

# Data Model Identification and Intelligent Control

Stefano Di Lena

2025



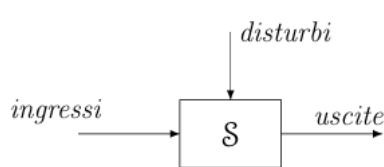
# Indice

<b>1</b>	<b>Identificazione dei Sistemi</b>	<b>1</b>
1.1	Modelli matematici . . . . .	2
1.1.1	Predizione . . . . .	2
1.1.2	Simulazione . . . . .	2
1.1.3	Ottimizzazione . . . . .	3
1.1.4	Controllo . . . . .	3
1.1.5	Analisi . . . . .	3
1.1.6	Fault Detection . . . . .	3
<b>2</b>	<b>Famiglie di Modelli</b>	<b>4</b>
2.1	Box-Jenkins . . . . .	5
2.1.1	OE (Output Error) . . . . .	5
2.1.2	IE (Input Error) . . . . .	5
2.1.3	ARX (AutoRegressive with eXogenous variables) . . . . .	6
2.1.4	ARMAX . . . . .	6
2.1.5	ARARX . . . . .	7
<b>3</b>	<b>Criterio di Scelta</b>	<b>7</b>
3.1	Estrazione del predittore . . . . .	7
3.1.1	Predittore ad un passo . . . . .	7
3.1.2	ARX . . . . .	9
3.1.3	ARMAX . . . . .	9
<b>4</b>	<b>Validazione</b>	<b>9</b>
4.1	Analisi Asintotica . . . . .	9
4.2	Analisi di Bianchezza . . . . .	11
<b>5</b>	<b>Ottimizzazione ai Minimi Quadrati</b>	<b>11</b>
5.1	Analisi Probabilistica . . . . .	13
5.1.1	Persistente eccitazione . . . . .	14
5.2	Identificabilità . . . . .	16
5.3	Ordine Ottimo di un Modello . . . . .	16
5.4	Analisi Spettrale . . . . .	18
5.4.1	Pre-filtraggio . . . . .	19
5.5	Operazioni di Pre-processing sui Dati . . . . .	20
5.6	Algoritmo dei minimi quadrati estesi . . . . .	21
<b>6</b>	<b>Identificazione di sistemi non lineari</b>	<b>22</b>
6.1	Modelli a dinamica esterna . . . . .	22
6.2	Modelli di Hammerstein e Wiener . . . . .	23
6.3	Scelta del segnale di Ingresso . . . . .	23
6.4	Modelli in forma di variazione . . . . .	25
6.4.1	Modelli affini nell'azione di controllo . . . . .	26

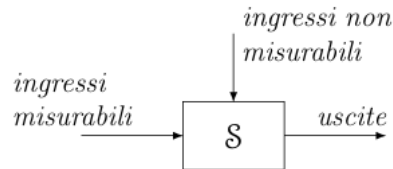
<b>7</b>	<b>Approssimatori</b>	<b>26</b>
7.1	Famiglie . . . . .	27
<b>8</b>	<b>Neural Networks</b>	<b>28</b>
8.1	Backpropagation . . . . .	32
8.1.1	Strategie di Apprendimento . . . . .	34
8.1.2	Error backpropagation: algorithm . . . . .	34
8.2	Radial basis function network . . . . .	35
8.3	Considerazioni . . . . .	36

# 1 Identificazione dei Sistemi

*Identificare* significa trovare un modello matematico basandosi su **dati sperimentali**. In generale, un sistema è un insieme di componenti interagenti che produce una o più uscite, influenzate da ingressi e disturbi. Nel modello che lo descrive si distinguono, quindi, cause ed effetti.



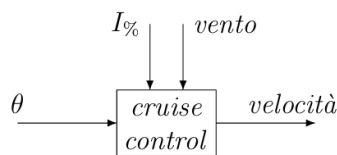
(a) modello per il controllo



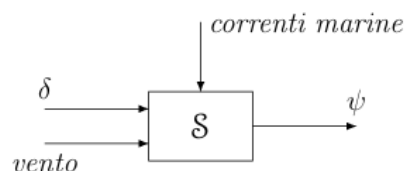
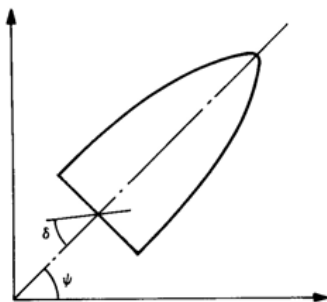
(b) modello per l'identificazione

I *disturbi* sono spesso cause non manipolabili (non sempre sono misurabili).

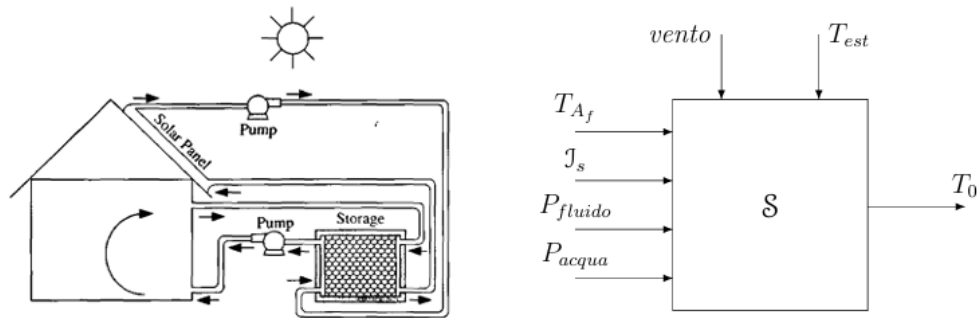
**Esempio 1: cruise control** Possiamo controllare la velocità  $v$  di un veicolo, tramite un ingresso  $\theta$  (angolo di apertura dal pedale acceleratore). Durante la marcia il sistema è sollecitato da una serie di disturbi: la pendenza della strada ( $I\%$ ), la presenza di vento, etc.



**Esempio 2: rotta veicolo marittimo** Avendo una barca a vela, gli ingressi misurabili possono essere  $\delta$  (angolo del timone, misurato rispetto alla direzione della rotta desiderata) ed il vento (considerato come spinta riesco ad ottenere). Le correnti marine sono perturbazioni del sistema e le considero come ingressi non misurabili. L'angolo  $\psi$  sarà l'uscita del sistema e fornisce la rotta finale.



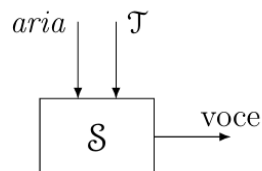
**Esempio 3: solare-termico** Un sistema per il controllo di accumulo dell'acqua calda, basato su energia solare.



La temperatura dell'acqua è l'uscita del sistema (valore che si vuole controllare).

Gli ingressi misurabili sono: l'irraggiamento solare  $J_s$ , la temperatura dell'acqua fredda  $T_A$ , la portata per il fluido di scambio e la portata in uscita dell'acqua calda. Un ingresso non misurabile potrebbe essere la temperatura dell'ambiente esterno.

**Esempio 4: voce** La voce umana è determinata dalla tensione delle corde vocali  $\mathcal{T}$  e l'aria. Questi sono però ingressi non misurabili, quindi il modello matematico prende il nome di *serie temporale* (perché non ha ingressi misurabili).



## 1.1 Modelli matematici

Un *buon* modello è un modello che ha una buona precisione e raggiunge l'obiettivo prefissato.

### 1.1.1 Predizione

Fornisce una predizione dell'uscita basandosi sulla storia precedente del sistema.

### 1.1.2 Simulazione

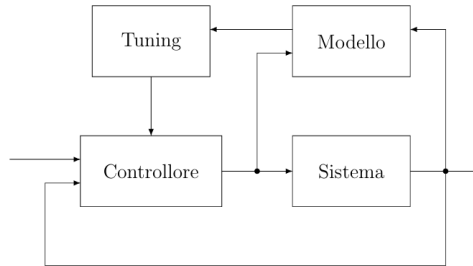
Uso la storia passata dell'ingresso per simulare l'uscita. Dato che non si hanno informazioni sull'uscita passata, i valori in uscita da questo modello non è detto coincidano con quelli veri.

### 1.1.3 Ottimizzazione

Si vuole ottenere un miglioramento dei risultati, controllando e reiterando eventualmente il processo.

### 1.1.4 Controllo

Si applicano delle leggi di controllo che riescano ad adattare i parametri alle modifiche del sistema avvenute durante lo svolgimento.

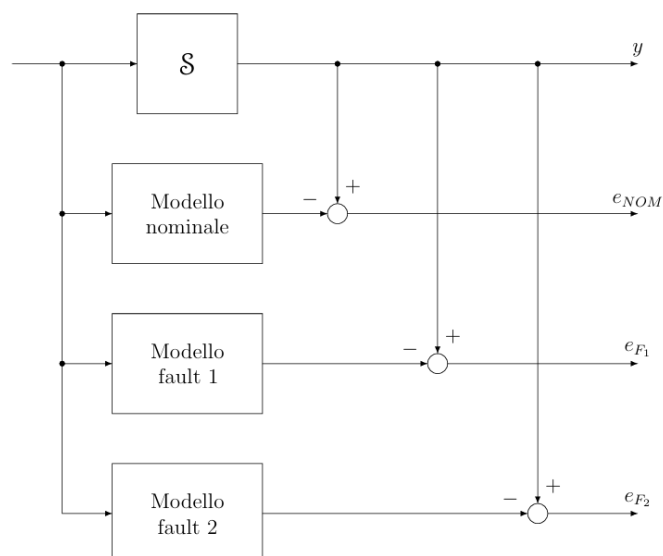


### 1.1.5 Analisi

Il sistema pone delle domande alle quali il modello fornisce delle potenziali risposte.

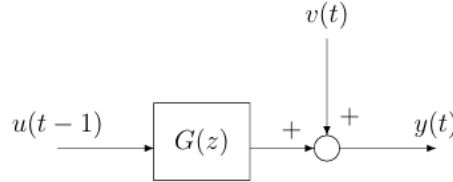
### 1.1.6 Fault Detection

Per effettuare la diagnosi dei guasti è usato questo tipo di modello. Vengono calcolati gli errori fra l'uscita del sistema e le uscite di ciascun modello, riuscendo ad ottenere una misura numerica di quanto si è vicini ad una condizione di guasto.



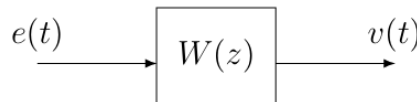
## 2 Famiglie di Modelli

Analizziamo sistemi LTI (Lineari Tempo Invarianti), SISO (Single-Input Single-Output), TD (Tempo Discreto), con funzione di trasferimento strettamente propria (grado del numeratore minore del grado del denominatore).



Nello schema in teoria dovrebbero stare  $Z^{-1}\{u(t-1)\}$  e  $Z^{-1}\{y(t)\}$ . È stato fatto un abuso di notazione per alleggerire la trattazione degli schemi.

In figura  $u(t)$  è detto *variabile esogena* (che proviene dall'esterno),  $v(t)$  *variabile di rumore* (ed è un segnale canonico in grado di descrivere i disturbi sconosciuti e non misurabili).



L'ingresso  $e(t)$  è un rumore bianco, segnale casuale per eccellenza. Esso ha uno spettro uniforme a tutte le frequenze ed è un esempio di segnale probabilistico **i.i.d.** (identicamente distribuito).

Se la f.d.t.  $W(z)$  è asintoticamente stabile, lo spettro del segnale in uscita è pari al prodotto tra il modulo al quadrato della funzione di risposta armonica discreta e la densità spettrale del segnale di ingresso:

$$\phi_v(\omega) = |W(e^{j\omega})|^2 \cdot \phi_e(\omega)$$

Possiamo esprimere una funzione di trasferimento in modo parametrico:

$$G(z) = k \frac{z^m + b_1 z^{m-1} + \dots + b_m}{z^n + a_1 z^{n-1} + \dots + a_n}$$

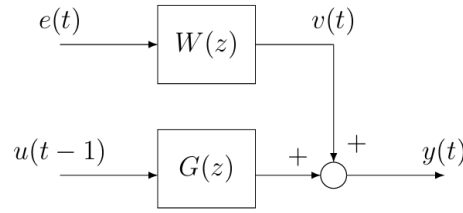
e non:

$$y(t) = \sum_{i=1}^{\infty} g(t-i)u(i)$$

Identificare un modello in forma *parametrica* consiste nell'identificazione dei coefficienti del numeratore e del denominatore della f.d.t.



