Università degli Studi di Roma "La Sapienza"

Facoltà di Ingegneria
Dipartimento INFO-COM
Dottorato in Ingegneria dell'Informazione e della Comunicazione
XIV Ciclo

Tesi di Dottorato

Ottimizzazione Strutturale di Reti Neurofuzzy

Docente guida:
Ch.mo Prof. Giuseppe Martinelli

Dottorando:
Massimo Panella

Roma, Dicembre 2001

A mia madre.

“Benché io sia tra quelli che hanno molto coltivato le matematiche, non ho mai tralasciato di meditare sulla filosofia, fin dalla giovinezza; poiché ritenevo sempre che fosse possibile stabilire in tal campo alcunché di solido, per mezzo di dimostrazioni chiare. M’ero già molto inoltrato nel paese degli Scolastici, allorché le matematiche e gli autori moderni mi indussero ad uscirne, ancora giovane. I loro eleganti procedimenti per spiegare meccanicamente la natura mi attrassero, e io disprezzai con ragione il metodo di coloro che non impiegano se non forme o facoltà che non insegnano nulla. Ma poi, avendo cercato di approfondire i principi stessi della meccanica, per rendere ragione delle leggi della natura che l’esperienza ci fa conoscere, mi accorsi che non basta considerare unicamente una massa estesa, ma occorre impiegare anche il concetto della forza, che è intelligibilissimo, benché appartenga al dominio della metafisica. Inoltre m’andavo convincendo che l’opinione di coloro che trasformano o degradano le bestie in pure macchine, benché in apparenza possibile, è inverosimile, anzi, contraria all’ordine delle cose.”

[dal *Nuovo sistema della natura*, Gottfried Wilhelm Leibniz]

INDICE

Capitolo 2 - Il problema della rappresentazione dei dati

Capitolo 3 - Proprietà delle reti neurofuzzy

INTRODUZIONE

Lo scopo di un sistema di modellamento è quello di definire un sistema di calcolo che permetta di risolvere in modo automatico ed efficiente molti dei problemi presenti nella gran parte delle attività scientifiche, soprattutto di tipo ingegneristico. Alla base di tali problemi si pone un'attività di analisi della struttura dei dati a disposizione, che rappresenta anche la simulazione di uno degli aspetti più affascinanti ed incogniti del funzionamento del cervello umano e del pensiero da esso prodotto.

Le applicazioni di un sistema di modellamento sono le più svariate e sono oramai affermate non solo nell'ambito della ricerca scientifica, ma anche in quello della produzione industriale e commerciale. Si va dai sistemi di autodiagnostica clinica ai dispositivi di riconoscimento ottico e vocale, dai sistemi di analisi e predizione finanziaria a quelli di compressione dei segnali, dai problemi di approssimazione funzionale a quelli di classificazione non supervisionata, e così via. A monte della progettazione di un sistema di modellamento sta sicuramente la scelta del modello matematico che si vuole implementare e dell'algoritmo utilizzato per determinarne i parametri. Nondimeno, a fronte delle molteplici soluzioni che si possono ottenere a valle del modellamento, diventa anche importante poter individuare le modalità con cui scegliere la soluzione "ottima" al problema con un criterio che risulti il più oggettivo possibile.

La tendenza recente nell'ambito della ricerca sta portando alla definizione di strutture, algoritmi e procedure di ottimizzazione che sono sempre più integrate tra loro e fanno uso delle moderne tecnologie di calcolo e di inferenza induttiva. Accanto ai modelli computazionali aristotelici, basati sulla millenaria logica esclusiva, e a quelli probabilistici, la cui formalizzazione teorica risale oramai a qualche secolo fa, si sono affermate tecniche più flessibili e sofisticate, prime fra tutte le reti neurali e la logica fuzzy.

E’ in questo ambito che ci si è mossi per sviluppare il lavoro di ricerca che ha caratterizzato il corso di dottorato e che si vuole presentare in questa tesi. Come indicato dal suo titolo, sono stati scelti proprio i modelli neurofuzzy come base di sviluppo di nuovi sistemi di modellamento che utilizzano anche le più recenti tecniche di ottimizzazione strutturale. In altre parole, la sintesi dei parametri del modello è integrata dalla determinazione automatica della complessità strutturale del modello stesso (ossia del numero dei suoi parametri), affinché ne sia massimizzata la capacità di generalizzazione. E’ bene precisare che, per ragioni di sintesi e di chiarezza espositiva, in questa tesi non sarà presentata tutta l’attività svolta nei tre anni di dottorato. Si è preferito concentrare l’attenzione su un particolare sistema di modellamento e sulla procedura di sintesi per esso realizzata, in modo tale da evidenziare i contributi più originali e significativi che hanno caratterizzato il lavoro di ricerca. Ad ogni modo, di seguito è brevemente riassunta l’attività di ricerca svolta durante il corso di dottorato, con l’aggiunta dei riferimenti bibliografici relativi alle pubblicazioni in cui il sottoscritto ha apportato, in qualità di autore, un contributo significativo alla loro stesura.

Durante il primo anno sono state poste le basi per il lavoro di ricerca e per lo sviluppo di nuovi modelli neurofuzzy. Particolare attenzione è stata dedicata allo studio del modellamento dei dati da elaborare mediante un processo di clustering in un opportuno spazio di rappresentazione degli stessi. Dai cluster determinati è infatti possibile sintetizzare le regole (componenti) che costituiscono l’architettura di una rete neurofuzzy. Una volta fissato il modello di riferimento di tali cluster (gaussiane, iperellissi, iperparallelepipedi, ecc.), il problema è poi determinare i parametri che caratterizzano il modello di ciascun cluster (fitting) e, se non fissato a priori, anche il numero complessivo dei cluster stessi.

Riguardo al primo problema, è stato introdotto l’utilizzo di tecniche di modellamento locale dei dati basate sull’analisi alle componenti principali (PCA). L’utilizzo della PCA è sicuramente vantaggioso nel determinare una corretta corrispondenza tra il modello del cluster e la reale distribuzione dei dati che, localmente, sono ad esso associati. Un primo contributo è stato fornito nell’ambito delle reti neurali basate sui modelli Min-Max di Simpson. E’ stato infatti proposto in [Frattale Mascioli 1999] un algoritmo denominato “Min-Max Generalizzato”

(GMMB), nel quale il modello di ciascun cluster è un iperparallelepipedo che può essere orientato in qualsiasi direzione dello spazio dei dati in base alle direzioni principali dei dati stessi. Il nuovo algoritmo, nettamente più efficace dell'algoritmo Min-Max originale, è stato utilizzato in applicazioni reali riguardanti, ad esempio, il controllo di traiettorie di un robot in ambiente sconosciuto. Ma soprattutto, questo algoritmo è stato alla base dello sviluppo degli algoritmi "Min-Max Generalizzati Gerarchici" (HGMMB) applicati con successo all'indicizzazione di sequenze per la gestione di database video [Frattale Mascioli 2000].

Nel secondo anno di dottorato, sempre sulla base delle tecniche di modellamento locali basate sulla PCA, sono stati estesi alcuni degli algoritmi di clustering che fanno uso di modelli statistici basati su misture di cluster gaussiani. In questo caso, il modello gaussiano di ciascun cluster ricorre all'uso di *variabili latenti* che permettono una rappresentazione più parsimoniosa (cioè con un numero inferiore di parametri) del modello stesso. Ancora una volta, le variabili latenti sono legate ad un numero opportuno di componenti principali ricavate dalla PCA sul sottoinsieme locale di dati che il cluster gaussiano sta modellando. Come dimostrato in [Frattale Mascioli 2000(bis)], si ottiene così un duplice vantaggio: la possibilità di ottenere una riduzione lineare locale dei dati e quindi una riduzione globalmente non lineare degli stessi; una migliore corrispondenza tra il modello gaussiano e la distribuzione reale dei dati, in quanto si tiene conto solo delle sue caratteristiche predominanti (le sue prime componenti principali).

