; 11.4%; 12.1%; 12.8%, 13.5%; 14.1%; 14.8%; 15.5%; 16.2%; 17%].

4.1 *Predizione della risposta al gradino di duty cycle*

Nelle prime simulazioni del convertitore si è fissata la corrente di uscita I_{out} pari a 2.9 A e si è variato il duty cycle δ a gradino dal 10% al 17% in modo da cambiare il punto di lavoro del sistema.

La figura 4.1 riporta l'intero transitorio osservato dal momento dell'applicazione del gradino di duty cycle. Si ricorda che il modello lineare calcola la tensione di uscita sulla base della funzione di trasferimento $G(s)$ linearizzata nel punto di lavoro $D = 10\%$, $I_{out} = 2.9$ A. Al contrario, il modello black-box proposto utilizza la rete neurale dinamica per modificare i coefficienti della funzione di trasferimento $G(s)$ stessa al variare del punto di lavoro in cui si trova il convertitore. La stessa cosa avviene per il guadagno statico: il modello lineare somma alla perturbazione $\hat{v}_o$ un valore costante pari alla tensione continua V_o che si ottiene nel punto di lavoro considerato; invece il modello NN somma alla perturbazione $\hat{v}_o$ una tensione V_o variabile con il punto di lavoro. Il risultato, come si può osservare dalla figura 4.1, è che nel caso del modello lineare si ha un errore a regime elevato proprio perché il valore in continua sommato alla perturbazione è costante, mentre nel caso del modello NN quest'ultimo viene variato ottenendo di conseguenza un errore a regime molto più piccolo.

Nel dettaglio la tensione di uscita predetta dal modello lineare si assesta a regime attorno a un valore pari a 177 V, valore leggermente diverso da quello ottenuto nella

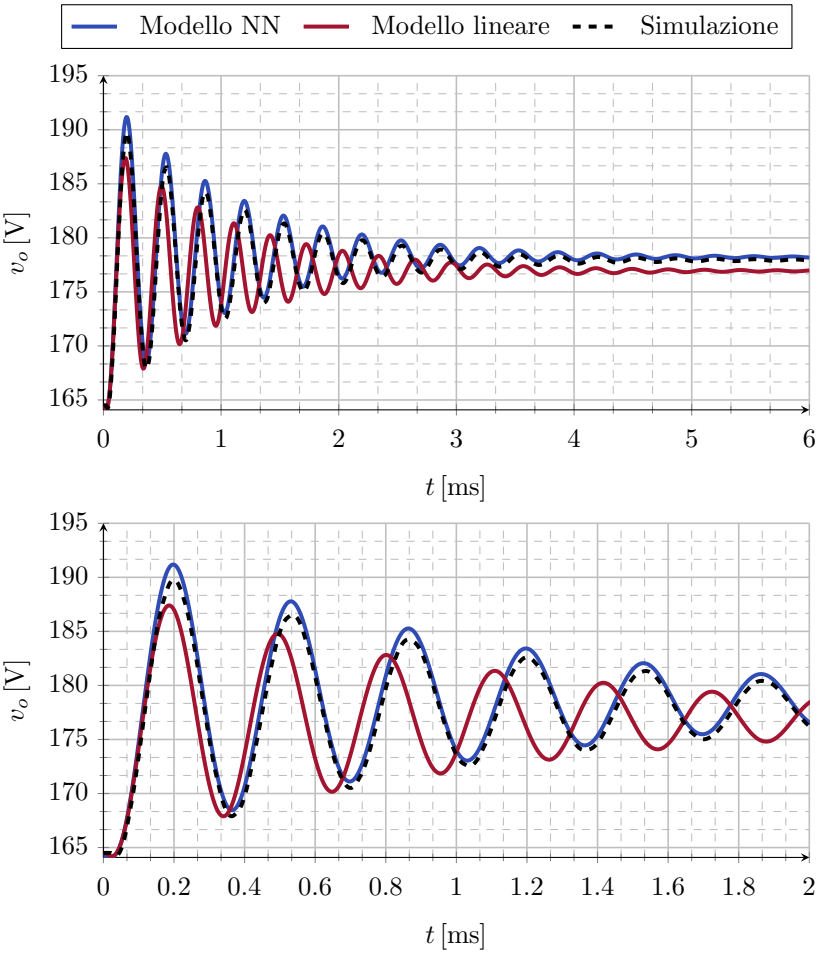


Figura 4.1. Predizione della tensione di uscita v_o del convertitore a una variazione a gradino del 7% del duty cycle al tempo $t = 0$ ms. Il valore della tensione predetto dal modello NN è confrontato con la tensione *reale* ottenuta dal convertitore simulato e con la tensione stimata dal modello lineare. Il primo grafico riporta il transitorio complessivo, mentre il secondo ne riporta un *ingrandimento* nell'intervallo [0 ms; 2 ms]. La corrente di uscita è mantenuta fissata al valore $I_{out} = 2.9$ A.

simulazione del convertitore pari a 178 V. Ne segue un errore di 1 V. Chiaramente è un errore pressoché trascurabile in relazione al range di tensione considerato [160 V; 200 V].

Tuttavia non è trascurabile l'errore associato alla stima della frequenza di risonanza. Sempre nella figura 4.1 sono riportati i dettagli del transitorio nel intervallo temporale [20 ms; 22 ms]. Si può osservare che la frequenza dell'oscillazione della tensione predetta dal modello lineare è ben diversa da quella che caratterizza la tensione *reale* ottenuta simulando il convertitore. Quello che succede è che, una volta applicato il gradino di duty cycle, il sistema *si porta* in un nuovo punto di lavoro. Ma ovviamente il modello lineare

stima la tensione di uscita sulla base della funzione di trasferimento $G(s)$ ottenuta nel punto di lavoro prima del gradino. Ciò che ne consegue è che il modello lineare stesso, considerando costante la $G(s)$, non tiene conto della variazione a seconda del punto di lavoro della frequenza di risonanza e pertanto l'andamento oscillatorio della tensione predetta sarà errato. Viceversa la rete neurale dinamica, rilevando in ingresso il gradino di duty cycle, modifica i coefficienti della $G(s)$ e di conseguenza riesce a riprodurre l'andamento oscillatorio corretto.

Modello	Err. massimo [V]	Err. massimo [%]	Err. medio [V]
Modello NN	1.74	0.92	0.48
Modello lineare	8.4	4.43	2

Tabella 4.1. Errore massimo e medio commesso dai modelli considerati nella predizione della tensione di uscita nel intervallo [0 ms; 6 ms] del convertitore boost simulato. L'errore massimo percentuale è calcolato rispetto alla tensione massima di uscita pari a 190 V ottenuta nella simulazione del convertitore.

La tabella 4.1 riporta più nel dettaglio gli errori di stima della tensione di uscita relativi ai vari modelli.