## 2.1 Box-Jenkins



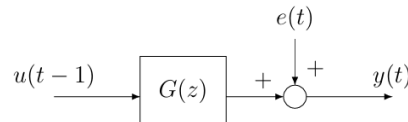
Il modello di Box-Jenkins rispetta le seguenti ipotesi:

1.  $W(z)$  deve essere asintoticamente stabile (quindi deve avere tutte le radici del polinomio al denominatore all'interno del cerchio di raggio unitario). I polinomi al numeratore e denominatore di questa f.d.t. devono essere monici (coefficiente di grado massimo uguale ad 1), co-primi (ovvero primi tra loro, il loro massimo comune divisore deve essere una costante non nulla), inoltre la funzione deve essere propria (num e den di ugual grado).
2.  $G(z)$  deve essere asintoticamente stabile, strettamente propria (grado del polinomio al denominatore deve essere maggiore di quello del numeratore).

Le equazioni del modello sono:

$$\begin{cases} y(t) = G(z) \cdot u(t-1) + W(z) \cdot e(t) \\ v(t) = W(z) \cdot e(t) \end{cases}$$

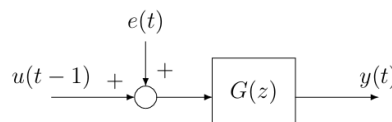
### 2.1.1 OE (Output Error)



Si assume una  $W(z)$  unitaria (uguale ad 1).

$$y(t) = G(z) \cdot u(t-1) + e(t)$$

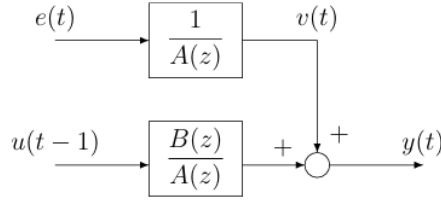
### 2.1.2 IE (Input Error)



Si ha come fonte di incertezza l'errore sulla misura di ingresso (modello meno comune).

$$y(t) = G(z) \cdot [u(t-1) + e(t)]$$

### 2.1.3 ARX (AutoRegressive with eXogenous variables)



Sono modelli a due gradi di libertà la cui uscita è costituita dalla somma di tre componenti: un contributo *autoregressivo* sull'uscita (i cui campioni sono calcolati come combinazione lineare dei valori precedenti mediando con dei pesi), uno *esogeno* (è una media mobile, perché calcolo la media pesata dei campioni) ed il contributo puro della variabile d'errore (rumore bianco con media nulla e varianza  $\lambda^2$ ).

$$y(t) = \underbrace{a_1 y(t-1) + \dots + a_{n_a} y(t-n_a)}_{AR} + \underbrace{b_1 u(t-1) + \dots + b_{n_b} u(t-n_b)}_X + e(t)$$

Da cui, per la proprietà di traslazione nel tempo della Z-trasformata:

$$\underbrace{1 - a_1 z^{-1} - \dots - a_{n_a} z^{-n_a}}_{A(z)} \cdot y(t) = \underbrace{b_1 + b_2 z^{-1} + \dots + b_{n_b} z^{-n_b+1}}_{B(z)} \cdot u(t-1) + e(t)$$

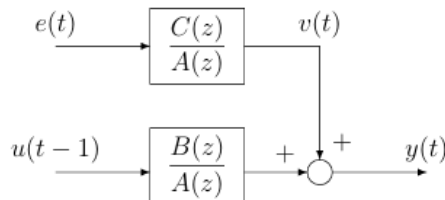
Quindi:

$$y(t) = \frac{B(z)}{A(z)} \cdot u(t-1) + \frac{1}{A(z)} \cdot e(t)$$

### 2.1.4 ARMAX

AutoRegressive Moving Average eXogenous model, hanno tre gradi di libertà, oltre ai contributi precedenti si aggiunge una media mobile per pesare l'errore.

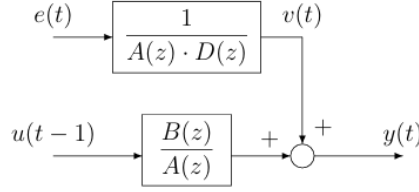
$$C(z) = 1 + c_1 z^{-1} + \dots + c_{n_c} z^{-n_c}$$



$$y(t) = \frac{B(z)}{A(z)} \cdot u(t-1) + \frac{C(z)}{A(z)} \cdot e(t)$$

### 2.1.5 ARARX

Anche detti ARXAR. Oltre al contributo autoregressivo sull'uscita né è presente un'ulteriore sulla variabile d'errore.



$$y(t) = \frac{B(z)}{A(z)} \cdot u(t-1) + \frac{1}{A(z) \cdot D(z)} \cdot e(t)$$

## 3 Criterio di Scelta

Devo capire qual'è il miglior modello.

Se ho una sequenza di dati osservata sperimentalmente, con N campioni:

$$y_{t_0}^{t_N} = \{y(t_0), y(t_1), \dots, y(t_N)\}$$

$$u_{t_0}^{t_N} = \{u(t_0), u(t_1), \dots, u(t_N)\}$$

La *famiglia di modelli* è rappresentata dall'insieme:

$$\mathbb{M} = \{\mathcal{M}(\theta) | \theta \in \Theta\}$$

con  $\mathcal{M}$  indichiamo il modello, funzione del suo vettore di parametri  $\theta$ , individuati nello spazio vettoriale  $\Theta$ .

**Esempio** Se utilizziamo la notazione: ARX(3,3); indichiamo un modello ARX avente 3 coefficienti da identificare per il polinomio  $A(z)$  e 3 per  $B(z)$ , da cui si evince che i polinomi saranno di secondo grado e  $\Theta = \mathbb{R}^6$ .

### 3.1 Estrazione del predittore

#### 3.1.1 Predittore ad un passo

È indicato dalle seguenti forme

$$\hat{y}(t) = \hat{y}(t|t-1) = \hat{y}(t|t-1, \theta)$$

e presenta un errore di predizione:

$$\epsilon(t) = y(t) - \hat{y}(t)$$

Per una buona identificazione, vogliamo minimizziamo questo errore; calcolando un indice di qualità del modello:

$$J(\theta) = \frac{1}{N} \sum_{i=1}^N |y(t) - \hat{y}(t)|^2$$

Partendo dal modello di Box-Jenkins:

$$y(t) = G(z) \cdot u(t-1) + W(z) \cdot e(t)$$

moltiplicando ambo i membri per l'inverso di W:

$$W^{-1}(z) \cdot y(t) = W^{-1}(z) \cdot G(z) \cdot u(t-1) + W(z) \cdot e(t)$$

e sommando ad ambo i membri  $y(t)$ :

$$y(t) + W^{-1}(z) \cdot y(t) = W^{-1}(z) \cdot G(z) \cdot u(t-1) + W(z) \cdot e(t) + y(t)$$

si ottiene:

$$y(t) = W^{-1}(z) \cdot G(z) \cdot u(t-1) + W(z) \cdot e(t) + [1 - W^{-1}(z)] \cdot y(t)$$

Dove:

$$W(z) = \frac{z^m + \alpha_1 z^{m-1} + \dots}{z^m + \beta_1 z^{m-1} + \dots}$$

e

$$W^{-1}(z) = \frac{z^m + \beta_1 z^{m-1} + \dots}{z^m + \alpha_1 z^{m-1} + \dots}$$

Eseguendo la divisione (lunga) tra polinomi:

$$\begin{array}{r|l} z^m + \beta_1 z^{m-1} + \beta_2 z^{m-2} + \dots & z^m + \alpha_1 z^{m-1} + \alpha_2 z^{m-2} + \dots \\ z^m + \alpha_1 z^{m-1} + \alpha_2 z^{m-2} + \dots & 1 + (\beta_1 - \alpha_1) z^{-1} + \gamma z^{-2} + \dots \\ \hline // + (\beta_1 - \alpha_1) z^{m-1} + (\beta_2 - \alpha_2) z^{m-2} + \dots & \\ // + (\beta_1 - \alpha_1) z^{m-1} + \alpha_1 (\beta_1 - \alpha_1) z^{m-2} + \dots & \\ \hline & // + \gamma z^{m-2} + \dots \end{array}$$

possiamo scrivere il polinomio  $W^{-1}(z)$  in maniera diversa, riuscendo ad ottenere nella formula scritta in precedenza il termine di ritardo moltiplicato per  $y(t)$ , il che va ad indicare che nella parte destra dell'equazione stiamo considerando i valori passati di  $y(t)$ .

Adesso per ottenere la formula finale del predittore devo condensare la funzione  $e(t)$  in unico valore (scelgo di considerare il valore atteso, che nel caso di rumore bianco è nullo):

$$\hat{y}(t) = W^{-1}(z) \cdot G(z) \cdot u(t-1) + [1 - W^{-1}(z)] \cdot y(t)$$

### 3.1.2 ARX

Sapendo che:

$$G(z) = \frac{B(z)}{A(z)}, \quad W(z) = \frac{1}{A(z)}$$

sostituendole nell'equazione del predittore ad un passo, ottengo:

$$\hat{y}(t) = B(z) \cdot u(t-1) + [1 - A(z)] \cdot y(t)$$

### 3.1.3 ARMAX

Sapendo che:

$$G(z) = \frac{B(z)}{A(z)}, \quad W(z) = \frac{1}{A(z)}$$

sostituendole nell'equazione del predittore ad un passo, ottengo:

$$\hat{y}(t) = \frac{B(z)}{C(z)} \cdot u(t-1) + \left[1 - \frac{A(z)}{C(z)}\right] \cdot y(t)$$

## 4 Validazione

Se  $\Theta = \mathbb{R}^n$ ,  $n$  è il numero complessivo di parametri del modello.

Ci si chiede quale legame sussiste tra  $N$  ed  $n$ . Si intuisce che  $N \gg n$ .

### 4.1 Analisi Asintotica

Se un sistema è affetto da incertezza avremo una non ripetibilità dell'acquisizione dei dati sperimentali (quindi i dati osservati in due esperimenti, condotti nelle stesse condizioni, non saranno mai esattamente gli stessi); quindi nell'identificazione non otteniamo un predittore caratterizzato sempre gli stessi parametri.

Coma abbiamo già visto, sommando tutti i quadrati degli errori di predizione si ottiene un indice di qualità:  $J_N(\theta)$ , che misura lo scostamento quadratico della predizione dall'osservazione sperimentale. Il miglior modello è quello che minimizza questo indice.

In una situazione ideale (fenomeni stocastici caratterizzati), si può definire un indice che rappresenta la descrizione completa teorica di un dato fenomeno stocastico:

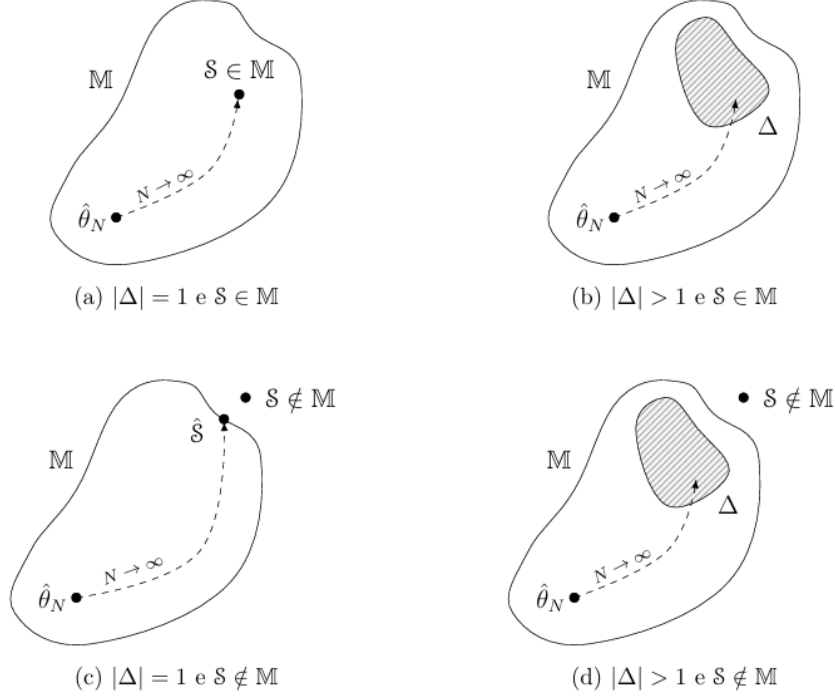
$$\bar{J}(\theta) = \{\epsilon^2(t)\}$$

L'indice  $J_N(\theta)$  però è ottenuto con  $N$  dati, quindi abbiamo una *stima campionaria*. All'aumentare di  $N$  la caratterizzazione del fenomeno stocastico migliora. Se  $N \rightarrow \infty$ , allora  $J_N(\theta) \rightarrow \bar{J}(\theta)$ .

L'insieme dei punti di minimo di  $J(\theta)$  nello spazio dei parametri è:

$$\Delta = \{\bar{\theta} \in \Theta | J(\bar{\theta}) \leq J(\theta), \forall \theta \in \Theta\}$$

Se  $\mathcal{S}$  è il **MGD** (*Meccanismo di Generazione dei Dati*), ovvero il modello da identificare; ad un aumento della quantità dei dati si tende ad un modello migliore. Si distinguono però 4 casi, a seconda della cardinalità.