Successivamente, è stato affrontato il problema cruciale nella sintesi delle reti neurofuzzy, ossia la determinazione della loro complessità al fine di ottimizzarne la capacità di generalizzazione. In altre parole, è stato affrontato il problema della determinazione del numero totale di cluster, ovvero di parametri, che caratterizzano la rete. Un importante ruolo è svolto in questo ambito dalle tecniche di free-clustering, le quali non fissano a priori il numero di cluster. Queste tecniche sono importanti laddove non sono possibili ipotesi attendibili a riguardo e/o si vuole rendere il processo di clustering il più possibile svincolato da parametri critici, quali appunto quelli relativi al numero totale di cluster. Sono state introdotte a tale proposito nuove tecniche di ottimizzazione gerarchiche e costruttive le quali, sulla base di opportune procedure di scissione dei cluster, ne aumentano progressivamente il numero

all'interno della struttura. Si ottengono così una migliore velocità di convergenza degli algoritmi di fitting dei cluster e la possibilità di raggiungere un ottimo che, se non proprio globale, è comunque un ottimo locale adeguato all'applicazione di interesse. Una volta determinata la gerarchia di cluster, il numero ottimale di essi può essere scelto utilizzando le classiche tecniche basate sulla teoria dell'apprendimento (Indici di Davies-Bouldin, Minimum Description Length, Akaike Information Criterion, Rasoio di Occam, ecc.) oppure, come meglio spiegato nel seguito, tramite il principio dello *spazio di scala*.

I contributi originali si sono concretizzati in questo contesto nello sviluppo di una versione ottimizzata dell'algoritmo Expectation-Maximization (EM), tipicamente usato per la determinazione dei suddetti cluster gaussiani. In [Panella 2000] è stata proposta una procedura di ottimizzazione denominata SHEM, la quale consiste nella determinazione del numero complessivo di cluster presenti nella struttura mediante l'approccio gerarchico costruttivo basato sullo spazio di scala. Il clustering basato sullo spazio di scala si ispira al modo di funzionare del cervello umano in relazione alla vista. Gli oggetti più facilmente individuabili in una scena sono quelli più persistenti al variare della distanza dell'osservatore. Sulla base di questo criterio, il clustering viene eseguito iterativamente variando un parametro (scala) che simula la distanza dell'oggetto da analizzare (struttura topologica dei campioni nello spazio di ingresso) dall'osservatore fittizio. La tecnica dello spazio di scala può essere inoltre inserita in un meccanismo più generale tale da imitare il modo in cui si verificano i processi naturali di aggregazione (attrazione gravitazionale ed elettrostatica) o scissione (big bang). I già citati algoritmi HGMMB, ad esempio, seguono una procedura gerarchica di scissione dei cluster di questo tipo. Seguendo un approccio costruttivo che aumenta progressivamente il numero di cluster, è possibile tenere sempre sotto controllo la complessità sia della rete che dell'algoritmo di sintesi; inoltre, sono del tutto eliminati i problemi di inizializzazione dell'algoritmo stesso, dato che all'inizio esiste un unico cluster che è possibile determinare in modo univoco. L'uso di tecniche legate allo spazio di scala negli algoritmi di clustering consente di individuare la partizione ottimale tramite un particolare indice di validazione, chiamato *indice di stabilità*, che sceglie come cluster ottimi all'interno

della gerarchia generata quelli che persistono maggiormente alla variazione del parametro di scala [Frattale Mascioli 2000].

Lo sviluppo delle tecniche di sintesi è stato messo in opera applicando le risultanti reti neurofuzzy a problemi reali di elaborazione dell'informazione. Oltre ai già citati problemi di clustering, sono stati affrontati anche i problemi di classificazione, approssimazione funzionale e predizione. In particolare, è stato sviluppato un algoritmo per la sintesi dei classificatori bayesiani sulla base di quanto già proposto in [Panella 2000]. I classificatori bayesiani sono infatti particolarmente adatti alla soluzione di problemi di classificazione non esclusiva, per i quali un campione può essere associato contemporaneamente a più di una classe. L'ottimizzazione proposta risulta più efficiente rispetto ai metodi più classici di ottimizzazione dei classificatori bayesiani stessi, sia in termini di accuratezza della classificazione che in termini di costo computazionale. Questa architettura per la classificazione non esclusiva è stata utilizzata, in collaborazione con l'Università degli Studi di Roma "Tor Vergata", in un contesto reale riguardante l'elaborazione del segnale musicale. A tale riguardo, sono stati proposti in [Costantini 2000, Costantini 2000(bis)] due sistemi di questo tipo per il riconoscimento (classificazione) dei singoli strumenti che compongono il segnale musicale stesso. In particolare, le note emesse dai diversi strumenti tradizionali, registrate ed opportunamente pre-elaborate, possono essere utilizzate come insieme di apprendimento (e di test) del suddetto classificatore bayesiano.

L'applicazione delle reti neurofuzzy ai problemi di classificazione ha anche riguardato lo sviluppo di un nuovo paradigma di apprendimento per i modelli di classificazione Min-Max. A tale riguardo, saranno presentati in [Rizzi 2002] i *classificatori a risoluzione adattativa* (ARC), basati su un algoritmo di apprendimento che fa uso di opportune procedure costruttive per la determinazione dei già citati iperparallelepipedi che costituiscono il classificatore. In aggiunta al suddetto lavoro, in cui è riassunta tutta una serie di risultati ottenuti durante il corso di dottorato, sono state anche presentate in [Rizzi 2000, Rizzi 2001] delle versioni di ARC che fanno rispettivamente uso di procedure di apprendimento ricorsive e dei modelli Min-Max generalizzati.

Durante il terzo anno di dottorato è stato finalizzato il lavoro degli anni precedenti andando a studiare l'applicazione più naturale delle reti neurofuzzy, ossia quella

relativa alla soluzione dei problemi di approssimazione funzionale. Tra gli approcci più interessanti, sono stati analizzati quelli riguardanti le reti FBF (Fuzzy Basis Function) e le reti ANFIS (Adaptive NeuroFuzzy Inference System). In particolare, è stato affrontato il problema dell'approssimazione funzionale dal punto di vista della sintesi delle reti neurofuzzy che sono in grado di espletare questo compito. Tali reti sono viste come un'estensione dei moderni circuiti digitali in grado quindi di risolvere problemi di elaborazione più complessi che vanno comunemente sotto il nome di Intelligent Signal Processing (ISP). Le procedure di clustering, classificazione, e di modellamento dei dati in generale precedentemente citate, oggetto del lavoro di ricerca nel corso di dottorato, sono state inquadrate ed utilizzate per la determinazione delle regole (componenti) alla base delle reti neurofuzzy [Frattale Mascioli 2001]. Tali procedure di modellamento vengono pertanto a ricoprire il ruolo della moderna sintesi circuitale al fine di determinare i circuiti basati su reti neurofuzzy utili alla risoluzione dei problemi tipici dell'ISP.

La possibilità di predire una grandezza di interesse riveste una grande importanza pratica. Dalla teoria dei sistemi dinamici caotici non lineari, è noto che il sistema stesso può essere modellato in un opportuno spazio (spazio di ricostruzione), sotto forma di un predittore. In particolare, il predittore può essere realizzato tramite un regressore non lineare, in grado di individuare tramite un'analisi di clustering le zone dello spazio degli ingressi in cui la funzione incognita può essere localmente approssimata tramite iperpiani interpolatori (linearizzazione a tratti della funzione incognita a partire da un campionamento della relazione ingresso-uscita).