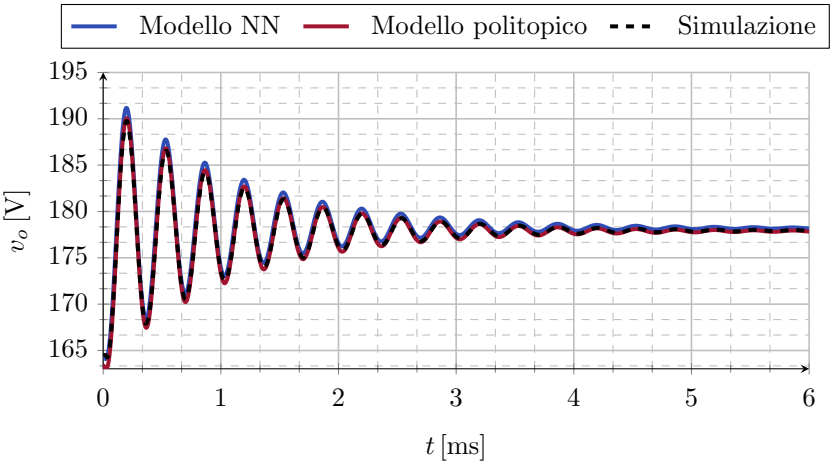


Figura 4.2. Confronto tra la tensione di uscita predetta dal modello NN e quella predetta dal modello politopico a fronte di una variazione a gradino del duty cycle del 7% nell'istante $t = 0$ ms. La corrente di uscita è mantenuta fissata al valore $I_{out} = 2.9$ A.

Nelle figure 4.2 e 4.3 la risposta al gradino ottenuta dal modello NN è confrontata con quella ottenuta dal modello politopico. Si può osservare che entrambi i modelli forniscono complessivamente una buona stima della risposta con qualche errore nella stima stessa nei primi istanti del transitorio. Nello specifico la risposta predetta dal modello politopico presenta una variazione a gradino di circa 1V nell'istante $t = 0$ ms in cui viene applicato il gradino di duty cycle, come mostrato nel secondo grafico della figura 4.3. Quello che

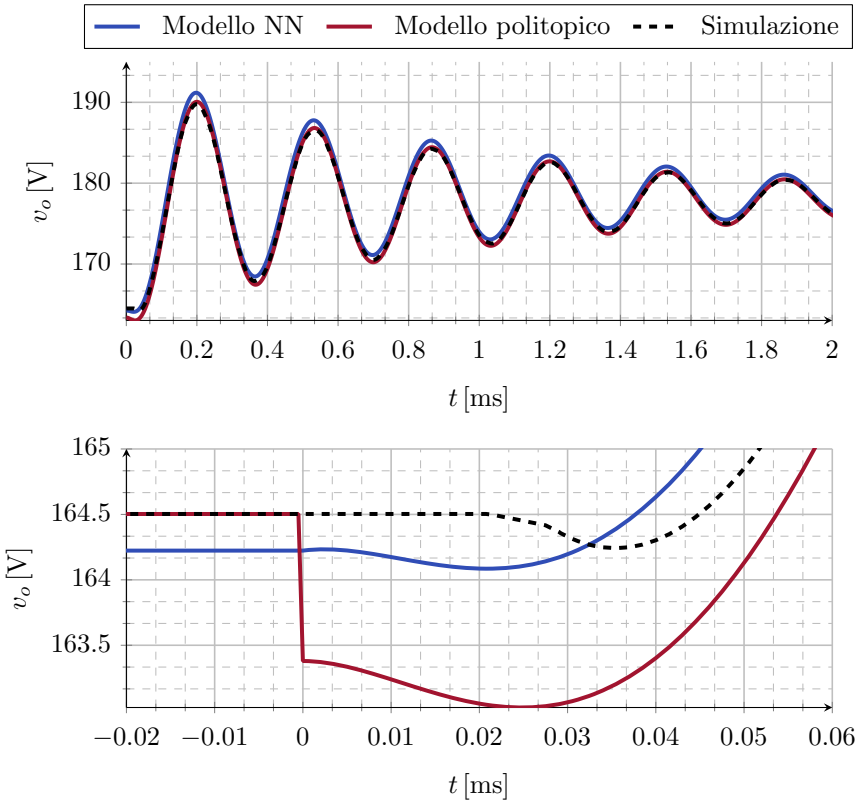


Figura 4.3. Andamento risposte stimate dal modello politopico e dal modello NN negli istanti iniziali del transitorio. Nel secondo grafico sono mostrate le varie risposte ottenute nell'intorno di $t = 0$ ms, istante in cui viene applicato il gradino di duty cycle.

succede nella predizione della risposta da parte del modello politopico è la dinamica seguente.

Prima del gradino sarà maggiormente pesata l'uscita del modello $M1$ relativo al punto di lavoro $D = 10\%$, $I_{out} = 2.9$ A, mentre dopo, a seguito della variazione a gradino del 7%, verrà pesata di più l'uscita del modello $M11$ relativo a $D = 17\%$. Dato che i due modelli $M1$ e $M11$ sono stati ottenuti su due punti di lavoro molto diversi, anche la loro uscita sarà altrettanto diversa. Pertanto l'uscita complessiva del modello politopico in corrispondenza del gradino di duty cycle presenterà una brusca transizione dall'uscita del modello $M2$ all'uscita del modello $M11$: il risultato, come spiegato in [3], è la variazione a gradino della risposta stimata riportata nella figura 4.3.

La risposta del modello NN invece non presenta variazioni brusche ma, a differenza della risposta del modello politopico, essa è caratterizzata da un piccolo errore a regime (lo si può notare sempre nel secondo grafico della figura 4.3) prima dell'applicazione del gradino. Ciò è dovuto all'errore di stima del guadagno K calcolato dalla rete neurale statica.

Un altro aspetto che emerge dal secondo grafico della figura 4.3 è che la tensione di uscita ottenuta in simulazione presenta un leggero ritardo rispetto a quelle predette. Si ricorda infatti che i fitting della funzione di trasferimento $G(s)$ sono stati eseguiti considerando valori della frequenza inferiori a 10 kHz. Dato che i risultati ottenuti nei fitting stessi sono stati utilizzati sia nella costruzione del modello politopico che nell'addestramento della rete neurale dinamica, sarà possibile predire il comportamento del sistema solo nella banda di frequenze $[0\,\text{Hz};\ 10\,\text{kHz}]$. Inoltre si ricorda che la tensione di uscita v_o è ottenuta filtrando quella misurata $v_{o\,meas}$ (vedere figura 3.2) attraverso un filtro a media mobile. Ovviamente tale filtro introduce un ritardo che può essere predetto solo eseguendo i fitting della $G(s)$ per frequenze superiori a f_{sw}. Ma essendo $f_{sw} = 20\,\text{kHz}$ e avendo eseguito i fitting fino alla frequenza 10 kHz, ne segue che il leggero ritardo introdotto dal filtro non sarà predetto né dal modello politopico né dal modello NN.

Err. massimo	1.5 V
Err. massimo percentuale	0.8%
Err. medio	0.1 V

Tabella 4.2. Errori di stima del modello politopico ottenuti nel transitorio complessivo (figura 4.2) nell'intervallo $[20\,\text{ms};\ 26\,\text{ms}]$. L'errore percentuale è calcolato rispetto alla tensione massima in uscita pari a 190 V ottenuta in simulazione.