- $|\Delta| = 1; \mathcal{S} \in \mathbb{M}$ . Ho un solo punto di minimo ed il modello identificato appartiene nella famiglia de modelli scelta. Allora  $\hat{\theta}_N$  converge verso  $\mathcal{S}$  nell'insieme  $\mathbb{M}$ .
- $|\Delta| = 1; \mathcal{S} \notin \mathbb{M}$ . Allora  $\hat{\theta}_N$  converge verso un modello  $\hat{\mathcal{S}}$ , che è la migliore stima del MGD all'interno della famiglia di modelli scelta e giacerebbe lungo la frontiera dell'insieme  $\mathbb{M}$ .
- $|\Delta| > 1; \mathcal{S} \in \mathbb{M}$ . Può accadere che  $\hat{\theta}_N$  converga verso un elemento di  $\Delta$  oppure che esso oscilli fra i vari punti di esso senza convergere.
- $|\Delta| > 1; \mathcal{S} \notin \mathbb{M}$ . Può accadere che  $\hat{\theta}_N$  converga verso un elemento di  $\Delta$  oppure che esso oscilli fra i vari punti di esso senza convergere.

**dim. Caso 1** Sia  $\mathcal{S} \in \mathbb{M}$  e siano le uscite del meccanismo di generazione dei dati:  $y(t|\theta^0)$ , con  $\theta^0$  vettore dei parametri veri. L'errore di predizione è:

$$\epsilon(t) = y(t) - \hat{y}(t|\theta)$$

Aggiungendo e sottraendo una quantità:

$$\epsilon(t) = y(t) - \hat{y}(t|\theta) + \hat{y}(t|\theta^0) - \hat{y}(t|\theta^0)$$

questo dovrebbe essere pari a zero; ma non lo è, perché la predizione è ottenuta sbarazzandosi dei valori casuali e sostituendoli con il valor medio.

In particolare quindi distinguiamo<sup>1</sup> :  $e(t) = y(t) - \hat{y}(t|\theta^0)$   
 Considerando un valor medio nullo:

$$Var\{\epsilon(t)\} = Var\{e(t)\} + Var\{\hat{y}(t|\theta^0) - \hat{y}(t|\theta)\} = \lambda^2 + Var\{\hat{y}(t|\theta^0) - \hat{y}(t|\theta)\}$$

Sapendo che  $e(t)$  è correlato solo a se stesso,  $\hat{y}(t|\theta^0) - \hat{y}(t|\theta)$  è correlato ai valori passati.  $\theta^0$  è un punto di minimo, quindi appartiene a  $\Delta$ . Ma non è detto però che  $\theta^0$  sia l'unico punto di minimo, inoltre ottengo  $J = 0$  solo se stimo un modello perfetto in assenza di incertezza.

## 4.2 Analisi di Bianchezza

In una situazione di identificazione ideale, tutto ciò che sta da dedurre dai dati sperimentali è stato spiegato dal modello identificato; perciò il residuo della predizione deve essere il più possibile un segnale casuale, dal momento che non deve contenere nessun *offset*<sup>2</sup>.

Il *rumore bianco* è il segnale casuale per eccellenza. Per valutare quanto siano casuali gli errori di predizione residui si effettua un'**analisi di bianchezza**, ovvero un test di somiglianza degli errori di predizione residui al rumore bianco. Per il confronto è considerata la *funzione di autocorrelazione*<sup>3</sup>. Quindi una stima campionaria della funzione di autocorrelazione dei residui della predizione deve essere quanto più possibile ad un impulso di Dirac (non sarà mai perfettamente coincidente perché non disponiamo di infiniti campioni). Un altro confronto è basato sullo *spettro*, deve risultare costante (caratteristica del rumore bianco).

## 5 Ottimizzazione ai Minimi Quadrati

Metodo di minimizzazione della funzione obiettivo molto comune, usato in tutti i casi in cui i parametri del modello hanno una dipendenza lineare dall'uscita (come gli ARX).

Supponendo di avere un'uscita sconosciuta  $y(t)$ , con  $N$  osservazioni sperimentali note di  $y$ ; si cerca una funzione che l'approssimi:

$$\hat{y}(t) = \theta_1 x_1(t) + \theta_2 x_2(t) + \dots + \theta_n x_n(t) = \theta^T \cdot \varphi(t) = \theta \cdot \varphi^T(t)$$

Definiamo il *vettore delle uscite* ( $N \times 1$ ):

$$\mathbb{Y} = \begin{bmatrix} y(t_1) \\ y(t_2) \\ \vdots \\ y(t_N) \end{bmatrix}$$

---

<sup>1</sup>l'incertezza per il quale un modello non sarà mai perfetto.

<sup>2</sup>esiste un'operazione di rimozione degli offset che prende il nome di *detrend*

<sup>3</sup>il rumore bianco ha una funzione di autocorrelazione che è un impulso di Dirac (ciascun campione è correlato solamente a se stesso)

e rispettivamente abbiamo il *vettore dei regressori* (noto) e quello *dei parametri* [entrambi  $(n \times 1)$ ]:

$$\varphi(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix}; \quad \theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$$

É definita inoltre la *matrice dei regressori*  $(n \times N)$ :

$$\phi = \begin{bmatrix} \varphi^T(t_1) \\ \varphi^T(t_2) \\ \vdots \\ \varphi^T(t_N) \end{bmatrix}$$

Per risolvere il problema dei minimi quadrati:

$$\begin{aligned} J(\theta) &= \frac{1}{N} \sum_{i=1}^N \epsilon^2(t) = \frac{1}{N} (Y - \phi\theta)^T \cdot (Y - \phi\theta) = \\ &= \frac{1}{N} [(Y^T - \theta^T \phi^T)(Y - \phi\theta)] = \\ &= \frac{1}{N} [Y^T Y - \theta^T \phi^T \phi \theta - \theta^T \phi^T Y - Y^T \theta \phi] = \\ &= \frac{1}{N} [Y^T Y - \theta^T \phi^T \phi \theta - 2\theta^T \phi^T Y] \end{aligned}$$

Per minimizzare l'indice, bisogna imporre l'annullamento del suo gradiente:

$$\nabla J = \left[ \frac{\partial J}{\partial \theta_1} \cdots \frac{\partial J}{\partial \theta_n} \right] = 2\phi^T \phi \theta - 2\phi^T Y = 0$$

Da cui si ricava:

$$\phi^T \phi \theta = \phi^T Y$$

ottenendo così la soluzione:

$$\theta_{MQ} = (\phi^T \phi)^{-1} \phi^T Y = \arg \min_{\theta \in \Theta} J(\theta) = \hat{\theta}_N$$

Se la matrice  $\phi^T \phi$  non è invertibile allora non esiste la soluzione al problema. Se esiste la soluzione si ottiene in un passo (perché è di tipo algebrico).

Se dobbiamo applicare l'algoritmo dei minimi quadrati per l'identificazione di un modello ARX avente  $n = n_a + n_b$  parametri, a partire da  $N$  osservazioni sperimentali. La soluzione al problema è:

$$\theta_{MQ} = \left[ \sum_{i=1}^N \varphi(t_i) \varphi^T(t_i) \right]^{-1} \left[ \sum_{i=1}^N y(t_i) \varphi(t_i) \right]$$



Il vettore delle uscite è il seguente:

$$Y = \begin{bmatrix} y(t_1) \\ y(t_2) \\ \vdots \\ y(t_N) \end{bmatrix}$$

I vettori dei regressori e dei parametri saranno invece:

$$\varphi(t) = \begin{bmatrix} y(t_i - 1) \\ \vdots \\ y(t_i - n_a) \\ u(t_i - 1) \\ \vdots \\ u(t_i - n_b) \end{bmatrix}; \quad \theta = \begin{bmatrix} a_1 \\ \vdots \\ a_{n_a} \\ b_1 \\ \vdots \\ b_{n_b} \end{bmatrix}$$

Si avrà:

$$\hat{Y}(t_i) = \theta^T \varphi(t_i)$$

La matrice dei regressori sarà strutturata nel seguente modo:

$$\phi = \begin{bmatrix} y(t_i) & 0 & 0 & 0 & \cdots & u(t_i) & 0 & 0 & 0 & \cdots \\ y(t_2) & y(t_2) & 0 & 0 & \cdots & u(t_2) & u(t_1) & 0 & 0 & \cdots \\ \vdots & \vdots & \ddots & \cdots & \vdots & \vdots & \vdots & \ddots & \cdots & \vdots \end{bmatrix}$$

Note queste matrici, posso calcolare direttamente  $\theta_{MQ}$ .

## 5.1 Analisi Probabilistica

La linearità nei parametri dei modelli ARX è una caratteristica favorevole per svolgere l'analisi asintotica e studiare il comportamento del modello al crescere del numero N di campioni.

Considerando un caso in cui il meccanismo di generazione dei dati è un ARX:

$$S = m(\theta^0)$$

quindi, l'equazione del MGD è:

$$y(t) = \varphi^T(t) \cdot \theta^0 + e(t)$$

dove  $e(t)$ ,  $\varphi(t)$  rappresentano rispettivamente l'incertezza ed il vettore dei regressori.

L'equazione del predittore è:

$$\hat{y}(t) = \varphi^T(t) \cdot \theta$$

quindi l'errore di predizione vale:

$$\epsilon(t) = \varphi^T(t) \cdot (\theta^0 - \theta) + e(t)$$

Il termine  $(\theta^0 - \theta)$  prende il nome di *errore di stima parametrica*.

L'indice da minimizzare è:

$$\begin{aligned}\bar{J}(\theta) &= E\{e^2(t)\} = E\{(\theta^0 - \theta)^T \varphi(t) \varphi^T(t) (\theta^0 - \theta) + e^2(t) + 2\varphi^T(t) (\theta^0 - \theta) e(t)\} = \\ &= (\theta^0 - \theta)^T E\{\varphi(t) \varphi^T(t)\} (\theta^0 - \theta) + E\{e^2(t)\} + E\{2\varphi^T(t) (\theta^0 - \theta) e(t)\}\end{aligned}$$

Se  $e(t)$  è il rumore bianco, è correlato solamente a se stesso, allora:

$$\bar{J}(\theta) = (\theta^0 - \theta)^T E\{\varphi(t) \varphi^T(t)\} (\theta^0 - \theta) + E\{e^2(t)\} + E\{2\varphi^T(t)\} (\theta^0 - \theta) E\{e(t)\}$$

inoltre  $E\{e^2(t)\} = \lambda^2$  e  $E\{e(t)\} = 0$ , quindi:

$$\bar{J}(\theta) = (\theta^0 - \theta)^T E\{\varphi(t) \varphi^T(t)\} (\theta^0 - \theta) + \lambda^2$$

Otengo quindi una forma quadratica, legata fortemente alla matrice dei minimi quadrati:

$$S(N) = \sum_{t=1}^N \varphi(t) \varphi^T(t)$$

la quale per definizione è semidefinita positiva. Affinché si abbia una soluzione però il suo determinante deve essere diverso da zero (la matrice deve essere invertibile). Una buona identificazione si ha quando si può manipolare  $S(N)$  per renderla "il più possibile" non singolare.

### 5.1.1 Persistente eccitazione

Per valutare la singolarità della matrice  $S(N)$  consideriamo un modello ARX(1,1).

$$y(t) = a_1 y(t-1) + b_1 u(t-1)$$

$$\varphi(t) = [y(t-1) \quad u(t-1)]^T$$

$$\varphi(t) \varphi^T(t) = \begin{bmatrix} y(t-1) \\ u(t-1) \end{bmatrix} \cdot [y(t-1) \quad u(t-1)] = \begin{bmatrix} y^2(t-1) & y(t-1)u(t-1) \\ u(t-1)y(t-1) & u^2(t-1) \end{bmatrix}$$

$$S(N) = \sum_{t=1}^N \varphi(t) \varphi^T(t) = \begin{bmatrix} \sum_{t=1}^N y^2(t-1) & \sum_{t=1}^N y(t-1)u(t-1) \\ \sum_{t=1}^N u(t-1)y(t-1) & \sum_{t=1}^N u^2(t-1) \end{bmatrix}$$

La matrice dei minimi quadrati tende a divergere al crescere di  $N$ , per questo la si *normalizza*:

$$R(N) = \frac{S(N)}{N} = \begin{bmatrix} \frac{1}{N} \sum_{t=1}^N y^2(t-1) & \frac{1}{N} \sum_{t=1}^N y(t-1)u(t-1) \\ \frac{1}{N} \sum_{t=1}^N u(t-1)y(t-1) & \frac{1}{N} \sum_{t=1}^N u^2(t-1) \end{bmatrix}$$

Poiché i segnali  $u$  ed  $y$  sono stazionari, i termini della matrice sono le stime campionarie delle rispettive funzioni di correlazione per  $\tau = 0$ , che indicheremo con  $\gamma$ :

$$R(N) = \begin{bmatrix} \gamma_{yy}(0) & \gamma_{yu}(0) \\ \gamma_{uy}(0) & \gamma_{uu}(0) \end{bmatrix}$$

Considerando invece un modello  $ARX(n_a, n_b)$ :

$$R(N) = \begin{bmatrix} \Gamma_{yy}(0) & \Gamma_{yu}(0) \\ \Gamma_{uy}(0) & \Gamma_{uu}(0) \end{bmatrix}$$

dove:

- $\Gamma_{yy}$  è il blocco relativo all'autocorrelazione del segnale in uscita. Non è manipolabile e la sua dimensione è  $(n_a \times n_a)$ .
- $\Gamma_{yu} = \Gamma_{yu}$  sono i blocchi relativi alla cross-correlazione fra il segnale in uscita e quello in ingresso. Non sono manipolabili e le loro dimensioni sono  $(n_a \times n_b)$  e  $(n_b \times n_a)$ .
- $\Gamma_{uu}$  è il blocco relativo all'autocorrelazione del segnale di ingresso. È manipolabile e la sua dimensione è  $(n_b \times n_b)$ .