La rete ANFIS basata su regole di Sugeno del primo ordine rappresenta una possibile soluzione a tale approccio. E' stata quindi messa a punto in [Panella 2001(ter)] una nuova ed originale tecnica di sintesi delle reti ANFIS, basata sulla determinazione diretta degli iperpiani interpolatori che determinano la struttura della funzione da approssimare e, successivamente, sulla soluzione di un opportuno problema di classificazione definito nello spazio di ingresso. Tale procedura di sintesi per iperpiani, chiamata OHCS, sarà oggetto di questa tesi e pertanto sarà ampiamente discussa nel seguito; anch'essa fa uso dei principi di ottimizzazione strutturale precedentemente citati. Analogamente, è stata introdotta in [Panella 2001] una nuova architettura neurofuzzy, simile all'ANFIS, ma basata su una mistura di cluster

gaussiani. A tale architettura può quindi essere direttamente applicato l'algoritmo SHEM per l'ottimizzazione della mistura. I predittori ottenuti mediante le tecniche OHCS e SHEM sono stati favorevolmente impiegati in problemi di predizione inquadrati in alcuni importanti progetti di ricerca di interesse nazionale. Si è trattato, in particolare, della previsione del consumo di energia elettrica (dati ACEA ed ENEL) e della previsione del comportamento degli agenti chimici inquinanti presenti in alcune importanti località italiane [Panella 2001(bis)].

In questa tesi particolare enfasi sarà posta sui sistemi di clustering costruttivo che permettono di implementare procedure di ottimizzazione strutturale particolarmente efficaci, sia dal punto di vista dell'accuratezza del modello ottenuto, sia del costo computazionale necessario per la sua sintesi. A proposito del modello considerato, ci si concentrerà sullo studio delle reti ANFIS per la soluzione dei problemi di approssimazione funzionale, e quindi si descriverà nel dettaglio la procedura di sintesi OHCS. I risultati ottenuti, che saranno ampiamente discussi nel seguito, hanno mostrato un indubbio miglioramento delle prestazioni relative agli algoritmi più celebri presenti nella letteratura tecnica del settore. Ciò risulta vero soprattutto in termini di flessibilità, tipica dell'approccio costruttivo, e di scelta del numero ottimo di regole da utilizzare nell'architettura della rete neurofuzzy. Soprattutto, gli algoritmi proposti si sono mostrati capaci di mantenere tali prestazioni su applicazioni tra le più diverse fra loro, dimostrando così di non essere limitati ad una specifica classe di problemi.

Nel cap. 1 saranno definiti in modo formale i problemi di modellamento e saranno inquadrati in alcune delle moderne discipline di calcolo prima menzionate. Nell'ambito del Soft Computing, che racchiude la gran parte di esse, particolare enfasi sarà posta nella descrizione delle reti neurali e della logica fuzzy. Saranno inoltre formalizzati, in modo del tutto innovativo ed originale, alcuni dei concetti fondamentali che hanno portato allo sviluppo delle efficaci tecniche di sintesi proposte in questa tesi.

Nel cap. 2 saranno descritte con un certo dettaglio le tecniche di rappresentazione dei dati sia in ingresso che in uscita al processo di modellamento, soprattutto in merito alle possibili operazioni di preprocessing e/o postprocessing effettuabili su essi. Tale descrizione risulta necessaria laddove si voglia costruire una solida base di valutazione

sia dei risultati che dello stesso lavoro svolto. Il fine, ovviamente, è quello di ottenere un certo grado di oggettività riguardo a “cosa” si stia analizzando ed a “quanto accuratamente” lo si stia facendo.

Nel cap. 3 saranno introdotti i concetti di base relativi alle reti neurofuzzy, con particolare riguardo alle reti di Mamdani ed alle reti ANFIS. In tale capitolo sarà dato un taglio particolare alla presentazione di tali sistemi di modellamento, inquadrando gli stessi nell’ambito delle più moderne tecniche di elaborazione dei segnali e di sintesi circuitale. Ma soprattutto, al fine di comprendere meglio l’efficacia delle soluzioni proposte, saranno anche evidenziate le problematiche presenti nei sistemi considerati, cercando di spiegare come e perché esse si possono presentare. Ciò ha richiesto un lungo lavoro di ricerca nella letteratura tecnica del settore, con una conseguente attività di sintesi dei risultati pubblicati. Tale impegno si è anche rivelato essenziale al fine di individuare le possibili tecniche, soprattutto euristiche, utili per la soluzione di alcuni dei problemi che tipicamente si manifestano durante la progettazione o lo sviluppo di un algoritmo di apprendimento.

Nel cap. 4 sarà proposta la tecnica di sintesi delle reti ANFIS basata su un originale approccio chiamato HCS (Hyperplane Clustering Synthesis). Ad una descrizione accurata dell’algoritmo di apprendimento, seguirà una descrizione dettagliata del suo funzionamento, del tipo di strutture che esso genera e del tipo di ottimizzazioni ad esso applicabili. Sarà quindi proposta un’ottimizzazione strutturale chiamata OHCS (Optimized HCS), che si è rivelata particolarmente efficace nel massimizzare la capacità di generalizzazione di una rete ANFIS.

Infine, nel cap. 5, la procedura di sintesi OHCS sarà utilizzata per la soluzione di alcuni specifici problemi di predizione. In particolare, saranno esposti i concetti di base relativi al problema di predizione di sequenze caotiche mediante la teoria della ricostruzione dinamica. In altre parole, si tratta di risolvere un problema di predizione tramite uno di approssimazione funzionale, in cui ben si adattano sistemi di modellamento come le reti neurofuzzy, rese più flessibili e robuste anche grazie all’ottimizzazione strutturale OHCS ivi proposta. Le conclusioni in merito ai concetti illustrati saranno discusse contestualmente alla presentazione dei risultati ottenuti.

Prima di proseguire, è doveroso rivolgere un ringraziamento al prof. Giuseppe Martinelli, nonché all'ing. Antonello Rizzi ed al prof. Fabio Massimo Frattale Mascioli, per la collaborazione fornita durante tutto il corso di dottorato. E' infatti da loro che ho ricevuto gli imprescindibili stimoli a cercare di risolvere i dubbi ed i problemi che ogni giorno hanno accompagnato la mia attività di ricerca.

capitolo
1

SISTEMI DI MODELLAMENTO E RETI NEUROFUZZY

1.1 Premessa

Lo sviluppo della civiltà umana, assieme alla sua crescente domanda di maggiore qualità ed efficienza nei servizi da essa richiesti, rende i nostri giorni sempre più appartenenti all'era dell'informazione. Un'era, cioè, dove l'attività fondamentale di ogni uomo e di ogni macchina è basata sullo scambio di informazioni con altri oggetti ed individui. Continuano quindi a nascere problemi di generazione, trasmissione, ed elaborazione dell'informazione, sempre più complessi e sempre più legati all'intelligenza che i sistemi impiegati per la loro soluzione devono possedere. Si potrebbe infatti affermare che ciò che caratterizza l'intelligenza non è la capacità di effettuare calcoli, ma piuttosto l'abilità del sistema di interagire con l'ambiente circostante ed estrapolare da esso le giuste informazioni necessarie per poter riconoscere o prevedere alcune situazioni. Da quanto appena detto si capisce perché i sistemi biologici, come ad esempio il cervello umano ed il suo complesso sensoriale, riescono ancora oggi ad essere più efficienti dei moderni e potenti sistemi di calcolo artificiali nella soluzione di alcuni problemi di trattamento dell'informazione.

In questo capitolo saranno presentate le moderne evoluzioni delle tecnologie di calcolo e di ragionamento, nonché il ruolo che esse giocano, o potrebbero giocare, nella soluzione dei suddetti problemi. Lo scopo è quello di inquadrare nell'ambito di queste tecnologie l'evoluzione dei sistemi di modellamento, dimostrando come

sempre più si cerchi di unire ai pregi dell'intuizione e della flessibilità del ragionamento umano la precisione e la velocità dei sistemi di calcolo artificiali.

1.2 Processi e modelli

Nei successivi paragrafi verranno introdotte alcune importanti definizioni che saranno di aiuto nella successiva presentazione dei sistemi di calcolo proposti. In particolare, verranno introdotte con formalismo logico-matematico le definizioni di alcuni dei problemi di analisi dei dati e di elaborazione dell'informazione, la soluzione dei quali ha costituito lo stimolo essenziale all'attività di ricerca riassunta in questa tesi.