La tabella 4.2 riporta gli errori di stima ottenuti dal modello politopico. Si può notare che gli errori sono leggermente più piccoli rispetto a quelli al modello NN mostrati nella tabella 4.1. È possibile quindi ottenere una buona stima della risposta del convertitore in termini della tensione di uscita sia con il modello politopico che con il modello NN. Tuttavia si ricorda che il modello politopico considerato in questo specifico caso è stato ottenuto combinando 11 modelli locali ricavati su 11 punti di lavoro diversi del sistema. In un caso più generale può accadere che per identificare un dato sistema sia necessario considerare un numero molto elevato di punti di lavoro e, di conseguenza, un numero altrettanto elevato di modelli locali. Inoltre, come spiegato nel capitolo 2, il numero di modelli locali (e quindi la complessità del modello politopico) aumenta esponenzialmente all'aumentare del numero di variabili in ingresso.

Modello	Err. massimo V	Err. massimo [%]	Err. medio V
Modello NN	1.58	0.84	0.28
Modello politopico	2.25	1.21	0.29

Tabella 4.3. Errori ottenuti nella stima della risposta del convertitore a una serie di variazioni a gradino del duty cycle. L'errore massimo percentuale è calcolato rispetto alle tensione massima di uscita ottenuta nella simulazione del convertitore pari a 187 V.

Nella figura 4.4 è riportato un ulteriore confronto tra il modello NN e il modello politopico. In questo caso, invece di applicare un solo gradino come in precedenza, è stata applicata una sequenza di variazioni a gradino di ampiezza differente. Il tempo

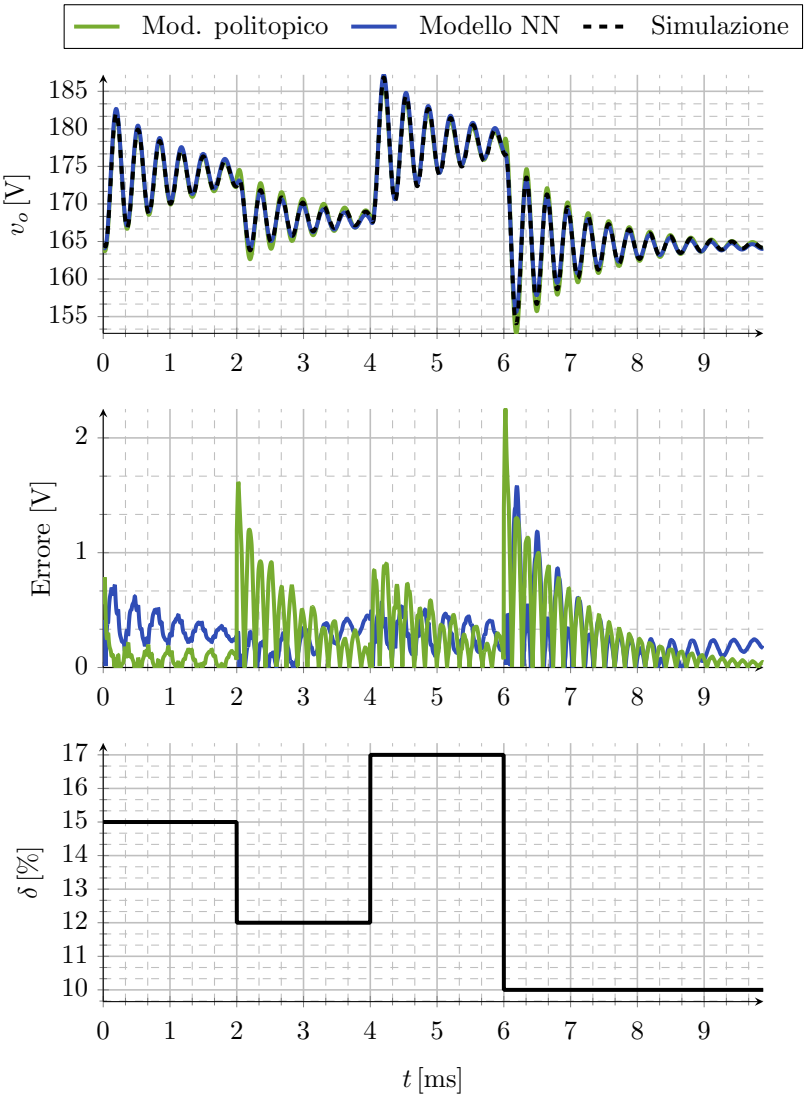


Figura 4.4. Risposta del convertitore in termini della tensione di uscita predetta dal modello politopico e dal modello NN a fronte di alcune variazioni a gradino del duty cycle. Il secondo e il terzo grafico riportano rispettivamente l'errore di stima dei due modelli e l'andamento a gradini del duty cycle δ applicato. Nella simulazione la corrente di uscita è stata fissata a $I_{out} = 2.9$ A.

che intercorre tra l'applicazione di un gradino e il successivo è di 2 ms. Gli errori di stima ottenuti sono illustrati nella tabella 4.3. L'errore maggiore è ottenuto dal modello politopico negli istanti corrispondenti alle variazioni a gradino come si può osservare dal secondo grafico in figura 4.4. Questo perché, da quanto visto precedentemente, in corrispondenza di una variazione a gradino del duty cycle anche l'uscita del modello politopico varia a gradino (vedere il secondo grafico della figura 4.3) determinando così un

evidente errore sulla risposta predetta. Complessivamente le oscillazioni della tensione di uscita vengono riprodotte in maniera accurata sia dal modello politopico che dal modello NN con una buona stima della frequenza di risonanza e dello smorzamento.

4.2 *Risposta alla rampa*

Oltre alla risposta al gradino, sono state effettuate delle simulazioni per valutare la predizione della risposta alla rampa di duty cycle. La figura 4.5 mostra la tensione di uscita del convertitore predetta dai vari modelli black-box a fronte di alcune variazioni a rampa del duty cycle. Come già discusso nel paragrafo precedente, il modello lineare non tiene conto, a differenza del modello politopico e del modello NN, della diversa dinamica del convertitore al cambiare del punto di lavoro. Pertanto il modello lineare presenterà l'errore di stima maggiore come si può osservare dal secondo grafico in figura 4.5 che riporta l'errore di stima nel tempo commesso dai vari modelli e dai valori in tabella 4.4.

Modello	Err. massimo [V]	Err. massimo [%]	Err. medio V
Modello lineare	1.01	0.57	0.3
Modello politopico	0.08	0.05	0.06
Modello NN	0.5	0.28	0.2

Tabella 4.4. Errori di stima della risposta del convertitore a una serie di variazioni a rampa del duty cycle. L'errore massimo percentuale è calcolato rispetto alla tensione massima osservata in simulazione pari a 178V.