Avendo una matrice partizionata nel modo seguente:

$$U = \begin{bmatrix} F & K \\ K^T & G \end{bmatrix}$$

per il **lemma di Schur**, condizione necessaria e sufficiente affinché  $U$  sia definita positiva:

$$\begin{cases} G > 0 \\ F - K^T G^{-1} K > 0 \end{cases}$$

La struttura della matrice  $R(N)$  è simile a quella della matrice  $U$ . Il blocco  $G$  corrisponde a  $\Gamma_{uu}$ , e quindi lo posso manipolare, mentre i blocchi  $K = \Gamma_{yu}$ ,  $F = \Gamma_{yy}$  no, in quanto dipendenti dall'uscita.

Nella pratica si cerca di scegliere una matrice  $(n_b \times n_b)$  definita positiva come di seguito:

$$\Gamma_{uu} = \begin{bmatrix} \gamma_{uu}(0) & \gamma_{uu}(1) & \gamma_{uu}(2) & \cdots & \gamma_{uu}(n_b - 1) \\ \gamma_{uu}(1) & \gamma_{uu}(0) & \gamma_{uu}(1) & \cdots & \gamma_{uu}(n_b - 2) \\ \vdots & \vdots & \ddots & \cdots & \vdots \\ \gamma_{uu}(n_b - 1) & \gamma_{uu}(n_b - 2) & \gamma_{uu}(n_b - 3) & \cdots & \gamma_{uu}(0) \end{bmatrix}$$

La struttura di questa matrice (elementi coincidenti lungo le varie diagonal) la rende una matrice di **Toeplitz**.

Sulla diagonale principale si trova la varianza del segnale di ingresso, sulle diagonal al di sopra ed al di sotto di quella principale si trovano le autocorrelazioni in un passo (e così via fino ad arrivare alle funzioni di autocorrelazione in  $n_b - 1$  passi).

La scelta di una matrice  $\Gamma_{uu}$  definita positiva è una condizione necessaria ma non sufficiente (almeno sono convinto di non essere io a sbagliare l'ingresso, ma non posso garantire di avere una situazione identificabile).

Un segnale è detto **persistentemente eccitante di ordine  $N$**  se la sua matrice  $\Gamma_{uu}$  è non singolare (quindi è invertibile).

Se un segnale è persistentemente eccitante di ordine  $N$ , lo è anche per qualsiasi altro ordine minori di  $N$ .

Alcuni esempi di segnali di questo tipo:

- **rumore bianco**: può essere di ordine qualsiasi;
- **rumore colorato**: può essere di ordine qualsiasi;

- **PRBS** (*Pseudo Random Binary Signals*): è di ordine  $\ell$ , con  $\ell$  numero di registri a scorrimento impiegati per generarlo.
- **somma di sinusoidi equispaziate in frequenza**: è di ordine  $m$ , con  $m$  numero di sinusoidi (il rumore bianco rientra in questa categoria, con  $m \rightarrow \infty$ ).

## 5.2 Identificabilità

Studia la possibilità di trovare effettivamente un modello. È suddivisibile in due tipologie:

- *strutturale* (errore nella scelta del modello che causa una scarsa identificabilità);
- *sperimentale* (errore nella scelta degli ingressi).

## 5.3 Ordine Ottimo di un Modello

Uno dei problemi centrali dell'identificazione è la scelta della complessità dei modelli. Un modello risulta efficace quando può spiegare oltre ai dati disponibili nell'identificazione anche altri dati non considerati nel training.

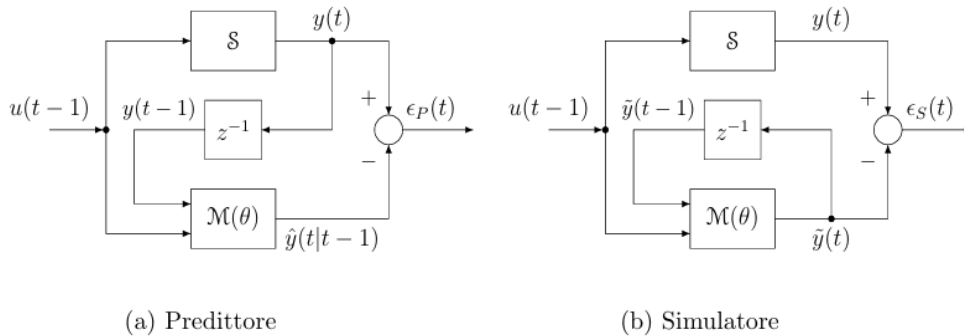
Utilizzando il criterio:

$$\mathcal{J}(\theta) = \frac{1}{N} \sum_{t=1}^N \epsilon^2(t)$$

con:

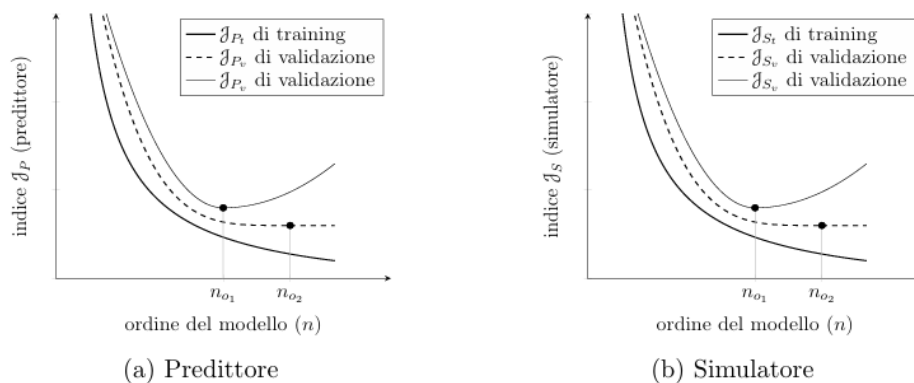
- $n$ : numero di parametri
- $\theta$ : parametri
- $\epsilon = \begin{cases} \epsilon_p(t) = y(t) - \hat{y}(t|t-1) : \text{errore di predizione} \\ \epsilon_s(t) = y(t) - \tilde{y}(t) : \text{errore di simulazione} \end{cases}$

L'utilizzo di un simulatore in genere permette di ottenere risultati migliori (ma costruirne uno è più complicato).



Se il numero di dati da identificare è elevato, possiamo usare una parte di essi per effettuare il training di un modello e la restante parte per la validazione dello stesso. Al crescere dell'ordine del modello l'indice diminuisce. L'ordine ottimo si ottiene però in corrispondenza di un punto di minimo dell'indice. Perché per ordini troppo elevati l'andamento di  $\mathcal{J}$  per i dati di validazione tende a peggiorare, perché tende a seguire troppo i dati di training (anche il rumore presente), andando in una situazione di *overfitting*.

Può capitare che l'andamento dell'indice diventi costante dopo un certo ordine, in tal caso si hanno infiniti minimi. È semplice dedurre che verrà scelto il numero minimo di parametri da introdurre che portino comunque al modello ottimo.



Se non si dispone di un numero sufficiente di dati per effettuare training, l'ordine ottimo deve essere stimato con approcci differenti. Si ricorre a, l'utilizzo di indici di ottimalità (i primi due tendono a sovrastimare l'ordine corretto del modello, il terzo da una stima leggermente più precisa).

### 1. Final Prediction Error:

$$FPE = \frac{N+n}{N-n} \mathcal{J}(\hat{\theta}_N)$$

### 2. Akaike Information Criterion:

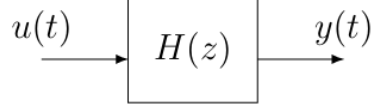
$$AIC = \frac{2n}{N} + \ln(\mathcal{J}(\hat{\theta}_N))$$

### 3. Minimum Description Length:

$$MDL = \ln(N) + \frac{n}{N} + \ln(\mathcal{J}(\hat{\theta}_N))$$

## 5.4 Analisi Spettrale

Effettuando un passaggio dal dominio del tempo a quello della frequenza (tramite le trasformate di Fourier), si possono evidenziare alcune peculiarità.



Il *teorema del filtraggio* dei segnali afferma che, dato un filtro lineare asintoticamente stabile, se  $u(t)$  è stazionario (come il rumore bianco) e  $\Phi_{uu}(\omega)$  è il suo spettro, si dimostra che lo spettro dell'uscita sarà:

$$\Phi_{yy}(\omega) = |H(e^{j\omega})|^2 \Phi_{uu}(\omega)$$

il filtro, quindi, ridistribuisce la potenza del segnale nel dominio della frequenza, alterando la forma dello spettro.

Il *teorema di Wiener* afferma che, la trasformata di Fourier della funzione di autocorrelazione di un segnale stazionario coincide con la sua densità spettrale:

$$\mathcal{F}\{\gamma_{uu}(\tau)\} = \Phi_{uu}(\omega) \iff \mathcal{F}^{-1}\{\Phi_{uu}(\omega)\} = \gamma_{uu}(\tau)$$

Supponendo di avere un MGD (Box-Jenkins):

$$S : y(t) = G_0(z)u(t-1) + W_0(z)e(t)$$

ed un modello del predittore:

$$m : \hat{y}(t) = (1 - W^{-1}(z))y(t) + W^{-1}(z)G(z)u(t-1)$$

L'errore di predizione è:

$$\begin{aligned} \epsilon(t) &= y(t) - \hat{y}(t) = W^{-1}y(t) - W^{-1}Gu(t-1) = \\ &= W^{-1}[y(t) - Gu(t-1)] = \\ &= W^{-1}[G_0u(t-1) + W_0e(t) - Gu(t-1)] \end{aligned}$$

Definendo  $\Delta G = (G_0 - G)$ , otteniamo:

$$\epsilon(t) = W^{-1}[\Delta Gu(t-1) + W_0e(t)]$$

vale quindi la seguente catena di uguaglianze:

$$\bar{\mathcal{J}}(\theta) = E\{\epsilon^2(t)\} = \gamma_{\epsilon\epsilon}(0) = \mathcal{F}\{\Phi_{\epsilon\epsilon}(\omega)\}|_{\tau=0} = \frac{1}{2\pi} \int_{-\frac{\pi}{\tau}}^{\frac{\pi}{\tau}} \Phi_{\epsilon\epsilon}(\omega) e^{2\pi j t} d\omega$$

valutando in  $t = 0$ :

$$\bar{\mathcal{J}}(\theta) = \frac{1}{2\pi} \int_{-\frac{\pi}{\tau}}^{\frac{\pi}{\tau}} \Phi_{\epsilon\epsilon}(\omega) d\omega = \frac{1}{2\pi} \int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} |W^{-1}(e^{j\omega}, \theta)|^2 [|\Delta G(e^{j\omega}, \theta)|^2 \Phi_{uu}(\omega) + |W_0(e^{j\omega})|^2 \lambda_e^2] d\omega$$

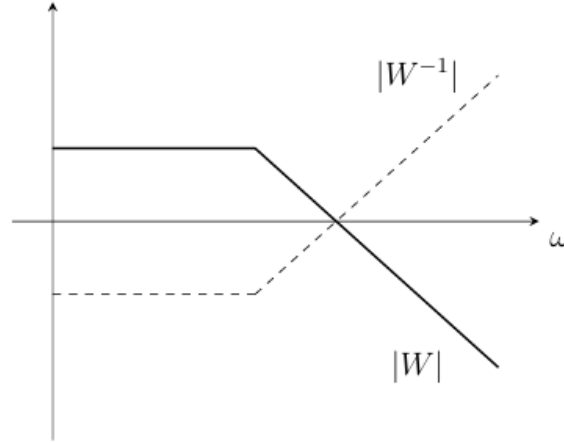
dove  $\lambda_e$  è lo spettro del rumore bianco. Supponendo che la potenza dell'incertezza sia trascurabile, il calcolo si conduce a:

$$\bar{J}(\theta) = \frac{1}{2\pi} \int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} |W^{-1}(e^{j\omega}, \theta)|^2 |\Delta G(e^{j\omega}, \theta)|^2 \Phi_{uu}(\omega) d\omega$$

L'indice quindi dipende da una funzione *peso*:

$$g(\omega, \theta) = \frac{\Phi_{uu}(\omega)}{|W(e^{j\omega}, \theta)|^2}$$

Maggiore è il peso di una componente, tanto più significativa sarà la riduzione dell'indice da parte dell'algoritmo di ottimizzazione (nella ripartizione in frequenza saranno minimizzate maggiormente le frequenze aventi la funzione peso è maggiore). Si ottiene quindi un  $W$  di tipo *passa-basso* (avente quindi più poli che zeri).