Con il termine *processo* (o *sistema*) *orientato* si intende un fenomeno, reale o ipotetico, che ponga in relazione alcune variabili considerate come ingressi ad altre considerate come uscite. Il comportamento del processo orientato è pertanto caratterizzato dall'andamento delle sue uscite, e di un eventuale suo stato interno, in funzione dei suoi ingressi. In tal senso si può parlare di *sintesi* del modello (ovvero di *modellamento*) del processo orientato P, intendendo con tale attività la determinazione di un *modello* del processo P che rappresenti in modo astratto il comportamento del processo stesso con una stabilita precisione.

Spesso, tuttavia, il processo risulta *non orientato*, in quanto esso non è caratterizzabile dalla relazione tra ingressi ed uscite o, più semplicemente, alcune tra queste variabili non risultano osservabili. In tal caso, il comportamento di un processo non orientato è noto solo attraverso le configurazioni che esso assume, le quali sono definite sulla base dell'insieme di variabili osservate. Il modello di un processo non orientato non si riferirà, quindi, ad alcuna relazione ingresso-uscita, ma piuttosto alla relazione tra le varie configurazioni assunte dal processo nella sua evoluzione. Nel seguito della discussione, ove non diversamente specificato, col solo termine di "processo" si intenderà un processo orientato.

Un processo P si dice *computabile* se esiste un algoritmo in grado di calcolare le uscite in un numero di passi finito, sulla base dei valori assunti dagli ingressi e da un eventuale stato interno al processo. Si noti che, dal momento che siamo interessati alla

sintesi di modelli di processi e alla loro successiva implementazione (in software o in hardware), il requisito di computabilità è strettamente necessario.

Una vasta classe di processi può essere efficacemente rappresentata da *sistemi astratti orientati* a dimensione finita:

$$\begin{aligned} \dot{\underline{\sigma}}(t) &= f(\underline{\sigma}(t), \underline{x}(t); t) \\ \underline{y}(t) &= g(\underline{\sigma}(t), \underline{x}(t); t) \end{aligned} \qquad (1.1)$$

dove $\underline{\sigma} \in \Sigma$ è il vettore di stato, $\underline{x} \in X$ è il vettore degli ingressi ed $\underline{y} \in Y$ è il vettore delle uscite; le funzioni $f(\cdot)$ e $g(\cdot)$ sono dette, rispettivamente, *funzione di stato* e *funzione d'uscita*. Il modello si dice *lineare* se tali sono le funzioni di stato e di uscita. La rappresentazione 1.1 è valida nel caso in cui Σ, X e Y siano degli spazi normati; è importante sottolineare il fatto che, in molti casi pratici, questa condizione non è soddisfatta. Un processo P si dice *stazionario* se esiste una rappresentazione di P tale che tanto $f(\cdot)$ quanto $g(\cdot)$ non dipendono dal tempo t:

$$\begin{aligned} \dot{\underline{\sigma}}(t) &= f(\underline{\sigma}(t), \underline{x}(t)) \\ \underline{y}(t) &= g(\underline{\sigma}(t), \underline{x}(t)) \end{aligned} \qquad (1.2)$$

Un processo P si dice *combinatorio* se è nulla la dimensione dello spazio di stato. Pertanto, un processo stazionario e combinatorio ammette una rappresentazione del tipo:

$$\underline{y}(t) = g(\underline{\sigma}(t)) \qquad (1.3)$$

Il modellamento di un processo, orientato o non, può essere ottenuto tramite i due approcci brevemente discussi nei seguenti paragrafi.

1.2.1 Modellamento analitico

Ogniqualvolta l'attività di modellamento comporta una conoscenza della struttura (fisica o logica) interna al processo da modellare, si parla di modellamento *analitico*. Una tale attività consiste essenzialmente nei seguenti punti:

- la determinazione delle quantità rilevanti (variabili e costanti) interne al processo;

- la definizione di opportune relazioni tra tali quantità; tali relazioni possono derivare da leggi fisiche, oppure essere delle relazioni logiche tra le variabili considerate;
- la definizione di particolari e ricorrenti sezioni (fisiche o funzionali) del processo in esame e la loro caratterizzazione tramite relazioni matematiche volte a specificare le dipendenze intercorrenti tra le variabili coinvolte in ciascun blocco; si noti che la caratterizzazione di tali sezioni o blocchi logici è essa stessa un'attività di modellamento;
- la caratterizzazione (fisica o logica) della struttura stessa del processo e la definizione di opportune leggi di interazione tra elementi caratteristici (leggi o ipotesi strutturali).

Nella teoria dei circuiti, ad esempio, il passaggio da una generica struttura elettromagnetica ad un circuito a costanti concentrate viene effettuato definendo alcuni elementi circuitali caratteristici (generatori, resistori, condensatori, induttori, ecc.), i quali sono caratterizzati da relazioni costitutive determinate sulla base delle peculiarità fisiche della particolare sezione della struttura che ciascuno di essi rappresenta.

Altro esempio di modellamento analitico sono le procedure di stima di tipo *bayesiano*, le quali sono legate alla conoscenza a priori di alcune grandezze statistiche che caratterizzano il sistema su cui effettuare la stima.

Tramite il modellamento analitico è dunque possibile indagare sulla natura delle leggi che governano il funzionamento stesso del processo in esame. Questo è il pregio fondamentale di tale procedura, ma anche il suo limite principale. Infatti, in un gran numero di casi il processo deve essere considerato come una scatola nera, accessibile solamente tramite le sue variabili di ingresso e di uscita, senza la possibilità di formulare ipotesi a priori riguardo alla conoscenza del processo in esame. In queste situazioni è dunque necessario considerare una differente procedura di modellamento.

1.2.2 Modellamento data driven

Per modellamento *data driven* deve intendersi una qualunque attività che si prefigga lo scopo di sintetizzare il modello di un processo P, a partire dalla

conoscenza delle relazioni intercorrenti tra un insieme di campioni che caratterizzano il comportamento del processo stesso. Nel caso di processo orientato, ad esempio, tali campioni potrebbero essere quelli degli ingressi e/o delle relative uscite misurate per il processo. Nel caso di un processo non orientato, tali campioni potrebbero coincidere con alcune delle configurazioni assunte dal sistema.

E' importante sottolineare che il modellamento data driven, nel quale rientrano le procedure di stima di tipo *fisheriano*, non può e non deve essere considerato come sostituto della tradizionale attività di modellamento analitico; esso deve piuttosto essere pensato come un utile strumento in grado di risolvere istanze di problemi di modellamento che non possono essere affrontate con tecniche classiche. In pratica, le tecniche di modellamento data driven consentono di sintetizzare sistemi di calcolo in grado di "apprendere uno specifico compito tramite esempi". Al fine di stabilire una volta per tutte il significato rigoroso di questa espressione, che è alla base della definizione di rete neurale che sarà introdotta nel seguito, è necessario anzitutto distinguere tra due classi notevoli di problemi di modellamento data driven e fornire una tassonomia essenziale degli specifici problemi che possono essere affrontati.

1.3 Modellamento supervisionato e non supervisionato

E' oramai opinione comune che una tassonomia completa ed esaustiva dei problemi di modellamento data driven è praticamente impossibile da ottenere, soprattutto a causa della dipendenza del problema dal contesto in cui esso deve essere risolto. In questo paragrafo, pertanto, si porrà l'attenzione sulle analogie e sulle differenze esistenti tra i più generali problemi di modellamento, le definizioni dei quali verranno poi riprese nei capitoli pertinenti e discusse in maggior dettaglio.

Una possibile tassonomia dei più importanti problemi di modellamento data driven è illustrata in fig. 1.1. Si consideri un processo orientato $P : X \to Y$, dove X è lo spazio di ingresso (dominio di P) e Y è lo spazio di uscita (codominio di P). Definiamo *pattern* (o *campione*) *ingresso-uscita* di P la coppia $<\underline{x}, \underline{y}>$, dove $\underline{y} \in Y$ è l'uscita corrispondente al vettore $\underline{x} \in X$ secondo P, ossia $\underline{y}=P(\underline{x})$. Un qualunque insieme di pattern ingresso-uscita verrà detto *campionamento ingresso-uscita* di P. Con queste

posizioni, un problema di modellamento si dirà *supervisionato* se il processo da modellare è orientato e se è utilizzato (esplicitamente o implicitamente) un campionamento ingresso-uscita di P. Viceversa, un problema di modellamento si dirà *non supervisionato* se tale campionamento non è utilizzato, ovvero esso non è ottenibile (il processo è non orientato).