Il modello politopico, diversamente da quanto accadeva con la risposta al gradino, presenta degli errori molto più piccoli. Inoltre non si osservano più quelle variazioni a gradino nella risposta stimata (vedere il secondo grafico della figura 4.3). Infatti in questo caso il duty cycle δ varia linearmente nel tempo e con esso anche le funzioni di peso del modello politopico. Quindi se si prende ad esempio la prima variazione a rampa del duty cycle nell'intervallo [0 ms; 5 ms] riportata nell'ultimo grafico, quello che succede è che inizialmente nel modello politopico verrà pesato maggiormente il primo modello locale $M1$ relativo al punto di lavoro $D = 10\%$, $I_{out} = 2.9$A. Successivamente all'aumentare del duty cycle quest'ultimo verrà pesato sempre di meno e comincerà ad *attivarsi* il modello locale adiacente $M2$ (associato al punto di lavoro in termini di duty cycle $D = 10.7\%$) che vedrà la rispettiva funzione di peso aumentare. Di nuovo con l'aumento graduale del duty cycle, l'uscita del modello $M2$ sarà pesata sempre di meno a favore dell'uscita del modello $M3$ che invece verrà pesata sempre di più. Dopo un po' di tempo sarà pesato maggiormente il modello $M4$ e, se il duty cycle continua ad aumentare, sarà la volta del modello $M5$ e così via. Il fatto che il duty cycle aumenti a rampa fa sì che i modelli locali vengano *ordinatamente attivati* uno alla volta: prima $M1$, poi $M2$, poi $M3$, etc …Il risultato è che l'uscita complessiva del modello politopico sarà priva di variazioni brusche in quanto modelli locali adiacenti ottenuti su punti di lavoro molto vicini presentano uscite molto simili. Al contrario in presenza di variazioni a gradino del duty cycle non si ha più questo meccanismo. Di fatto si era visto che nel caso di un gradino di ampiezza pari a 7% (figura 4.3), prima è *attivo* il modello $M1$ e poi a seguito del gradino, invece

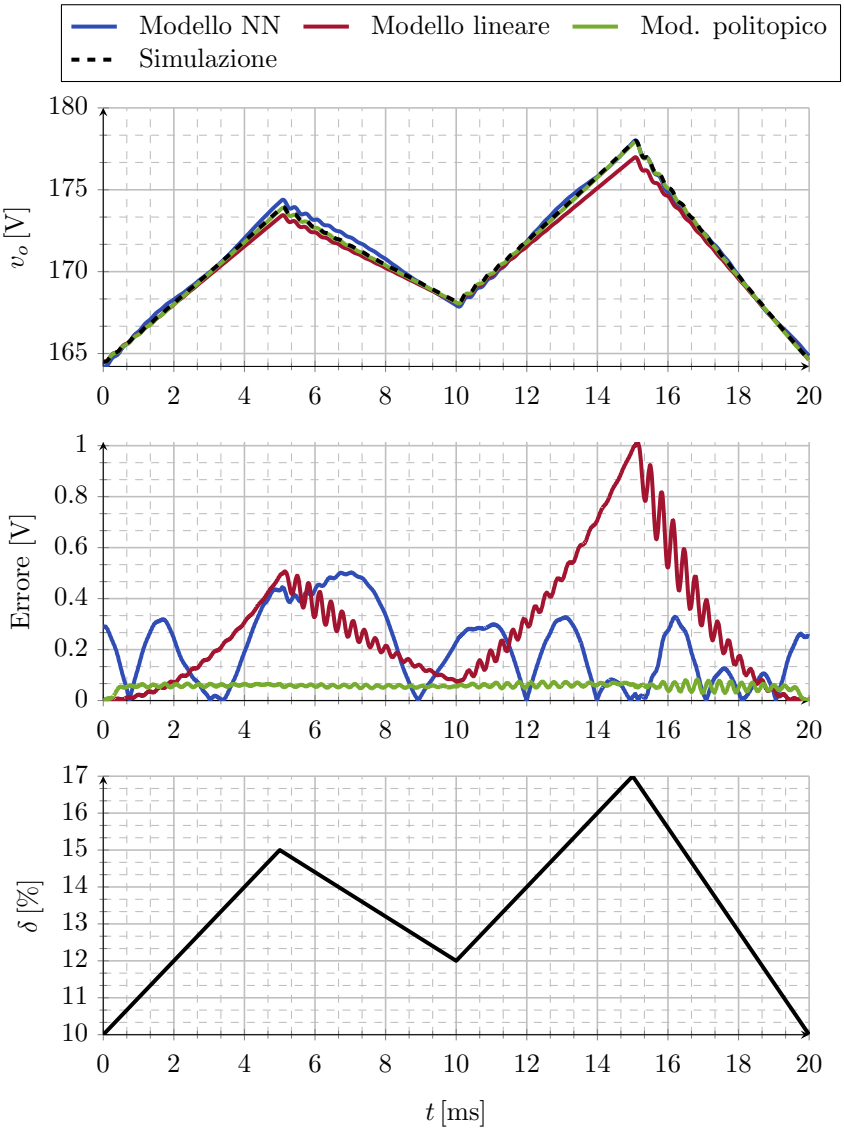


Figura 4.5. Risposta alla rampa del convertitore in termini della tensione di uscita predetta dai modelli black-box considerati. Il secondo grafico riporta nel tempo i rispettivi errori di stima, mentre l'ultimo grafico riporta l'andamento del duty cycle applicato. La corrente di uscita è stata mantenuta costante al valore $I_{out} = 2.9$ A.

che venire *attivato* il modello adiacente $M2$, viene *attivato* il modello $M11$ la cui uscita è molto diversa rispetto a quella del modello $M1$ stesso. Ciò comporta che nella risposta al gradino l'errore del modello politopico sarà maggiore rispetto a quello ottenuto nella risposta alla rampa.

Il modello NN commette degli errori maggiori rispetto al modello politopico; in

particolare l'errore di stima è più elevato nell'intervallo [4 ms; 8 ms]. Tale errore è dovuto alla rete neurale dinamica e alla rete neurale statica le quali commettono a loro volta degli errori nella stima dei coefficienti della funzione di trasferimento $G(s)$ e del guadagno K.

4.3 *Stima della risposta per diversi valori della corrente di uscita e della frequenza di taglio del filtro LPF*

Nel modello NN la frequenza di taglio del filtro LPF è stata fissata a 100 Hz dato che tale frequenza è molto distante dai poli e dagli zeri che la funzione di trasferimento $G(s)$ variabile tramite la rete neurale dinamica può avere. Per capire l'influenza del filtro LPF sulla stima della risposta, si sono effettuate alcune simulazioni per diversi valori della frequenza di taglio f_{LPF}. La figura 4.6 riporta le risposte ottenute al variare della frequenza