Gli errori alle bassi frequenze sono maggiormente presenti, perché hanno una tendenza minore ad essere attenuati. La tecnica dei minimi quadrati allora identifica meglio il comportamento alle alte frequenze, per questo motivo la funzione peso non deve privilegiare le alte frequenze ma essere il più possibile "piatta".

#### 5.4.1 Pre-filtraggio

Scegliendo  $W$  in modo appropriato per colorare lo spettro, è possibile appiattire la funzione. Questo, però, comporta due identificazioni. Per evitare ciò è possibile filtrare ingresso ed uscita con lo stesso filtro.

Supponendo di avere un filtro  $L(z)$  asintoticamente stabile, posso filtrare l'ingresso ottenendo:

$$u_L(t) = L(z)u(t)$$

e l'uscita allora sarà:

$$y_L(t) = L(z)y(t)$$

L'errore di predizione è:

$$\epsilon(t) = W^{-1}[y(t) - G(z)u(t-1)]$$

sostituendo ingresso ed uscita filtrati ottengo:

$$\epsilon_L(t) = W^{-1}[y_L(t) - G(z)u_L(t-1)] = W^{-1}L(z)[y(t) - G(z)u(t-1)] = L(z)\epsilon(t)$$

Quindi, applicare un filtro ad ingresso ed all'uscita (per l'ipotesi di linearità) è equivalente ad applicare il filtro all'errore di predizione. La funzione peso diventa:

$$g(\theta, \omega) = \frac{|L(e^{j\omega})|^2}{|W(e^{j\omega})|^2} \Phi_{uu}(\omega)$$

Questa tecnica viene utilizzata per evitare la necessità di reiterare il processo di identificazione. È un'operazione che deve essere eseguita prima di effettuare l'ottimizzazione dei parametri. Porta dei benefici se gli errori in frequenza non sono bilanciati (nel caso in cui siano ben distribuiti in frequenza essa non comporta alcun miglioramento). È una tecnica usata spesso perché conveniente a livello di costi.

## 5.5 Operazioni di Pre-processing sui Dati

Per migliorare i risultati dell'identificazione è possibile effettuare operazioni preventive sui dati, quali:

1. **Downsampling:** Se il tempo di campionamento risulta troppo piccolo i poli del sistema tendono a concentrarsi attorno al punto  $z = -1$  e l'algoritmo dovrebbe scovare delle differenze lievi (non sembrano esserci differenze significative tra un istante ed il successivo, dell'uscita di un segnale). Quindi si opera un *sottocampionamento*, introducendo una frequenza di campionamento fittizia adeguatamente minore di quella originaria.
2. **Antialiasing:** Se non si sceglie opportunamente la frequenza di campionamento<sup>4</sup>, si presentano degli effetti indesiderati di *aliasing*, per cui le code spettrali del segnale si sovrappongono e rendono il segnale difficile da ricostruire correttamente. Dei filtri vengono quindi usati per garantire la corretta ricostruzione dei segnali (tagliando lo spettro di un segnale in accordo con la condizione imposta dal teorema di campionamento).
3. **Peak Shaving:** Se un segnale presenta dei picchi (*spike*) indesiderati, dovuti ad anomalie occasionali (in cui l'errore di predizione diventa molto elevato) l'algoritmo di ottimizzazione si sforzerà a riprodurre anche questi fenomeni non voluti. Per risolvere il problema si può usare un filtro mediano per livellare i picchi, portandoli a valori vicini alla media campionaria relativa ad dati in input.

---

<sup>4</sup>almeno pari al doppio della frequenza massima del segnale da campionare (per il teorema di Shannon).



4. **Signal Slicing:** Se nei dati è presente una variazione di offset (discontinuità significative tra i valori assunti dai dati lungo tutti gli istanti di tempo). Questi comportamenti sono dovuti a cambi nelle condizioni operative, quindi conviene: individuare i punti di lavoro in corrispondenza dei quali si tagliano i diversi sottoinsieme di dati (*slicing*), rimuovere gli offset di I/O (*detrend*) e rimettere insieme le porzioni di dati separate (*merging*). Successivamente si potrà procedere con l'identificazione, ricordando di ripristinare poi gli offset precedentemente rimossi.
5. **Signal Scaling:** Nei sistemi MIMO l'indice è pari alla somma degli errori di predizione:

$$J = \frac{1}{n \cdot N} \left( \sum_{j=1}^n \sum_{i=1}^N |\epsilon_j(\tau_i)|^2 \right)$$

Quando le uscite sono eterogenee (non dimensionalmente confrontabili o con ordine di grandezza differente) bisogna normalizzare tutti i segnali (*scaling*). In questo modo gli indici integrali degli errori sono confrontabili ed i valori minimizzati sono consistenti.

## 5.6 Algoritmo dei minimi quadrati estesi

È usato per l'identificazione dei modelli ARMAX, che sono non lineari nei parametri. Il generico predittore ha equazione:

$$\hat{y}(k) = (1 - W^{-1})y(k) + W^{-1}Gu(k - 1)$$

l'errore di predizione ha espressione:

$$\epsilon(k) = y(k) - \hat{y}(k) = W^{-1}y(k) - W^{-1}Gu(k - 1) = \frac{A}{C}y(k) - \frac{B}{C}u(k - 1)$$

da cui:

$$C(z)\epsilon(k) = A(z)y(k) - B(z)u(k - 1)$$

sapendo che:

$$C(z) = 1 + c_1z^{-1} + \dots + c_{n_c}z^{-n_c}$$

$$A(z) = 1 - a_1z^{-1} - \dots - a_{n_a}z^{-n_a}$$

$$B(z) = b_1 + b_2z^{-1} + \dots + b_{n_b}z^{-n_b+1}$$

possiamo quindi riscrivere l'equazione come:

$$\begin{aligned} \epsilon(k) + c_1\epsilon(k - 1) + \dots + c_{n_c}\epsilon(k - n_c) &= y(k) - a_1y(k - 1) - \dots - \\ &- a_{n_a}y(k - n_a) - b_1u(k - 1) - \dots - b_{n_b}u(k - n_b) \end{aligned}$$

allora otteniamo il predittore:

$$\hat{y}(k) = a_1y(k - 1) + \dots + a_{n_a}y(k - n_a) + b_1u(k - 1) + \dots + b_{n_b}u(k - n_b) +$$

$$+\epsilon(k-1) + \dots + c_{n_c}\epsilon(k-n_c)$$

Verranno utilizzati come regressori quindi non solo gli ingressi e le uscite ma anche gli errori di predizione passati. Il predittore in questa forma sembra *apparentemente* lineare, questo perché per ottenere gli errori di predizione dovremmo prima utilizzare un modello ARX (quindi assumendo che i parametri  $c_i$  siano tutti nulli), dopodiché sarà possibile applicare l'algoritmo dei minimi quadrati estesi e stimare i parametri  $c_i$  sulla base degli errori ottenuti (iterando il processo poi si potranno affinare i valori di questi ultimi). Quando gli errori di predizione ottenuti in una data iterazione sono adeguatamente indistinguibili da quelli ottenuti nell'iterazione precedente allora anche i parametri del modello non possono più migliorare significativamente.

## 6 Identificazione di sistemi non lineari

Effettuiamo delle assunzioni, volte a semplificare la trattazione:

- tutti gli ingressi e le uscite hanno stesso ordine dinamico  $m$ ;
- l'ingresso  $u(k)$  non influenza immediatamente l'uscita  $y(k)$ , sistema strettamente proprio;
- non vi sono ulteriori ritardi tra ingresso ed uscita ( $n_k = 1$ );
- non viene prestata attenzione al modello dell'incertezza perché troppo complesso.

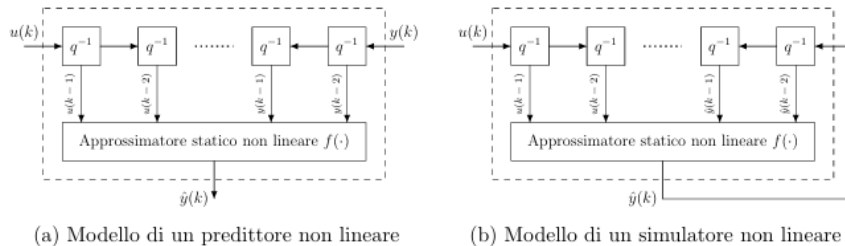
Il modello **NARX** (*Nonlinear AutoRegressive eXogenous model*) è rappresentato dalla seguente equazione (lo si ottiene come estensione del modello ARX in forma di predittore):

$$\hat{y}(t) = f(y(t-1), y(t-2), \dots, y(t-m), u(t-1), u(t-2), \dots, u(t-m))$$

dove  $f$  è una funzione *non* nota. Una struttura più generale è fornita dalla rappresentazione in variabili di stato:

$$\begin{cases} x(k+1) = h(x(k), u(k)) \\ y(k) = g(x(k)) \end{cases}$$

### 6.1 Modelli a dinamica esterna



Le funzioni lineari che caratterizzano questi modelli sono intrinsecamente statiche (prive di memoria), quindi, la dinamica è impostata esternamente al sistema non lineare passando la sua storia precedente.

Ricaviamo i modelli non lineari di ARX, ARMAX, OE:

- NARX:

$$\varphi(k) = [u(k-1) \quad \cdots \quad u(k-m) \quad y(k-1) \quad \cdots \quad y(k-m)]^T$$

- NARMAX:

$$\varphi(k) = [u(k-1) \quad \cdots \quad u(k-m) \quad y(k-1) \quad \cdots \quad y(k-m) \quad \epsilon(k-1) \quad \cdots \quad \epsilon(k-m)]^T$$

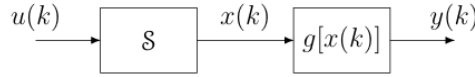
- NOE:

$$\varphi(k) = [u(k-1) \quad \cdots \quad u(k-m) \quad \hat{y}(k-1) \quad \cdots \quad \hat{y}(k-m)]^T$$

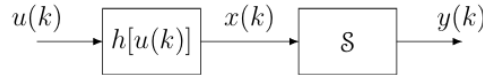
## 6.2 Modelli di Hammerstein e Wiener

Se un dato *sistema dinamico lineare* ( $\mathcal{S}$ ) presenta delle non-linearità statiche (come potrebbe essere ad esempio una saturazione o una dead-zone), possiamo modellare queste ultime con dei blocchi puramente algebrici.

Se le non-linearità si trovano a valle di  $\mathcal{S}$  si ottiene un modello di Wiener:



Se invece sono a monte si ha un modello di Hammerstein:



Se sono presenti sia a monte che a valle il modello si dice di Hammerstein-Wiener:



## 6.3 Scelta del segnale di Ingresso

Per via dei legami esistenti tra i campioni successivi dei segnali, gli ingressi dell'approssimatore statico  $f(\cdot)$  non saranno mai del tutto indipendenti.

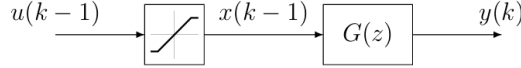
Infatti, solo  $u(k)$  è scelto dall'utente, mentre tutti gli altri ingressi ritardati dell'approssimatore e le uscite ne sono solo una conseguenza.

Minore è la frequenza del segnale in ingresso, tanto più la distribuzione dei dati ottenuti tenderà ad aderire al comportamento statico della non linearità.

Analogamente, minore è il tempo di campionamento<sup>5</sup>, maggiore sarà la relazione esistente tra  $u(k-1)$  ed  $u(k)$ .

Per poter coprire regioni ampie dello spazio d'ingresso si devono usare ingressi di eccitazione ad elevata variabilità (sia in ampiezza che in frequenza).