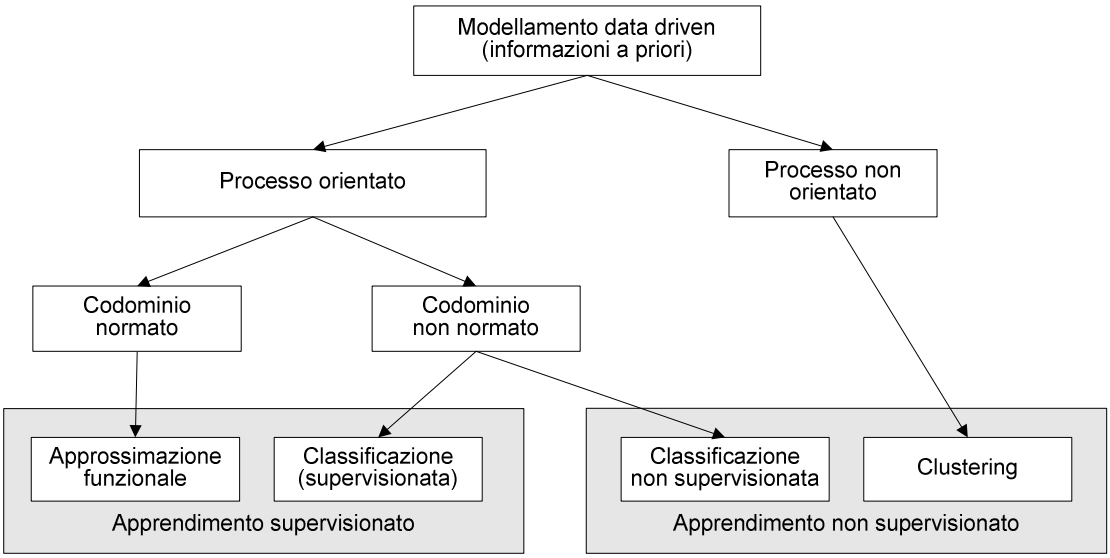


Figura 1.1 - Tassonomia dei problemi di modellamento data driven.

E' evidente che, nel caso di processo non orientato, non si è interessati al comportamento ingresso-uscita di un processo, ma alla sua caratterizzazione sulla base delle mutue relazioni tra le configurazioni del processo. Quest'ultime, come già detto, sono ottenute sulla base di un certo insieme di grandezze osservabili. Il compito che un problema di modellamento non supervisionato è chiamato ad espletare consiste quindi nel caratterizzare il processo tramite un'opportuna tassonomia delle sue possibili configurazioni, sulla base di un numero limitato di osservazioni. Dato un processo non orientato P, definiamo *pattern* (o *campione*) di P una qualunque configurazione del processo; lo spazio di tutti le possibili configurazioni viene detto *spazio caratteristico* di P. Definiamo, inoltre, *campionamento* di P un qualunque sottoinsieme dello spazio caratteristico di P.

Nel seguito verranno definiti i particolari problemi di modellamento associati alle varie situazioni possibili. Un ulteriore approfondimento sulla struttura delle possibili soluzioni ai problemi di classificazione e di clustering sarà fornita nel par. 2.9.

1.3.1 Problema di approssimazione funzionale

Siano S_{tr} e S_{ts} due campionamenti distinti di un processo orientato $P : X \rightarrow Y$, tali che $S_{tr} \cap S_{ts} = \varnothing$, con X e Y spazi normati. Usualmente l'insieme S_{tr} viene detto *insieme di apprendimento* (*training set*), mentre S_{ts} viene detto *insieme di test* (*test set*). Il problema consiste nel sintetizzare un modello M di P, sulla base delle sole informazioni contenute in S_{tr}, tale che sia minimizzata un'opportuna funzione d'errore definita su S_{ts}. Si vuole, cioè, che il modello sintetizzato riesca ad approssimare in modo opportuno il comportamento del processo per ogni possibile pattern di ingresso $\underline{x}$ che può essere fornito al processo. Tale proprietà è anche detta *capacità di generalizzazione* del modello di approssimazione funzionale.

In merito alle definizioni appena fornite è necessario precisare alcuni concetti. Innanzitutto, la funzione ingresso-uscita di P da approssimare è stata qui intesa in senso lato. Infatti, anche se il campionamento può essere effettuato sulla base degli effettivi ingressi ed uscite di P, tramite la relazione $\underline{y}=P(\underline{x})$, il legame ingresso-uscita da stimare potrebbe essere, ad esempio, la densità di probabilità $p(\underline{y}|\underline{x})$ [Panella 2001]. C'è poi da considerare che a tale classe appartengono molti dei problemi ingegneristici più importanti. Ad un problema di approssimazione funzionale possono essere infatti ricondotte tutte le procedure di stima ed in particolare quelle di *identificazione*, *predizione*, *interpolazione*, *filtraggio*, ecc.

Altra precisazione riguarda le modalità su come a partire dalle misurazioni sul processo si passi poi alla generazione dei pattern. Entrano qui in gioco le procedure di acquisizione e preprocessamento dei dati che saranno discusse nel successivo capitolo. A tale proposito, risulteranno anche chiare le ipotesi che permettono in questo contesto di affermare che lo spazio di ingresso X può essere sempre considerato normato, mentre così non è per il condominio in uscita Y.

1.3.2 Problema di classificazione (supervisionata)

Siano S_{tr} e S_{ts} due campionamenti distinti di un processo orientato $P : X \rightarrow L$, tali che $S_{tr} \cap S_{ts} = \varnothing$, dove X è uno spazio normato ed L è uno spazio non normato costituito da un insieme di etichette (*classi*). Il problema consiste nel sintetizzare un modello M di P, sulla base delle sole informazioni contenute in S_{tr}, tale che sia minimizzata un'opportuna funzione d'errore definita su S_{ts}. Anche in questo caso si vuole quindi massimizzare la capacità di generalizzazione del modello di classificazione.

E' essenziale notare che, per un problema di classificazione, lo spazio di uscita viene considerato privo di una norma perché non è possibile definirne alcuna, oppure perché rendere forzatamente normato il codominio di P potrebbe essere fuorviante ai fini del successo stesso del problema di modellamento. Questa notevole differenza, rispetto ai problemi di approssimazione funzionale, implica necessariamente che le funzioni di errore considerate devono essere di natura diversa. Ciò significa anche che è necessario adottare strategie di modellamento ben distinte.

Di fatto, l'assenza di una norma nel codominio di P rende il problema di classificazione più semplice da affrontare. Pertanto, sebbene tecnicamente possibile, non è mai opportuno utilizzare tecniche di approssimazione funzionale per risolvere problemi di classificazione. Ciò introdurrebbe anche un certo grado di arbitrarietà nella soluzione del problema, in quanto tale sarebbe l'assegnazione di un valore numerico (reale od intero) all'etichetta di classe. Quest'ultima infatti, quand'anche identificata da un numero, ha pur sempre un valore soltanto simbolico.

1.3.3 Problema di clustering

Dato un campionamento S di un processo non orientato P, il problema di clustering consiste nel determinare un modello M_P capace di raggruppare (*clusterizzare*) i pattern di S in opportuni sottoinsiemi (*cluster*) di S. I pattern appartenenti allo stesso cluster

devono possedere caratteristiche simili tra loro (*compattezza*) che, a loro volta, devono essere ben distinte (*separabilità*) da quelle dei pattern appartenenti ad altri cluster[1].

E' interessante notare come nei problemi di approssimazione funzionale e classificazione (supervisionata) il modello deve essere sintetizzato in modo da operare su tutto lo spazio di ingresso X, e quindi anche al di fuori del training set S_{tr} su cui esso è determinato. Nel caso del clustering, invece, il modello M_P consiste essenzialmente nella determinazione di un'opportuna metrica che misura il livello di similitudine dei soli pattern di S e non necessariamente di tutto lo spazio caratteristico di P. In ogni caso, anche nei modelli non supervisionati ad ogni pattern viene fornita una sorta di etichetta che ne indica la sua appartenenza ai vari cluster. In tal senso si può parlare di capacità di generalizzazione del modello di clustering, visto che tali etichette non sono comunque presenti nel campionamento del processo.