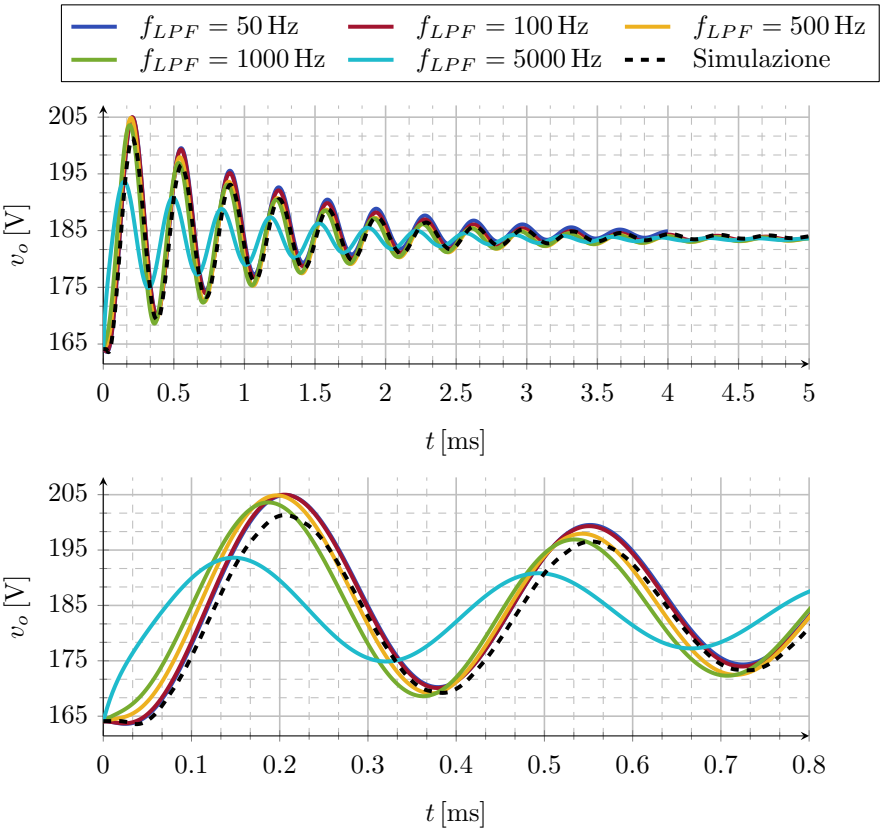


Figura 4.6. Risposta al gradino di duty cycle di ampiezza pari a 10% predetta dal modello NN per diversi valori della frequenza di taglio f_{LPF} del filtro LPF. Nel secondo grafico è riportato un *ingrandimento* nell'intervallo [0 ms; 0.8 ms]. La corrente di uscita è fissata a $I_{out} = 3.5$ A.

di taglio del filtro LPF. Come si può osservare all'aumentare della frequenza di taglio, la dinamica del filtro interferisce con la dinamica della funzione di trasferimento $G(s)$. Ciò

comporta un errore nella riproduzione del transitorio, in particolare, nella stima della frequenza di risonanza.

Freq. di taglio	Err. massimo [V]	Err. massimo [%]	Err. medio
50	4	1.9	0.63
100	4.2	2.1	0.68
500	6.4	3.2	0.86
1000	9.1	4.5	1.24
5000	17	8.4	2.65

Tabella 4.5. Errori ottenuti nella stima della risposta per diversi valori della frequenza di taglio del filtro LPF. L'errore massimo in percentuale è riferito a una tensione di uscita massima ottenuta in simulazione pari a 202 V.

Gli errori ottenuti al variare della frequenza di taglio f_{LPF} sono mostrati nella tabella 4.5. I casi peggiori sono associati a $f_{LPF} = 5000$ Hz e a $f_{LPF} = 1000$ Hz. Infatti per tale valore della frequenza di taglio il transitorio risulta completamente distorto dato che il filtro LPF agisce a una frequenza prossima ai poli e agli zeri della funzione di trasferimento $G(s)$. Al contrario per $f_{LPF} = 50$ Hz e per $f_{LPF} = 100$ LPF questo non accade e l'oscillazione del transitorio viene riprodotta correttamente.

Oltre a variare la frequenza di taglio, sono state effettuate delle simulazioni cambiando anche il valore della corrente di uscita I_{out}. La figura 4.7 riporta la tensione di uscita predetta sia dal modello NN che dal modello politopico a fronte di una serie di variazioni a gradino di duty cycle come quelle della figura 4.4. La differenza è che questa volta la corrente di uscita, anziché essere fissata a un valore pari a $I_{out} = 2.9$ A, è posta pari a $I_{out} = 4.5$ A. Dal momento che il modello politopico è stato *costruito* considerando punti di lavoro corrispondenti a una corrente di uscita $I_{out} = 2.9$ A, esso presenterà un errore di stima maggiore rispetto al modello NN. Quest'ultimo infatti sarà in grado di stimare la tensione di uscita del convertitore anche per diversi valori della corrente avendo addestrato le reti neurali nell'intervallo $I_{out} \in [2.5\text{ A}; 4.5\text{ A}]$. Nella tabella 4.6 sono

Modello	Err. massimo [V]	Err. massimo [%]	Err. medio
Modello NN	1.96	1.1	0.46
Modello politopico	3.47	1.9	1.41

Tabella 4.6. Errori di stima ottenuti nella risposta predetta a una serie di gradini di duty cycle con la corrente di uscita I_{out} fissata a un valore pari a 4.5 A. L'errore massimo in percentuale è riferito a una tensione di uscita massima ottenuta in simulazione pari a 186 V.

illustrati gli errori di stima ottenuti. Notare inoltre che, per quanto appena detto, tali errori sono più grandi di quelli riportati nella tabella 4.3 che erano stati ottenuti considerando una corrente di uscita pari a $I_{out} = 2.9$ A. Naturalmente se si volesse migliorare la stima del modello politopico, sarebbe necessario aumentare il numero di modelli locali in modo