## Esempio



Considerando il sistema dinamico non-lineare in figura, costituito da un blocco lineare  $G$  il cui ingresso è sottoposto a saturazione<sup>6</sup>

Il blocco statico di saturazione è modellato con una funzione arcotangente:

$$f(u) = 0.06 \cdot \arctg(u(k-1))$$

Il blocco dinamico lineare ha la f.d.t. seguente:

$$G(z) = \frac{1}{z + 0.94}$$

Il sistema del primo ordine di Hammerstein (il cui ingresso è compreso nell'intervallo  $[-4,4]$ ; con costante di tempo 16 sec) è descritto dalla seguente equazione alle differenze non lineare:

$$y(k) = 0.06 \cdot \arctg(u(k-1)) + 0.94 \cdot y(k-1)$$

campionando con  $T_0 = 1\text{sec}$ ; se l'ingresso è costante, l'uscita diviene una costante. La coppia costituita dall'ingresso e la corrispondente uscita (costanti) è detta punto di equilibrio del sistema.

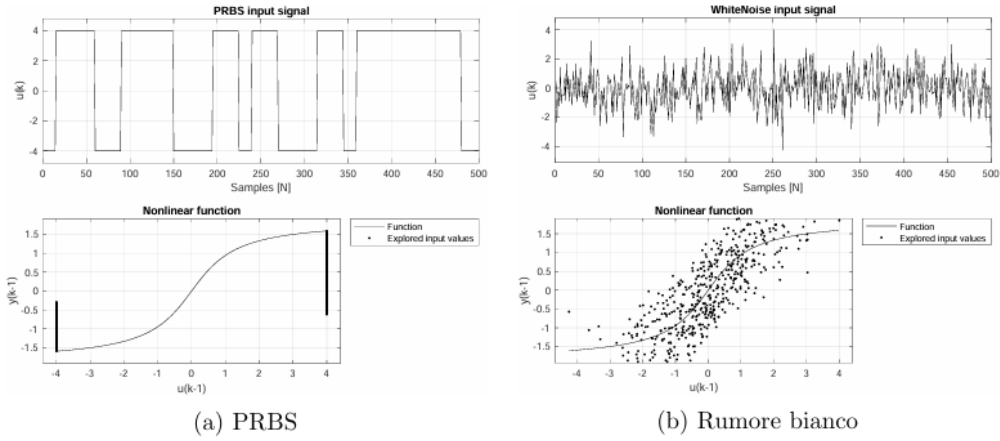
È importante che il segnale di ingresso esplori in maniera capillare il dominio della non-linearità.

Un segnale PRBS (PseudoRandom Binary Signal) può assumere solo due valori distinti ed è inadatto, quindi, per il sistema in esame (che ha un numero infinito di punti di equilibrio, introdotti a causa della saturazione).

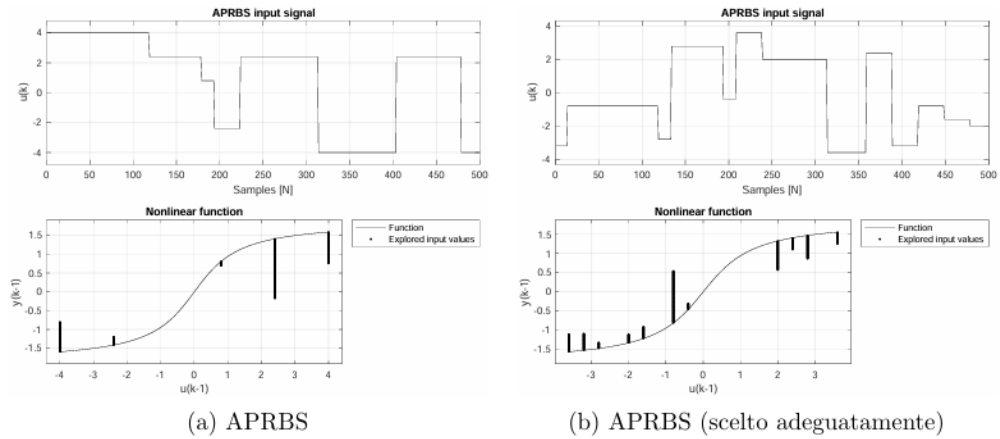
Per un'esplorazione adeguata del dominio è necessario un ingresso con ampiezza a più alta variabilità. Se scegliessimo un rumore bianco l'ampiezza del segnale varierebbe troppo rapidamente nel tempo, il che impedisce l'esplorazione dei punti di equilibrio.

<sup>5</sup>Al ridursi di  $T_s$  tutti i poli del sistema tendono a concentrarsi nell'intorno di  $z = 1$  (quindi sarà necessario un ordine elevato per rappresentare la dinamica del sistema; il che si traduce non solo in un aumento dei coefficienti ma in un aumento anche degli ingressi). Se invece si ha sotto-campionamento si rischia di non identificare in maniera sufficiente e completa la dinamica del sistema. Bisogna quindi scegliere con attenzione il passo di campionamento.

<sup>6</sup>Un tipo di non-linearità statica (può essere modellata tramite un blocco puramente algebrico); è impiegato quindi un modello di Hammerstein.



I segnali APRBS (Amplitude-modulated PRBS) sono in grado di produrre un ingresso che abbia una variabilità in ampiezza sufficiente che non vari troppo rapidamente. Ogni valore di ampiezza assunto rimane costante per un determinato *tempo di hold* (chiamato intervallo di minima permanenza).



Riducendo il valore del tempo di hold e modulando con una maggiore risoluzione riusciamo, a parità di numero di campioni, ad esplorare meglio il dominio di non-linearità (senza compromettere il raggiungimento dell'equilibrio).

## 6.4 Modelli in forma di variazione

L'approccio delle dinamiche esterne è basato sulla mappatura dei precedenti ingressi e delle precedenti uscite rispetto all'uscita reale.

L'uscita può essere espressa così:

$$y(k) = f(u(k-1), \Delta u(k-1), \dots, \Delta^{m-1}u(k-1), y(k-1), \Delta y(k-1), \dots, \Delta^{m-1}y(k-1))$$

Dove:

$$\Delta^m = (1 - q^{-1})^m$$

è detto operatore differenza.

Un sistema descritto in forma di variazione presenta i seguenti vantaggi:

- distribuzione di ingresso più uniforme;
- maggiore interpretabilità dei risultati (le differenze possono essere interpretate come le derivate del segnale).

Sfortunatamente, però, gli effetti del rumore diventano più grandi all'aumentare dell'ordine delle differenze (per questo motivo non sono quasi mai utilizzati per l'identificazione dei sistemi).

#### 6.4.1 Modelli affini nell'azione di controllo

Modelli con questa struttura si prestano bene a controllo:

$$y(k) = b_1 u(k-1) + f(u(k-2), \dots, u(k-m), y(k-1), \dots, y(k-m))$$

Se si ammette che il sistema possa essere descritto da questa equazione, l'azione di controllo di un *controllore inverso* si può calcolare semplicemente come:

$$u(k) = \frac{1}{b_1} [r(k+1) - f(u(k-1), \dots, u(k-m+1), y(k-1), \dots, y(k-m+1))]$$

dove  $r(k+1)$  denota l'andamento dell'uscita desiderata (ovvero il segnale di riferimento) nel successivo istante di tempo.

Il modello di  $f(\cdot)$  deve essere computazionalmente semplice perché bisogna calcolare l'azione di controllo nell'istante  $k$  a partire dalla misura dell'uscita nello stesso istante, cosa che richiede calcoli veloci.

## 7 Approssimatori

Cercare un approssimatore corrisponde a scegliere la tipologia di regressori da fornire in ingresso al sistema e poi trovare il vettore dei parametri seguendo lo stesso approccio impiegato nell'identificazione dei sistemi lineari (orientato alla minimizzazione dei minimi quadrati). Il modello è descritto dalla funzione:

$$y(k) = f(\varphi(k)) = f(x)$$

Con  $f$  non lineare ed  $x$  vettore dei regressori. L'obiettivo è trovare un approssimatore che approssimi  $f$  in tutto l'intervallo di ingressi di interesse:

$$\hat{y} = g(x) = g(x, w)$$

dove  $w$  indica il vettore dei parametri dell'approssimatore.

L'errore di predizione è:  $\epsilon = y - \hat{y}$ . L'obiettivo è trovare il vettore dei parametri che permette di minimizzare questo errore.

Possiamo diversificare gli approssimatori in:

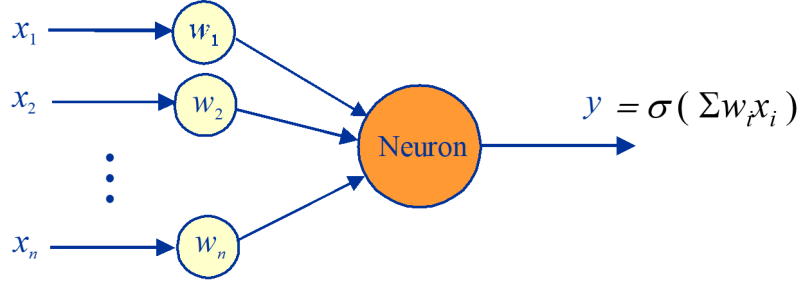
- *lineari*:  $\hat{y} = ax$  ( $w = a$ )
- *lineari nei parametri (LIP)*:  $\hat{y} = \phi_1(x)\theta_1 + \dots + \phi_n(x)\theta_n = \phi^T \theta$  ( $w = \theta$ )
- *non-lineari*:  $\hat{y} = g(x, w)$

## 7.1 Famiglie

- **Funzioni lineari o affini:** utilizzate per rappresentare l'uscita con l'equazione di una retta (la pendenza della retta rappresenta l'attributo lineare mentre il termine noto indica la caratteristica affine).
- **Polinomi:** l'uscita dell'approssimatore è espressa come combinazione lineare di potenze dell'ingresso. Il vantaggio di tali approssimatori è la caratteristica di linearità nei parametri, che permette di usare semplici algoritmi ai minimi quadrati. L'espansione in serie di Taylor è un metodo usato per cercare questo tipo di approssimatori. L'approssimazione polinomiale però ha lo svantaggio della *curse of dimensionality*, perché il numero di parametri cresce rapidamente al crescere del numero delle variabili e dell'ordine del polinomio. Occorre tenere presente che se si vuole utilizzare un approssimatore polinomiale per approssimare una funzione a due variabili, il polinomio dovrà contenere anche i prodotti tra le potenze di una variabile e quelle dell'altra (avremo una crescita notevole del numero dei parametri). Quindi questo tipo di approssimazione è scelta nel caso in cui la funzione sia ben rappresentabile impostando un ordine basso (ovvero se  $f$  non è molto variabile nella regione in cui deve essere approssimata).
- **Splines:** suddividendo la regione in cui vogliamo approssimare la funzione in varie sottoregioni ed usando un polinomio diverso (di ordine basso) per ciascuna sotto-regione otteniamo un insieme di funzioni polinomiali opportunamente raccordate e risolviamo i problemi descritti in precedenza. I nodi (knots o breakpoints) di una spline sono i punti di confine di ciascun sub-intervallo e possono essere uniformemente o non uniformemente spazati. Una spline di ordine  $k$  ad  $N$  sottoregioni approssima la funzione con  $N$  polinomi di ordine  $k-1$  (quindi ha  $Nk$  parametri).
- **Reti neurali:** categoria di approssimanti più moderna e potente, ispirata alle reti di cellule neurali presenti nel cervello degli organismi biologici.

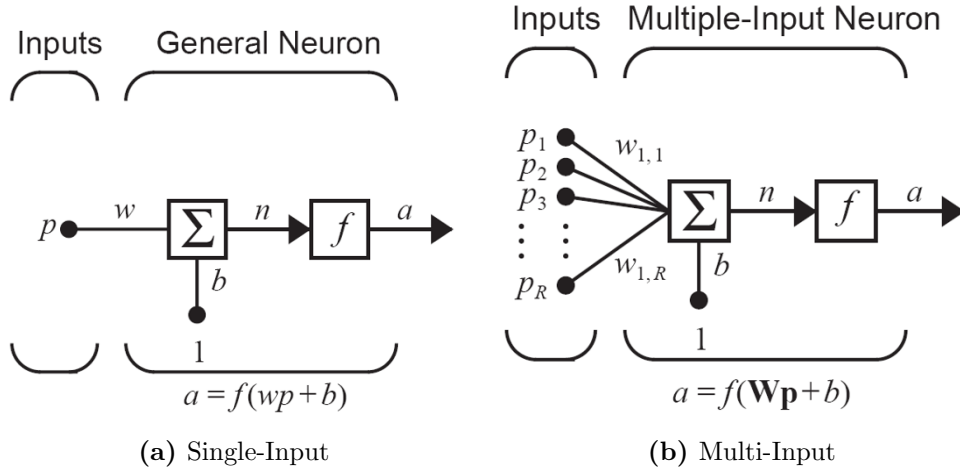
## 8 Neural Networks

Una rete neurale artificiale (**ANN: Artificial Neural Network**) è rappresentabile mediante un grafo orientato in cui ciascun nodo rappresenta un neurone. A ciascun arco del grafo corrisponde una sinapsi (rappresentata da un parametro numerico che prende il nome di peso).