Se tuttavia, come già evidenziato dalla tassonomia di fig. 1.1, non ha molto senso considerare un problema di modellamento non supervisionato nel caso di spazi di uscita normati, può invece avere senso considerare un problema di classificazione non supervisionata. In tal caso, pur essendo possibile definire le etichette di uscita del processo, non è possibile utilizzare le stesse durante il modellamento. Bisogna quindi risolvere un problema di clustering in cui lo spazio caratteristico del processo di solito coincide con quello di ingresso del processo stesso. Sulla base dei cluster determinati si possono successivamente costruire, mediante opportune procedure che saranno discusse nei successivi capitoli, modelli di classificazione che operino su tutto lo spazio di ingresso X e che permettano la soluzione di un problema di classificazione. L'ipotesi fondamentale affinché sia risolvibile un problema di classificazione non supervisionata è che pattern di X che sono raggruppati nello stesso cluster corrispondano alla stessa etichetta di classe. Tale ipotesi non è evidentemente vera in generale in quanto, ad esempio, il campionamento del processo potrebbe essere tale che pattern vicini nello spazio di ingresso (cioè nello stesso cluster) siano in

[1] Come sarà discusso nel seguito, il concetto di appartenenza di un pattern ad un cluster può variare a seconda del tipo di logica utilizzata (crisp, fuzzy o probabilistica). Nel caso di logica crisp, ad esempio, un pattern può appartenere ad uno ed un solo cluster, per cui M_P coinciderebbe in questo caso con una partizione del campionamento S.

prossimità di una regione di transizione da una classe all'altra e quindi appartengano a classi differenti.

1.4 Sistemi di modellamento artificiali

Non è difficile immaginare che quando si parla di approccio biologico al modellamento ci si riferisce all'attività umana di classificazione, interpolazione e raggruppamento di oggetti con i meccanismi che sono propri del complesso cervello-organi di senso. Il modo in cui essi funzionano non è ancora del tutto noto e, comunque, i limiti di immaginazione spaziale del cervello non permettono di lavorare su insiemi di dati che abbiano complessità molto elevate. Caratteristica peculiare di questo approccio è la soggettività dei risultati: essi possono variare da soggetto a soggetto a causa delle diverse interpretazioni che si possono dare ai dati osservati e che sono dovute alle differenti esperienze soggettive, alle differenti formazioni culturali individuali ed ai differenti scopi che si hanno in mente.

L'evoluzione delle macchine di calcolo, l'aumento delle loro prestazioni e la diminuzione dei costi di produzione, ha però reso possibile la risoluzione di problemi di modellamento anche molto complessi in modo automatico, ad un costo minimo e con un rapporto costo-prestazioni estremamente elevato. E' in questo ambito che si parla di approccio artificiale al modellamento, ossia della possibilità di utilizzare sistemi di calcolo artificiale per la sintesi di un modello. E' evidente che un soggetto umano può risolvere in modo estremamente efficace problemi di modellamento in due o tre dimensioni ad un costo praticamente nullo, mentre un sistema artificiale può risolvere problemi su un numero qualunque di dimensioni dei dati di ingresso ad una frazione del tempo che impiegherebbe, se ci riuscisse, un uomo. Il problema è legato in questo caso alla qualità della soluzione ed al costo necessario per produrla. Su cosa stia dietro al differente comportamento delle macchine di calcolo da quello del cervello, e non solo nei problemi di modellamento, si tornerà nel seguito; per il costo si può sinteticamente notare che esso è legato a tre fattori fondamentali:

- A quanto la macchina è dedicata al problema. Vale a dire se si usa, ad esempio, un computer general purpose, un'architettura parallela, una rete neurale

artificiale, ecc. Di solito, un maggiore grado di specificità della macchina corrisponde ad una maggiore richiesta di velocità nel calcolo.

- A quanto la macchina deve essere precisa ed accurata. Bisogna quindi prestare cura nella progettazione e nella programmazione per evitare problemi di precisione, instabilità numerica, ecc.
- Al costo di progettazione. A titolo di esempio, basti pensare alla differenza che può intercorrere tra il costo dovuto alla programmazione effettuata nell'ambito di una ricerca scientifica, e quindi intrinseco al costo della ricerca stessa, ed il costo dovuto alla commissione dello stesso programma ad una software house privata.

Da un punto di vista funzionale è sempre bene considerare un sistema di modellamento data driven costituito da due oggetti fondamentali (fig. 1.2):

1. il modello M, ossia il sistema di calcolo avente il compito di determinare un'uscita (classe, cluster di appartenenza, valore di una funzione) in corrispondenza a determinati pattern di ingresso;
2. l'*algoritmo di apprendimento* (o di *training*) TA, ossia una successione di istruzioni tramite le quali viene sintetizzato il modello sulla base di un training set.

Definiamo come *sistema di modellamento* una coppia <M, TA>; specificamente, per problemi di approssimazione funzionale, di classificazione e di modellamento non supervisionato si parlerà, rispettivamente, di sistema di classificazione, sistema di approssimazione funzionale e sistema di modellamento non supervisionato.

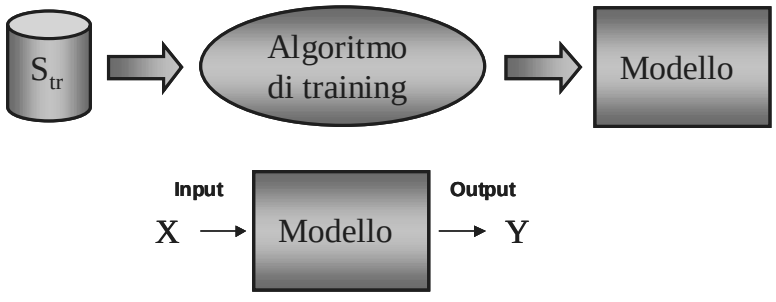


Figura 1.2 - Distinzione funzionale tra modello ed algoritmo di training.

La distinzione tra modello ed algoritmo di training è giustificata dal fatto che, in generale, una volta stabilito il tipo di modello da impiegare nella risoluzione di un particolare problema, è possibile scegliere un algoritmo di allenamento da un nutrito insieme di possibili candidati.

Il problema principale, nella progettazione di sistemi di modellamento artificiali, è pertanto quello di scegliere una classe di modelli adatta alla soluzione del problema in esame ed un corrispondente algoritmo di apprendimento. Il progettista/costruttore del sistema in questione potrebbe tentare un approccio di tipo analitico tendendo a trasferire alla macchina la propria conoscenza in merito al particolare problema, sotto forma di regole che la macchina deve seguire quando si trova di fronte a determinate situazioni. Un tale approccio si rivelerebbe ben presto inadeguato per diversi motivi: primo, il sistema così realizzato avrebbe una conoscenza al più uguale a quella del suo creatore e di tipo statico, cioè senza la possibilità di essere accresciuta; secondo, non sarebbe in grado di fronteggiare situazioni che non rientrano in alcun modo nelle categorie delineate dall'insieme di istruzioni che gli sono state impartite (impossibilità di deduzione); terzo, sarebbe necessaria una macchina del genere per ogni diverso problema da risolvere (struttura della macchina rigidamente vincolata al particolare problema); quarto, ma non ultimo per importanza, alcuni problemi che la macchina potrebbe essere chiamata a risolvere potrebbero avere una complessità tale che sarebbe impensabile tentare di risolverli con un approccio di tipo così rigido ed enumerativo.