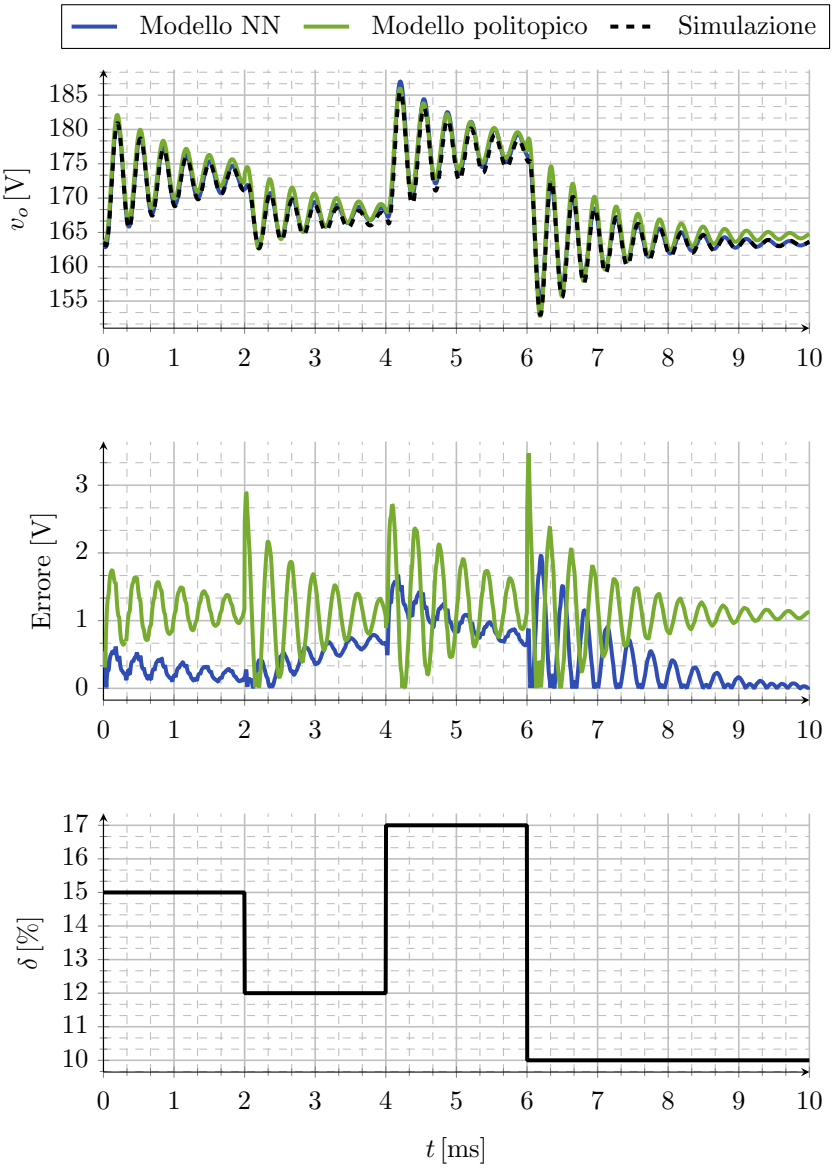


Figura 4.7. Risposta del convertitore in termini della tensione di uscita predetta dal modello NN e dal modello politopico a una serie di gradini di duty cycle. La corrente di uscita è fissata a $I_{out} = 4.5\,\text{A}$.

da considerare il comportamento del convertitore per diversi valori di I_{out}; ma come già visto ciò comporterebbe un aumento della complessità del modello politopico stesso.

Infine si osservi dalla figura 4.7 che, sebbene il modello politopico sia stato ottenuto per una corrente diversa da 4.5 A, esso è comunque in grado di riprodurre l'oscillazione

del transitorio in quanto (come visto nel capitolo 3 figura 3.5) la frequenza di risonanza è indipendente dal valore della corrente di uscita I_{out}.

4.4 *Confronto con la rete neurale NARX*

Nelle ultime simulazioni si è confrontata la risposta predetta dal modello NN con quella predetta dalla rete neurale NARX introdotta nel capitolo 2. La rete NARX è stata addestrata variando a gradino sia il duty cycle δ nell'intervallo $[10\%;\ 30\%]$, sia la corrente di uscita I_{out} nell'intervallo $[2.5\,\text{A};\ 4.5\,\text{A}]$ come descritto sempre nel capitolo 2. I punti di lavoro considerati nella fase di addestramento della NARX sono gli stessi considerati nell'addestramento della rete neurale statica e della rete neurale dinamica del modello NN, mentre la frequenza di campionamento impiegata nell'acquisire la tensione di uscita è stata fissata a $F_s = 20\,\text{kHz}$. La configurazione migliore ottenuta per la rete NARX è

Struttura rete	$40 - 20 - 15$
Numero di parametri	3655
Numero di neuroni	75
Funzione di attivazione	*sigmoide*

Tabella 4.7. Configurazione scelta per la rete neurale NARX.

riportata nella tabella 4.7. Per comparare le prestazioni del modello NN e della rete NARX stessa nella predizione della tensione di uscita del convertitore, si sono applicate diverse variazioni a gradino sul duty cycle mantenendo la corrente di uscita I_{out} costante pari a 4 A.

Nella figura 4.8 sono mostrati i transitori ottenuti e l'errore di stima commesso dai vari modelli. Si ricorda che la rete NARX implementa un'equazione alle differenze la quale descrive il comportamento del convertitore a tempo discreto. Più precisamente la tensione di uscita predetta $\tilde{v}_o(k)$ dalla rete al passo k corrisponde alla tensione di uscita predetta $\tilde{v}_o(kT_s)$ al tempo kT_s, dove $T_s = 1/F_s = 0.05$ ms è il tempo di campionamento. Invece il modello NN fornisce direttamente la risposta a tempo continuo del sistema. Pertanto per effettuare un confronto tra i due modelli si è campionata l'uscita del modello NN a frequenza F_s.

Modello	Err. massimo [V]	Err. massimo [%]	Err. medio [V]
Modello NN	7.3	3.3	1.23
Rete NARX	3.63	1.64	1.05

Tabella 4.8. Prestazioni del modello NN e della rete neurale NARX in termini di errore di stima nella predizione dei transitori riportati nella figura 4.8. L'errore massimo è riferito alla tensione di uscita massima ottenuta in simulazione pari a 220 V.

Si può notare che entrambi i modelli black-box riescono a riprodurre abbastanza fedelmente l'andamento oscillatorio della tensione di uscita del convertitore. Tuttavia

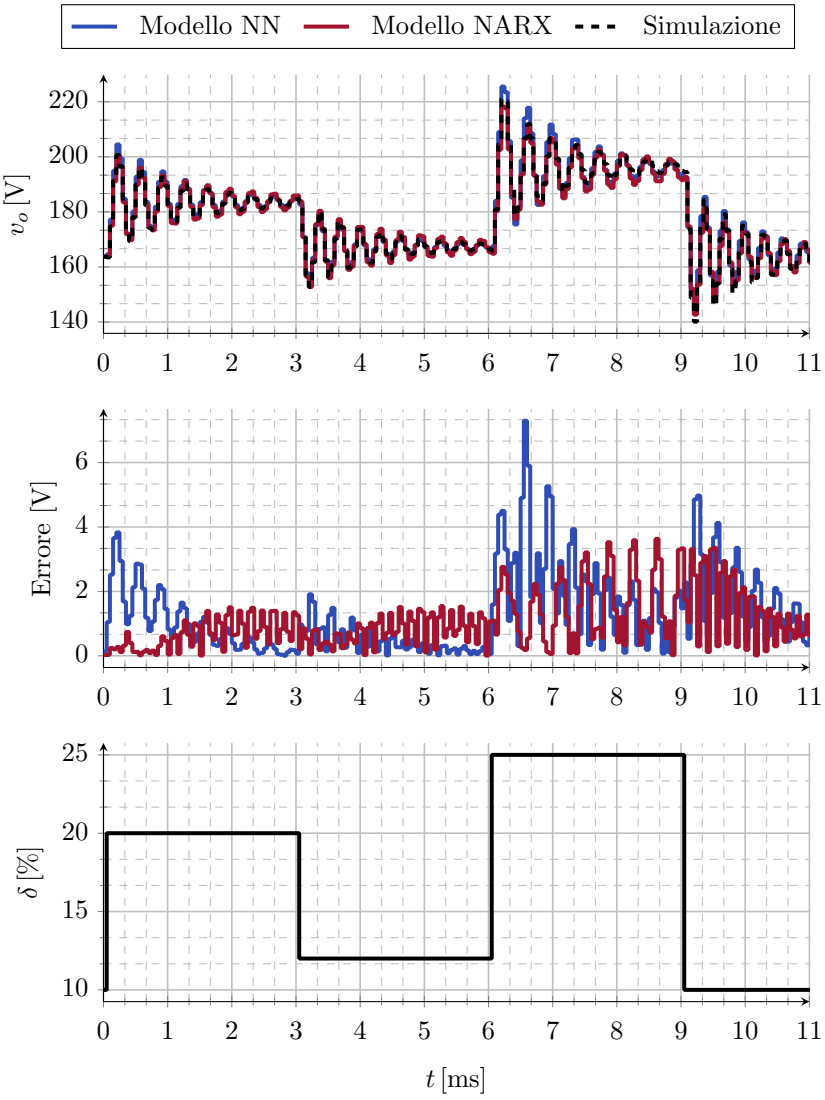


Figura 4.8. Risposta del convertitore a una serie di variazioni a gradino del duty cycle predetta dal modello NN e dalla rete neurale NARX. La corrente di uscita I_{out} è fissata a 4 A.