**Figura 8.1:** Neurone artificiale

Il singolo neurone è un sistema MISO (Multiple-Input Single-Output), in cui tutti gli ingressi  $x_i$  del neurone sono collegati ad esso tramite dei pesi  $w_i$ . Il nucleo ospita una funzione matematica elementare che elabora gli  $n$  ingressi pesati fornendo l'uscita  $y$  (calcolata come somma pesata). L'uscita di ciascun neurone presenta inoltre un peso costante  $w_0$  che prende il nome di *bias*.



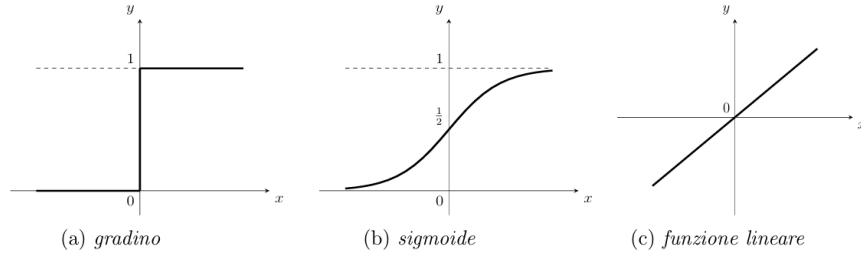
**Figura 8.2:** Neurone artificiale

dove  $p$  è l'ingresso,  $b$  è il bias,  $w$  il peso sinaptico associato all'ingresso ed  $a$  è l'uscita. Nel caso Multi-Input abbiamo un vettore di  $R$  ingressi, quindi avremo una matrice dei pesi  $W$  ed un vettore degli ingressi  $p$ , per cui  $Wp = \sum_{i=1}^R p_i w_{1i}$ .

La funzione che implementa il neurone è detta *di attivazione* ed è scelta in base all'applicazione a cui è destinata la rete neurale.

$$y^{step} = \begin{cases} 0 & \text{if } net \geq 0 \\ 1 & \text{if } net < 0 \end{cases} \quad y^{sigmoid} = \frac{1}{1 + e^{-net}} \quad y^{linear} = net$$



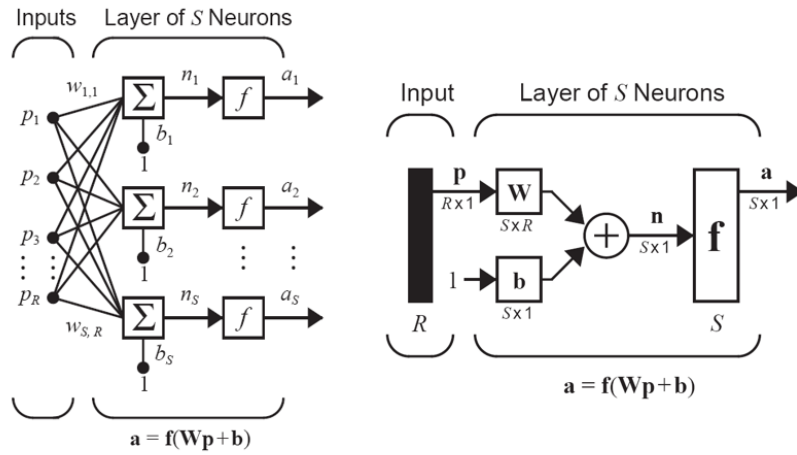


**Figura 8.3:** Funzioni di attivazione

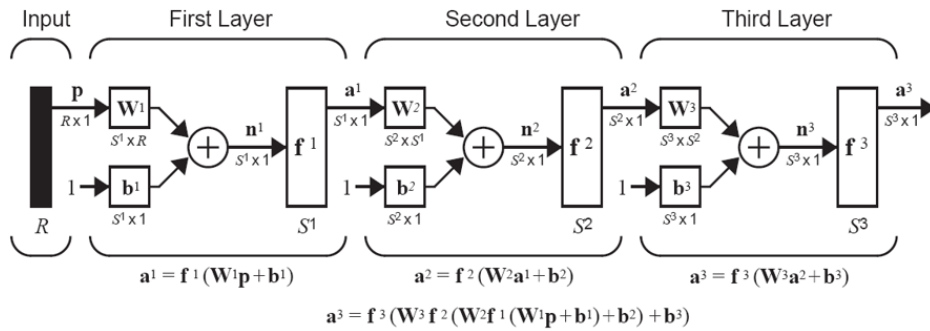
Il gradino a soglia  $\theta$  (secondo McCulloch e Pitts):  $y = \text{sign}(\sum_{i=1}^n w_i x_i - \theta)$ .

Se la somma pesata di ingressi viene superata dalla soglia, la differenza cambia di segno e l'uscita  $y$  può valere -1 o 1.

Lo strato di una rete di neuroni è ottenuto aggregando più neuroni in parallelo. Che può essere rappresentata anche in forma più compatta come nella figura a destra.



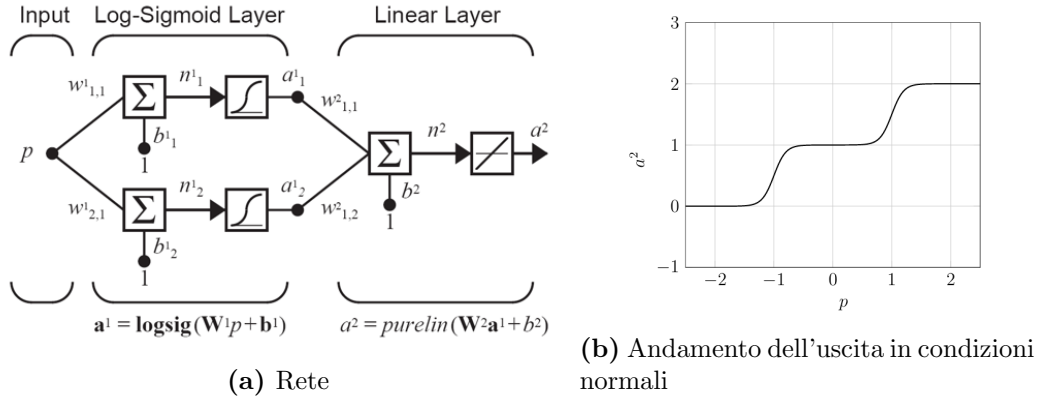
**Figura 8.4:** Strato di una rete neurale



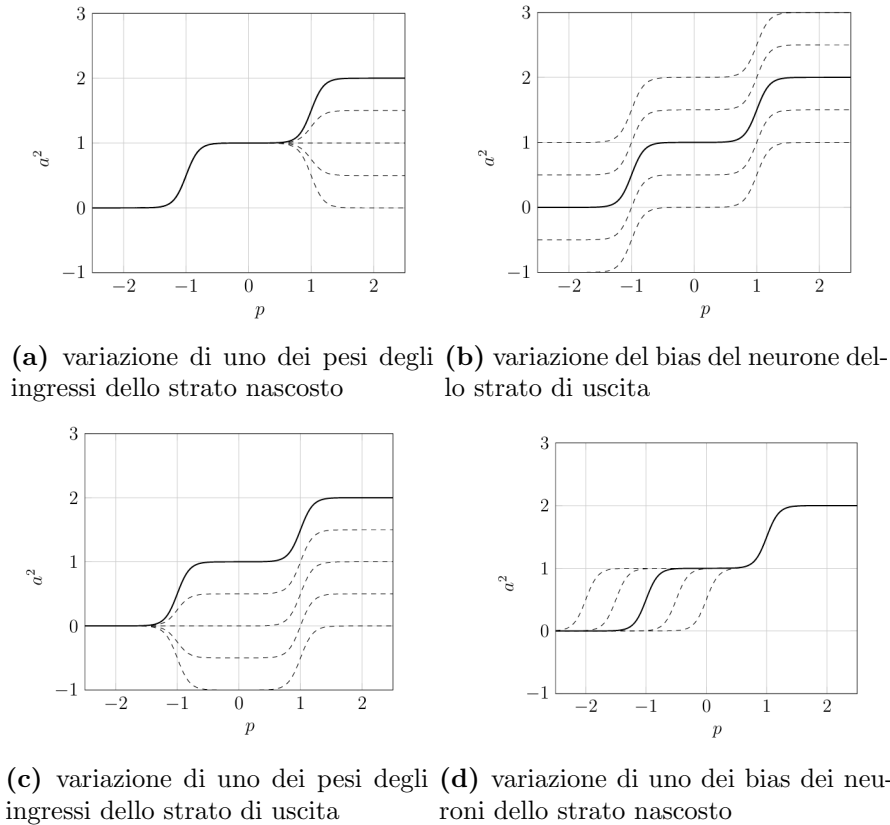
**Figura 8.5:** Multi-Layer network

## Esempio

Osserviamo la rete neurale composta da uno strato nascosto avente due neuroni con funzioni di attivazione sigmoidali ed uno strato di uscita che combina linearmente (funzione lineare) le uscite dello strato nascosto.



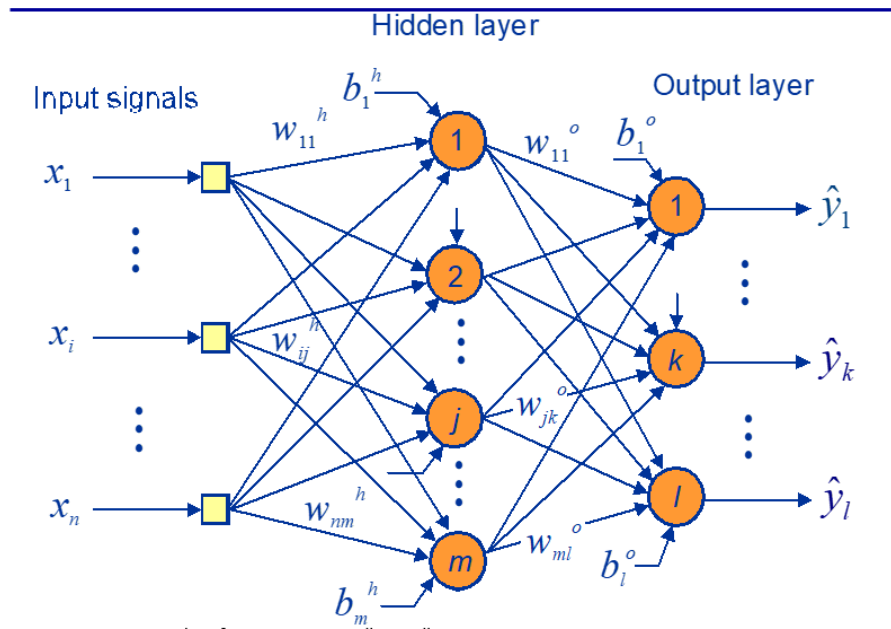
**Figura 8.6:** Hagan et. al Example



**Figura 8.7:** Risultati ottenuti con le rispettive modifiche

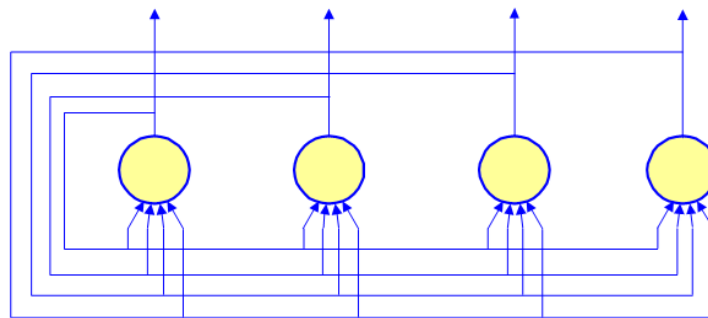
Una topologia comune è la *Multi-Layer Perceptron*, costituita da strati (o layer). Uno strato è definito come l'insieme di neuroni che condividono gli

stessi ingressi. Le uscite di uno strato costituiscono gli ingressi per lo strato successivo. In questa architettura vi è un Input Layer (attraverso il quale i dati iniziali sono immessi nella rete), un Output Layer (che contiene tanti neuroni quante sono le uscite del sistema) e da uno o più Hidden Layer (i quali possono contenere un numero qualsiasi di neuroni).



**Figura 8.8:** Rete neurale con architettura MLP

In figura notiamo una rete con  $n$  ingressi,  $l$  uscite e contenente un solo strato nascosto composto da  $m$  neuroni.