Guardando invece a come il cervello umano costruisce le sue categorie mentali è immediatamente chiara la soluzione al problema: il sistema deve essere in grado di costruire in maniera autonoma i modelli astratti sulla base degli esempi concreti che ha avuto la possibilità di vedere (apprendimento). Inoltre, il sistema deve essere dotato di una qualche capacità che gli permetta di gestire anche le situazioni che non siano mai capitate prima (generalizzazione). La capacità di generalizzazione, ossia l'abilità di caratterizzare un pattern mai visto in fase di allenamento, è il requisito fondamentale di ogni modello. Tale capacità dipende essenzialmente dal tipo di inferenza induttiva adottata dal modello. Per meglio chiarire questo punto, nel seguito vengono definiti i principi di inferenza deduttiva ed induttiva.

Il ragionamento principe della deduzione è il seguente: *se supponiamo che tutti gli elementi di un certo insieme* A *soddisfino un determinato predicato, allora ne*

deduciamo che tutti gli elementi di un qualsiasi suo sottoinsieme B *soddisfano sicuramente anch'essi quel predicato* (se supponiamo che gli italiani siano amanti della musica, allora i toscani sono amanti della musica). La *deduzione*, attività logica per eccellenza, è viziata da un'insuperabile sterilità, giacché si limita a trarre tutte le possibili conseguenze da alcune premesse, e dunque a esplicitare quanto implicitamente già contenuto in esse. L'irrilevanza della deduzione nella ricerca e nella costruzione di nuova conoscenza dipende dal fatto che il processo deduttivo è un passaggio dal generale al particolare: una conseguenza della validità di una o più premesse non può che essere la validità di un caso meno generale. E' chiaro che un processo di costruzione di nuove conoscenze deve essere fondato proprio su un percorso in senso inverso (dal particolare al generale): *se supponiamo che tutti gli elementi di un certo sottoinsieme* B *di un insieme* A *soddisfino un determinato predicato, allora azzardiamo la tesi che tutti gli elementi dell'intero insieme* A *soddisfano anch'essi quel predicato.* Questo procedimento spericolato, questo ardito salto nel buio, si chiama *induzione*. L'induzione, sia pure a rischio di commettere errori, ci consente di acquisire nuove informazioni, come evidenziato in fig. 1.3.

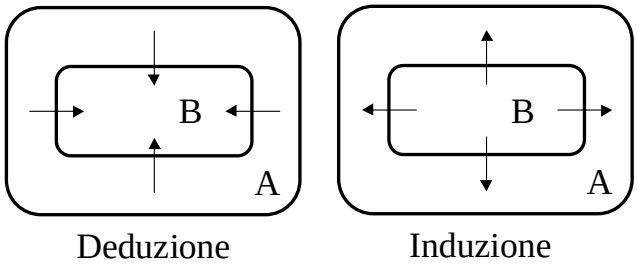


Figura 1.3 - Deduzione ed induzione.

Le definizioni di deduzione ed induzione appena esposte sono valide in un qualunque universo di riferimento. Se si opera in uno spazio normato X, essendo possibile definire una funzione distanza tra un insieme ed un punto esterno ad esso, la fastidiosa attitudine dell’inferenza induttiva a produrre grossolani errori (a “fare di tutta l’erba un fascio”) può essere alleviata da opportune scelte sulla metrica stabilita nello spazio X e dall’espressione adottata per la distanza di un punto da un insieme. Questa dipendenza, che sarà discussa nel par. 1.7.2 e nel cap. 2 insieme al problema

del preprocessamento dei dati, costituisce il punto cruciale di ogni sistema di modellamento che si basi su un principio di inferenza induttiva.

Chiarito cosa si intende in questo contesto per induzione e deduzione, possiamo evidenziare il fatto che mentre il modellamento analitico è una attività fondamentalmente deduttiva, il modellamento data driven (nella forma in cui lo si è definito) si basa su principi di inferenza induttiva; per questa ragione sarebbe molto più appropriato parlare, rispettivamente, di *modellamento deduttivo* ed *induttivo*.

L'inferenza induttiva appare fondamentale anche nell'ambito delle discipline legate allo studio dell'intelligenza artificiale. Di fatto, gli attuali calcolatori sono macchine deduttive eccellenti, dal momento che riescono ad effettuare in modo preciso e veloce ragionamenti deduttivi che possono anche essere molto complicati per un essere umano. Eppure nessuno è disposto ad affermare che i calcolatori siano dotati di intelligenza. Nonostante tutti i tentativi compiuti, la definizione di cosa di debba intendere per intelligenza è sempre una questione molto controversa, ma appare evidente come la capacità di compiere inferenze di tipo induttivo sia una caratteristica necessaria per attribuire ad un certo essere (biologico o artificiale) un qualche grado di intelligenza.

L'utilità e la necessità del meccanismo di inferenza induttiva rendono palesemente vantaggiosa la disponibilità di sistemi automatici di modellamento induttivo, giustificando così la ricerca di efficienti algoritmi di apprendimento.

1.5 Ottimizzazione strutturale dei sistemi di modellamento

Considerato un particolare problema di modellamento, si supponga di aver stabilito una misura di prestazione (*performance*) π. In generale, scelto un determinato modello, il suo funzionamento dipende da un certo insieme di parametri Ω, così che $\underline{y} = M(\underline{x}, \Omega)$, avendo indicato con M la relazione che il modello stesso stabilisce tra l'ingresso $\underline{x}$ e l'uscita ad esso attribuita. Dato un insieme di allenamento S_{tr}, il compito dell'algoritmo di training consiste nello stabilire una particolare determinazione $\overline{\Omega}$ di Ω, tale che la performance del particolare modello sintetizzato $\overline{M}(\underline{x}) = M(\underline{x}, \overline{\Omega})$ risulti massimizzata. $\overline{M}$ deve essere considerato, quindi, come una particolare istanza

del modello M. Una volta che $\overline{M}$ è stato generato, la performance π risulterà essere una funzione di $\overline{M}$ stesso e dell'insieme di test S_{ts}, ossia $\pi = \pi(\overline{M}, S_{ts})$.

A sua volta il comportamento dell'algoritmo di training TA dipende da un certo insieme di parametri di allenamento Θ, così che $\overline{M} = M(S_{tr}, TA(\Theta))$; pertanto, una volta stabiliti gli insiemi di allenamento e di test, nonché l'algoritmo di allenamento, la performance del sistema di modellamento <M, TA> dipende dai valori assunti dai parametri in Θ, cioè $\pi = \pi(\Theta)$.

Questi argomenti verranno opportunamente discussi ed ampliati nel seguito. Per il momento è importante sottolineare la dipendenza della performance dai parametri di allenamento. A causa di questa dipendenza, diverse determinazioni dell'insieme di parametri Θ comportano la sintesi di istanze distinte del modello, ciascuna caratterizzata da una certa performance. Il problema consiste nel fatto che, per definizione, i valori di Θ devono essere stabiliti prima che il processo di allenamento abbia luogo. Ciò significa che, se l'algoritmo di apprendimento considerato è caratterizzato da numerosi parametri e, soprattutto, se la performance risulta molto sensibile rispetto ad essi, la sintesi di un modello ottimo può diventare una lunga e complicata serie di tentativi alla ricerca di quella particolare determinazione di Θ che consente di generare il modello migliore.

Tali considerazioni portano a concludere che, sebbene la più importante caratteristica di un sistema di modellamento sia la sua capacità di generalizzazione, non meno importante è il suo *grado di automazione*. Vedremo in seguito che la definizione rigorosa di questa caratteristica comporta alcuni problemi. Per il momento è importante evidenziare il fatto che il grado di automazione di un sistema di modellamento dipende strettamente dal grado di automazione dell'algoritmo di allenamento, ma anche dal modello considerato, giacché alcuni modelli si prestano meglio di altri ad essere addestrati tramite algoritmi automatici.