si osservano degli errori commessi nella stima dell'ampiezza dell'oscillazione da parte del modello NN. In particolare gli errori massimi si ottengono a fronte delle variazioni a gradino di duty cycle di ampiezza più elevata negli istanti $t = 6.5$ ms e $t = 9.3$ ms. Le prestazioni in termini di errore di stima sono riportate nella tabella 4.8. Complessivamente la rete NARX fornisce una predizione della risposta migliore rispetto al modello NN commettendo un errore medio di stima pari a circa 1 V.

In figura 4.9 è riportata un'altra serie di transitori stimati dai due modelli. In questo caso

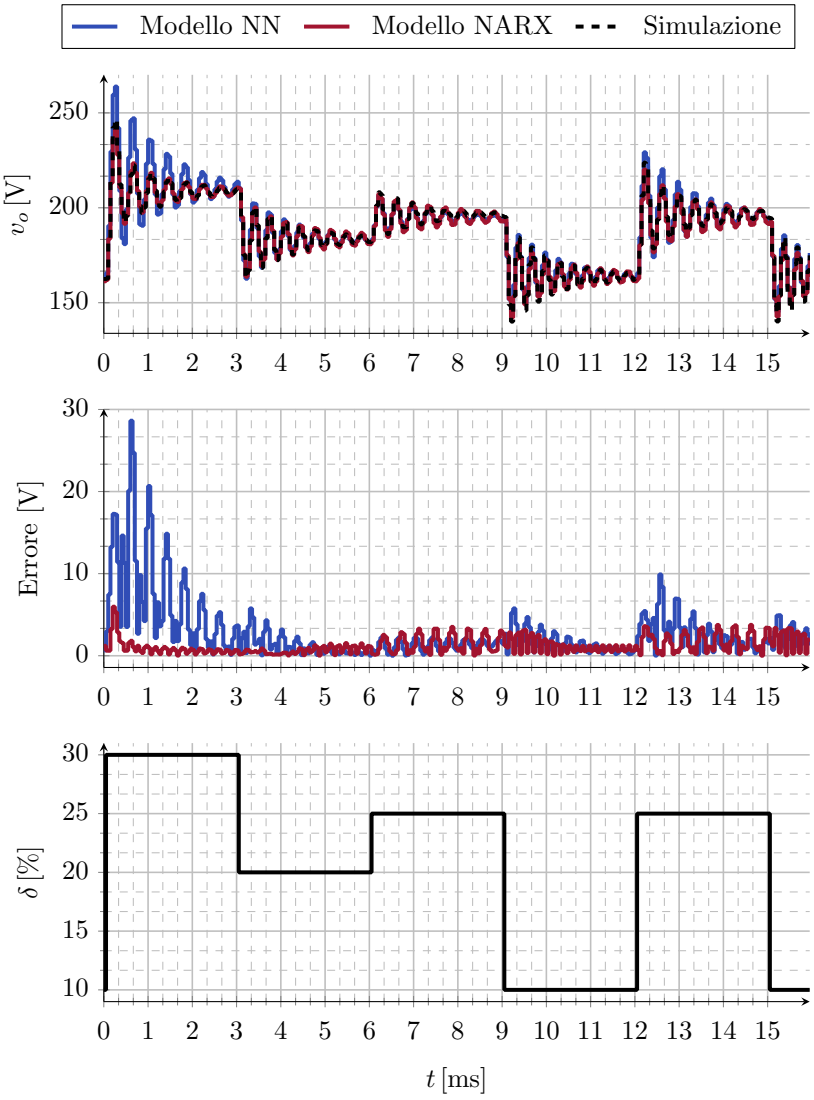


Figura 4.9. Risposta predetta, in termini della tensione di uscita, dal modello NN e dalla rete NARX a fronte di variazioni a gradino di duty cycle.

si sono effettuate delle variazioni a gradino di duty cycle di ampiezza molto più grande rispetto a prima. Più nel dettaglio, nell'istante iniziale $t = 0$ ms è stato applicato un gradino dal 10% al 30%. Quello che si osserva è che, nell'intorno di $t = 0$ ms, l'errore commesso dalla modello NN è molto più grande rispetto a quello riscontrato precedentemente in figura 4.8. Numericamente si ha un errore di stima massimo pari a circa 29 V (valore corrispondente al 12% della tensione massima $v_{o\,max} = 245$ V ottenuta in simulazione) come riportato in tabella 4.9.

Modello	Err. massimo [V]	Err. massimo [%]	Err. medio [V]
Modello NN	28.63	11.61	2.76
Rete NARX	5.97	2.42	1.24

Tabella 4.9. Prestazioni del modello NN e della rete neurale NARX in termini di errore di stima nella predizione dei transitori riportati nella figura 4.9. L'errore massimo è riferito alla tensione di uscita massima ottenuta in simulazione pari a 245 V.

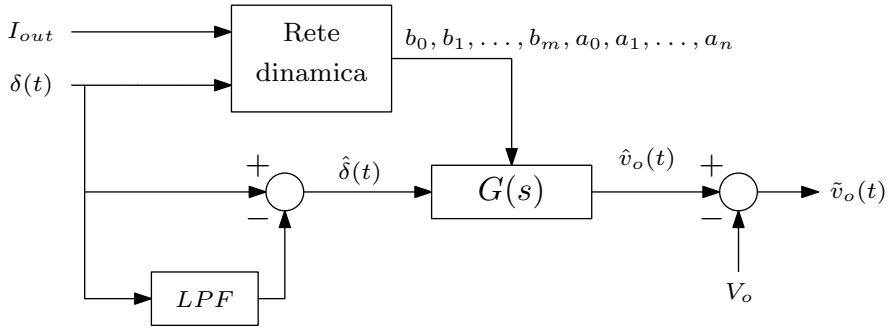


Figura 4.10. Schema del modello NN nel calcolo della tensione di uscita.