**Figura 8.9:** Rete neurale di Hopfield

L'architettura di una rete MLP<sup>7</sup> è di tipo *feed-forward*, mentre esistono altri tipi di reti in cui tutti i neuroni realizzano un *feed-back*<sup>8</sup>. L'introduzione dei

<sup>7</sup>Il termine  $w_{ij}^h$  indica il peso relativo alla connessione fra l' $i$ -esimo ingresso ed il  $j$ -esimo neurone,  $h$  sta ad indicare lo strato di destinazione (hidden), quando troviamo  $o$  ci riferiamo all'output layer.

<sup>8</sup>Questo tipo di reti vengono dette *ricorrenti*. Per realizzarle potrebbe essere necessario l'utilizzo di delay.

loop algebrici aumenta la potenza della rete ma complica le problematiche connesse ad esse.

## 8.1 Backpropagation

Addestrare una rete neurale significa trovare tutti i pesi per tutte le interconnessioni neuronali, inclusi i bias. Questi parametri poi possono essere migliorate operando successive iterazioni.

L'uscita dei neuroni nel  $j$ -esimo strato nascosto è:

$$h_j = \sigma \left( \sum_{i=1}^n w_{ij}^h x_i + b_j^h \right) = \sigma \left( \sum_{i=0}^n w_{ij}^h x_i \right) = \tanh \left( \sum_{i=0}^n w_{ij}^h x_i \right)$$

con  $\sigma \rightarrow \text{sigmoid}$ , impiegata come funzione di attivazione.

Ricordiamo l'espressione della tangente iperbolica:  $\tanh(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$ .

L'uscita dei neuroni è:

$$\hat{y}_k = \ell \left( \sum_{j=1}^m w_{jk}^o h_j + b_j^o \right) = \ell \left( \sum_{j=0}^m w_{jk}^o h_j \right) = \sum_{j=0}^m w_{jk}^o h_j$$

con  $\ell \rightarrow \text{linear}$ . Si è scelta la bisettrice  $y = x$ .

L'algoritmo di retropropagazione segue il classico approccio orientato alla minimizzazione dell'errore di predizione:

$$e_k = y_k - \hat{y}_k \quad k = 1, \dots, l$$

$$J(w) = \frac{1}{2} \sum_{k=1}^l \sum_{q=1}^N e_{kq}^2$$

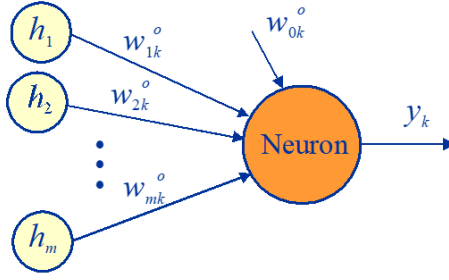
dove  $N$  è il numero di data points.  $l$  è il numero di uscite,  $y$  il dato,  $\hat{y}$  l'uscita della rete neurale ed  $w$  il vettore dei parametri che minimizza  $J$ .

$$w_k^o(p+1) = w_k^o(p) - \alpha \nabla J(w_k^o) \iff w_{jk}^o(p+1) = w_{jk}^o(p) - \alpha \frac{\partial J}{\partial w_{jk}^o}$$

L'apprendimento consiste nell'aggiornare i pesi seguendo un approccio di minimizzazione dell'errore lungo più epoche. Un'epoca, indicata con la lettera  $p$  è l'insieme di iterazioni necessarie ad effettuare un solo aggiornamento per tutti i pesi della rete neurale.

$$\nabla J(w_{jk}^o) = \left( \frac{\partial J}{\partial w_{1k}^o}, \frac{\partial J}{\partial w_{2k}^o}, \dots, \frac{\partial J}{\partial w_{mk}^o} \right)^T$$

dove  $\nabla J(w_{jk}^o)$  è il gradiente di  $J$  rispetto al peso,  $\alpha$  è il *learning rate*.



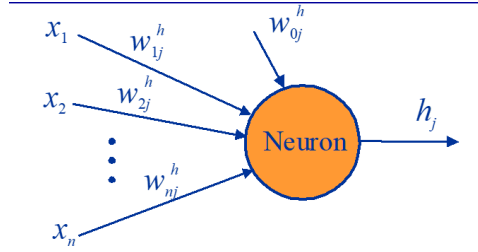
**Figura 8.10:** Neurone di uscita

$$J(w_{jk}^o, w_{ij}^h) = \frac{1}{2} \sum_{k=1}^l e_k^2$$

Le derivate nella backpropagation si calcolano con la "chain rule" (scomponendole in altre derivate):

$$\frac{\partial J}{\partial w_{jk}^o} = \frac{\partial J}{\partial e_k} \frac{\partial e_k}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial w_{jk}^o} = e_k \cdot (-1) \cdot h_j = -j_j e_k$$

Questo risultato vale per tutti i neuroni dello strato di uscita e permette di calcolare la derivata senza effettuare effettivamente operazioni di derivazioni. I parametri da cui dipende il gradiente richiesto sono noti dall'iterazione precedente (quindi già presenti in memoria).



**Figura 8.11:** Neurone dello strato nascosto

Per i neuroni dello strato nascosto vale un ragionamento analogo a quello fatto fin'ora. L'uscita del generico neurone dell'hidden layer è:

$$h_j = \tanh(net_j)$$

L'ingresso della funzione di trasferimento invece è:

$$net_j = \sum_{i=0}^n w_{ij}^h x_i$$

Anche in questo caso possiamo evitare di effettuare le derivate (salvo il calcolo della derivata della tanh:  $\sigma'$ ).

$$\frac{\partial J}{\partial w_{ij}^h} = \frac{\partial net_j}{\partial w_{ij}^h} \frac{\partial h_j}{\partial net_j} \frac{\partial J}{\partial h_j} = x_i \cdot \sigma'_j(h_j) \cdot \sum_{k=1}^l (-e_k w_{jk}^o)$$

### 8.1.1 Strategie di Apprendimento

**Batch:** tutti i campioni del data set sono considerati prima di effettuare l'aggiornamento dei pesi. Il gradiente calcolato per ogni singolo dato si somma a quello ottenuto dai precedenti per avere un valore di gradiente mediato su tutti i dati.

**Incremental:** i pesi si modificano dopo ogni singolo dato.

### 8.1.2 Error backpropagation: algorithm

Si *inizializzano* tutti i pesi con piccoli<sup>9</sup> numeri casuali.

*Ripetere:*

1. Inserimento dei dati di training e calcolo delle uscite della rete neurale;
2. **discesa del gradiente** sui neuroni dell'Output Layer:

$$w_{jk}^o(p+1) = w_{jk}^o(p) + \alpha h_j e_k$$

3. discesa del gradiente per l'Hidden Layer:

$$w_{ij}^h(p+1) = w_{ij}^h(p) - \alpha x_i \sigma'_j(h_j) \sum_{k=0}^l (-e_k w_{jk}^o)$$

fino al raggiungimento di un certo numero di epoche.

Un metodo alternativo è applicare il metodo del gradiente discendente del secondo ordine, stimando la matrice Hessiana (non abbiamo più il learning rate alfa):

$$w(p+1) = w(p) - H(w(p)) \nabla J(w(p))$$

con  $w(p) = w_{ij}^h, w_{jk}^o, \dots$

## Supervised Learning

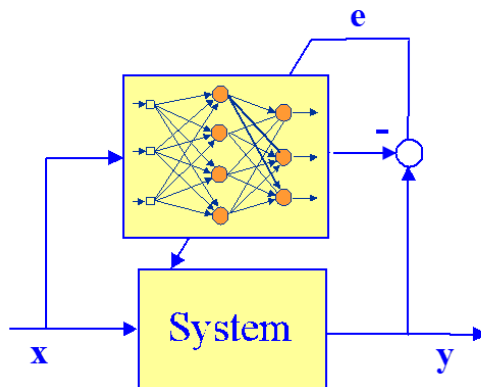
Addestramento di modelli (solitamente lineari), confrontando l'uscita del modello con l'uscita del MGD.

## Unsupervised Learning

L'obiettivo primario non è addestrare l'approssimatore a riprodurre un'uscita, ma piuttosto a svolgere uno specifico compito ("task"). La misura della bontà del modello non dipende dall'errore di predizione, ma dall'efficacia con cui il compito viene eseguito.

---

<sup>9</sup> vengono scelti valori non elevati perché le funzioni di attivazione dei neuroni potrebbero saturare.



**Figura 8.12:** Supervised learning

## 8.2 Radial basis function network

Le reti a base radiale sono particolari reti neurali aventi un solo strato nascosto e le cui funzioni di attivazione sono del tipo:

$$\sigma_i(x) = e^{-\frac{\|x - c_i\|^2}{s_i^2}}$$

dove  $x$  rappresenta il vettore degli ingressi,  $c_i$  il centro della funzione gaussiana ed  $s_i$  è lo spread (la varianza).

L'uscita della rete è data da:

$$y_j = \sum_{i=1}^n \theta_{ij} \sigma_i(x) = \sum_{i=1}^n \theta_{ij} e^{-\frac{\|x - c_i\|^2}{s_i^2}}$$

Una funzione a base radiale dipende unicamente dalla distanza dall'origine o da un centro specificato. La norma utilizzata è solitamente quella euclidea. Un neurone a base radiale è maggiormente attivato se l'ingresso è più vicino al centro della sua gaussiana; al contrario, se l'ingresso si allontana dal centro, l'uscita del neurone tende a zero (viene scarsamente attivato).

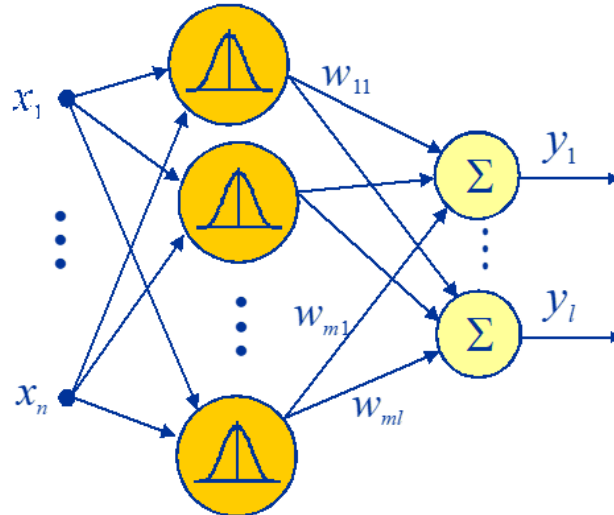
Ogni funzione gaussiana rappresenta il centro di un modello locale della funzione da approssimare. Se quest'ultima mostra variazioni significative solo in determinate regioni dello spazio degli ingressi, è utile condensare più gaussiane in quelle aree (utilizzando, invece, meno gaussiane con dispersioni più ampie nelle regioni meno rilevanti).

Il punto di forza delle reti neurali a base radiale è la *linearità nei parametri*. Se i centri e le varianze delle gaussiane sono fissati a priori si verifica questa proprietà.

Esistono due approcci principali per l'ottimizzazione:

- *LIP (lineare)*: i scelgono centri e dispersioni in modo da coprire uniformemente lo spazio degli ingressi. Si ottimizzano solo i pesi, che sono i parametri lineari sull'uscita.

- *non lineare*: Si includono nell'ottimizzazione anche i centri e le dispersioni di ciascuna gaussiana. Questo rende il problema di identificazione non lineare e richiede l'uso di algoritmi più complessi.



**Figura 8.13:** Rete neurale artificiale a base radiale con  $n$  ingressi ed  $l$  uscite

### 8.3 Considerazioni

Due teoremi che evidenziano l'efficacia e la potenza delle reti neurali (feed-forward) come approssimatori di funzioni sono:

1. **Cybenko** (1989): *Una rete neurale feed-forward che possiede almeno uno strato nascosto ed un numero sufficiente di neuroni al suo interno può approssimare una qualsiasi funzione  $R^p \rightarrow R^n$  definita in un intervallo compatto.*
2. **Barron** (1993): *Una rete neurale feed-forward con uno strato nascosto e neuroni nascosti, aventi funzioni di attivazione sigmoidali, può raggiungere un errore quadratico integrale dell'ordine  $J = O(1/h)$ .*

La combinazione di questi due teoremi sottolinea un vantaggio decisivo delle reti neurali artificiali rispetto ad altre categorie di approssimatori:

Una rete neurale opportunamente progettata è in grado di approssimare una qualunque funzione continua (*approssimatore universale*). La precisione raggiunta dipende solamente dal numero di neuroni nascosti ( $h$ ), ed è indipendente dalla dimensione dello spazio degli ingressi ( $p$ ).

Ricapitolando, mentre Cybenko afferma che la rete può approssimare, Barron indica quanto velocemente l'approssimazione diventa accurata in funzione della dimensione della rete nascosta.

Questo contrasta nettamente gli approssimatori polinomiali, il cui errore di approssimazione è dell'ordine  $J = O(1/h^{2/p})$ . Notiamo che l'errore aumenta man mano che la dimensione cresce (fenomeno noto come *Curse of Dimensionality*).