Al fine di spiegare perché si ottengono errori elevati in corrispondenza di grandi variazioni a gradino del duty cycle, si consideri lo schema in figura 4.10 il quale riassume come il modello NN stima la tensione di uscita del convertitore boost $\tilde{v}_o(t)$ a partire dal duty cycle $\delta(t)$ per un valore fissato della corrente di uscita I_{out}. Si supponga di applicare un gradino sul duty cycle da un valore D_1 a un valore D_2 nell'istante $t = 0$ ms. In tale istante la rete neurale dinamica varia in modo istantaneo i coefficienti della funzione di trasferimento $G(s)$ dato che il punto di lavoro si *sposta* da $(D_1;\, I_{out})$ a $(D_2;\, I_{out})$. Naturalmente per $t > 0$ ms il duty cycle $\delta(t)$ rimane costante e quindi anche i coefficienti stessi rimangono costanti. Ciò che ne consegue è che per $t > 0$ ms, subito dopo il gradino di duty cycle, il modello NN si comporta dal punto di vista della perturbazione $\hat{v}_o(t)$ come il modello linearizzato nel punto di lavoro $(D_2,\, I_{out})$ essendo fissati i coefficienti della $G(s)$. Se quindi la variazione imposta sul duty cycle è elevata in termini di ampiezza, allora anche la variazione sulla perturbazione $\hat{\delta}(t)$ sarà elevata e, di conseguenza, la stima della perturbazione sulla tensione $\hat{v}_o(t)$ non sarà corretta dal momento che la relazione $\hat{v}_o(s) = G(s)\,\hat{\delta}(s)$ associata al modello linearizzato nel punto di lavoro $(D_2,\, I_{out})$ è valida solo per valori abbastanza contenuti di $\hat{\delta}(t)$.

5

CONCLUSIONI

In questa tesi sono stati presentati e analizzati alcuni metodi per la modellizzazione black-box di convertitori di potenza dc-dc. In particolare si è sviluppato un modello black-box (chiamato "Modello NN") basato su reti neurali feed-forward e finalizzato a stimare l'andamento della tensione di uscita di un convertitore boost in funzionamento CCM al variare del duty cycle. Tale modello è stato poi confrontato con un modello politopico composta da 11 modelli locali e con un secondo modello basato sulla rete neurale NARX.

Dai risultati ottenuti nella simulazione dei vari approcci black-box, si è visto che il modello NN, il modello politopico e la rete NARX riescono a riprodurre fedelmente l'oscillazione della tensione in uscita del convertitore boost a seguito di variazioni a gradino del duty cycle. Il modello politopico nello specifico presenta tuttavia delle variazioni brusche della tensione predetta negli istanti corrispondenti alle variazioni a gradino stesse, mentre il modello NN commette errori di stima considerevoli quando l'ampiezza dei gradini è elevata.

Dal punto di vista della complessità, il numero di modelli locali che compongono il modello politopico aumenta in modo lineare all'aumentare del numero di punti di lavoro e in modo esponenziale all'aumentare del numero di ingressi. Per quanto riguarda la rete NARX, essa presenta una struttura più semplice rispetto al modello NN il quale comprende due reti neurali invece che una singola. Di contro il modello NN e il modello politopico contengono già intrinsecamente l'informazione sulla funzione di trasferimento $G(s)$ corrispondente alla dinamica *linearizzata* del convertitore. Infatti se si volesse determinare il modello linearizzato del sistema in un dato punto di lavoro a partire dal modello basato sulla NARX, sarebbe necessario svolgere un'analisi nel dominio della frequenza perturbando gli ingressi della rete NARX stessa e osservandone le uscite. Invece nel modello NN è sufficiente fornire in ingresso il punto di lavoro alla rete neurale dinamica che quindi calcolerà direttamente i coefficienti della funzione di trasferimento $G(s)$.

Infine si sottolinea che la rete neurale NARX è stata addestrata valutando la risposta al gradino del convertitore e pertanto il comportamento in alta frequenza di quest'ultimo non può essere stimato in modo corretto (lo spettro di un segnale ad onda quadra diminuisce all'aumentare della frequenza). Il problema non si presenta invece per il modello NN e per il modello politopico dato che nella stima della funzione di trasferimento $G(s)$ nei diversi punti di lavoro il sistema viene eccitato anche alle alte frequenze.

BIBLIOGRAFIA

[1] Luis Arnedo, Dushan Boroyevich, Rolando Burgos e Fred Wang. «Polytopic black-box modeling of dc-dc converters». en. In: *2008 IEEE Power Electronics Specialists Conference*. ISSN: 0275-9306. Rhodes, Greece: IEEE, giu. 2008, pp. 1015–1021. ISBN: 978-1-4244-1667-7. DOI: 10.1109/PESC.2008.4592063. URL: http://ieeexplore.ieee.org/document/4592063/.

[2] Zina Boussaada, Octavian Curea, Ahmed Remaci, Haritza Camblong e Najiba Mrabet Bellaaj. «A Nonlinear Autoregressive Exogenous (NARX) Neural Network Model for the Prediction of the Daily Direct Solar Radiation». en. In: *Energies* 11.3 (mar. 2018), p. 620. ISSN: 1996-1073. DOI: 10.3390/en11030620. URL: http://www.mdpi.com/1996-1073/11/3/620.

[3] Airan Frances, Rafael Asensi e Javier Uceda. «Blackbox Polytopic Model With Dynamic Weighting Functions for DC-DC Converters». en. In: *IEEE Access* 7 (2019), pp. 160263–160273. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2950983. URL: https://ieeexplore.ieee.org/document/8890629/.

[4] Pouria Qashqai, Rawad Zgheib e Kamal Al-Haddad. «GRU and LSTM Comparison for Black-Box Modeling of Power Electronic Converters». en. In: *IECON 2021 – 47th Annual Conference of the IEEE Industrial Electronics Society*. Toronto, ON, Canada: IEEE, ott. 2021, pp. 1–5. ISBN: 978-1-66543-554-3. DOI: 10.1109/IECON48115.2021.9589609. URL: https://ieeexplore.ieee.org/document/9589609/ (visitato il 07/02/2024).

[5] Gabriel Rojas-Duenas, Jordi-Roger Riba e Manuel Moreno-Eguilaz. «A Deep Learning-Based Modeling of a 270 V-to-28 V DC-DC Converter Used in More Electric Aircrafts». en. In: *IEEE Transactions on Power Electronics* 37.1 (gen. 2022), pp. 509–518. ISSN: 0885-8993, 1941-0107. DOI: 10.1109/TPEL.2021.3098468. URL: https://ieeexplore.ieee.org/document/9492829/.

[6] Amin Salehi e Morteza Montazeri-Gh. «Black box modeling of a turboshaft gas turbine engine fuel control unit based on neural NARX». en. In: *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* 233.3 (ago. 2019), pp. 949–956. ISSN: 1475-0902, 2041-3084. DOI: 10.1177/1475090218797496. URL: http://journals.sagepub.com/doi/10.1177/1475090218797496.

[7] Virgilio Valdivia Guerrero. «Behavioral Modeling and Identification of Power Electronics Converters and Subsystems Based on Transient Response». Tesi di dott. Università Carlos III di Madrid, 2013.

[8] Andrea Zilio, Davide Biadene, Tommaso Caldognetto e Paolo Mattavelli. «Black-Box Large-Signal Average Modeling of DC-DC Converters Using NARX-ANNs». en. In: *IEEE Access* 11 (2023), pp. 43257–43266. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3271731. URL: https://ieeexplore.ieee.org/document/10113325/.