La necessità di poter progettare sistemi di modellamento caratterizzati da un alto grado di automatismo scaturisce dalle seguenti considerazioni. Anzitutto, in un gran numero di applicazioni pratiche il test set può risultare, a causa della natura stessa del problema di modellamento considerato, quantitativamente e/o qualitativamente inadeguato o, addirittura, non disponibile. In questo caso, la performance del modello

sul test set non può essere considerata una stima attendibile dell'accuratezza che caratterizza il modello stesso sull'intero spazio caratteristico del processo. E' necessario allora riferirsi ad un qualche criterio tramite il quale sia possibile determinare a priori (ossia sulla base delle sole informazioni contenute nel training set) i valori dei parametri di training che consentano la sintesi di una "soluzione ragionevolmente accettabile". Definire in modo rigoroso cosa si debba intendere per "soluzione ragionevolmente accettabile" è una questione delicata, che merita un ulteriore approfondimento. Riprenderemo questa discussione nel secondo capitolo.

Come già accennato in precedenza, la sintesi del modello ottimo può portare ad una lunga fase di regolazione dei parametri di training, anche qualora si disponesse di un test set affidabile. Tale compito può rivelarsi complicato anche per utenti che vantano una certa esperienza. Questo problema non può più essere considerato marginale, qualora il sistema di modellamento fosse destinato ad essere utilizzato da un utente senza alcuna conoscenza sulle tecniche di modellamento. L'esigenza di rendere il più possibile automatica la procedura di generazione di un modello appare ancor più stringente qualora il sistema di modellamento stesso rappresentasse il nucleo di un dispositivo (hardware o software) più complesso, chiamato ad agire senza alcun supporto da parte dell'uomo. In questo caso, infatti, la regolazione dei parametri di training dovrebbe necessariamente avvenire in modo del tutto automatico, sulla base di un qualche criterio preselezionato.

In pratica, automatizzare un dato algoritmo di apprendimento, rispetto alla scelta di un suo parametro di allenamento $\theta \in \Theta$, significa stabilire un *criterio di selezione* in grado di valutare a priori l'idoneità di una soluzione tra le tante ottenute al variare del parametro considerato. Una volta stabilito il criterio di selezione, è necessario implementarlo in hardware o in software in modo che interagisca in modo efficiente con l'algoritmo di allenamento. Ciò non sempre è possibile in modo diretto; nella maggior parte dei casi l'implementazione di un criterio di ottimalità consiste in una qualche euristica in grado di selezionare una soluzione subottima. Infatti, nonostante la teoria dell'apprendimento suggerisca possibili criteri di ottimalità, la diretta implementazione degli stessi non è semplice e richiede di solito un costo computazionale notevole. Torneremo nel seguito su questo punto.

Le argomentazioni appena esposte portano ad affermare che la concreta fruibilità di un sistema di modellamento è un punto essenziale che dovrebbe sempre essere tenuto in considerazione nell'applicazione di tecniche di modellamento a problemi reali. Esiste un altro aspetto molto importante che merita di essere evidenziato. Si è detto che la relazione che un modello stabilisce tra un pattern d'ingresso e la sua uscita $\underline{y}=M(\underline{x})$ dipende da un certo insieme di parametri Ω. In effetti ciò è vero quando il modello è caratterizzato da una *complessità strutturale* costante, ossia quando la selezione del modello stesso fissa univocamente la natura e la cardinalità di Ω. Chiamiamo *modelli a struttura costante* i modelli che soddisfano questa proprietà. Ad esempio, dato un problema di approssimazione funzionale su uno spazio di ingresso ad n dimensioni, l'uso di un modello di tipo lineare (un iperpiano in $\Re^n$) stabilisce inequivocabilmente il numero di parametri (n+1, nell'ipotesi di considerare un sottospazio affine) ed ovviamente la loro natura (i coefficienti dell'iperpiano). In questo caso, il compito dell'algoritmo di apprendimento consiste nel risolvere un problema di regressione tramite una procedura di *ottimizzazione parametrica*. Esiste, però, una grande varietà di modelli per i quali ciò non si verifica. Quando la complessità strutturale del modello non è predeterminata (questo è il caso più interessante dal momento che a questo genere di modelli corrispondono i sistemi di modellamento più flessibili), il modello stesso è caratterizzato da un ulteriore insieme di *parametri strutturali* Σ. In questo caso diremo che il modello è *a struttura variabile*. L'atto di fissare una particolare determinazione $\overline{\Sigma}$ dei parametri strutturali, trasforma il relativo modello a struttura variabile M in un determinato modello $M_{\overline{\Sigma}}$ a struttura fissa, specificando, come già detto, la natura ed il numero dei suoi parametri. Pertanto, l'insieme dei parametri $\Omega_{\overline{\Sigma}}$ del modello $M_{\overline{\Sigma}}$ varia, evidentemente, al variare dei parametri strutturali, e dunque:

$$\underline{y}=M(\Sigma, \Omega_{\Sigma}, \underline{x})$$

La conseguenza di questo fatto è che, nonostante i parametri in Σ siano rigorosamente dei parametri del modello (che non devono essere confusi con i parametri di allenamento), questi, dal punto di vista del sistema di modellamento, giocano lo stesso ruolo dei parametri Θ. In tal caso, infatti, la performance del modello generato dipenderebbe tanto da Θ quanto da Σ, cioè $\pi= \pi(\Theta, \Sigma)$.

Un tipico esempio di modello a struttura variabile è la celebre rete neurale denominata Multi Layer Perceptron (MLP), che sarà brevemente introdotta nel par. 1.8. Nel MLP il numero di strati ed il numero di neuroni presenti in ciascuno strato costituiscono i parametri strutturali del modello. L'insieme Ω è costituito dai pesi delle connessioni tra neuroni, dalle soglie di attivazione e dai parametri che specificano la natura di ciascuna funzione di attivazione. Evidentemente, fintanto che non vengono fissati tali valori, la cardinalità e la struttura stessa dell'insieme Ω rimane indefinita. Se si utilizza un comune algoritmo di apprendimento, l'utente sarà costretto a scegliere a priori tanto i valori dei parametri Θ, quanto i valori dei parametri Σ.

Ai fini dell'automazione del modellamento, si tratta evidentemente di un'ulteriore complicazione. Diremo che un sistema di modellamento è *strutturalmente ottimizzato* se l'algoritmo di apprendimento adottato è in grado di determinare automaticamente la struttura ottima del modello. L'ottimizzazione strutturale consiste nel generare una serie di modelli e nel selezionare quello ottimo dal punto di vista della capacità di generalizzazione, secondo un criterio predeterminato. I principi generali della teoria dell'apprendimento [Haykin 1999, Martinelli 2001], stabiliscono che un modello è ottimo quando i suoi parametri minimizzano una funzione obiettivo costituita da due termini: il primo misura l'adeguatezza (performance) del modello rispetto agli esempi presenti nel training set; il secondo misura la complessità del modello risultante. La discussione sui funzionali utili a questo scopo sarà ampiamente ripresa nei capitoli successivi. Nella procedura di ottimizzazione strutturale, la sequenza di modelli può essere costruita secondo due tecniche alternative:

- ***Tecniche costruttive***. Viene generata una sequenza di modelli $\{M_i\}$ a complessità crescente. E' possibile, ma non necessario, che il modello M_{i+1} venga costruito a partire dal modello M_i, incrementandone la complessità strutturale mediante l'aggiunta di elementi del modello in grado di aumentare l'accuratezza del modello stesso sull'insieme di allenamento.
- ***Tecniche di potatura* (*pruning*).** A partire da un modello che soddisfa tutti i pattern dell'insieme di allenamento, si genera una successione di modelli $\{M_i\}$ caratterizzata da una complessità strutturale decrescente. Anche in questo caso è possibile, ma non necessario, che il modello M_{i+1} venga dedotto da M_i

eliminando alcuni elementi strutturali che comportano la minima perdita di prestazione del modello stesso sull'insieme di allenamento.

Nota. Come vedremo nel cap. 4, le procedure di ottimizzazione strutturale possono risultare intimamente legate a quelle di ottimizzazione parametrica per diversi motivi. Ad esempio, è questo il caso in cui alcuni parametri dell'algoritmo di apprendimento sono inizializzati in modo aleatorio. In questi casi, anche se la complessità strutturale è fissata, è necessario ottimizzare il sistema di modellamento per diverse inizializzazioni dei parametri. Tale procedura può evidentemente essere assimilata a quella di un'ottimizzazione strutturale in cui le determinazioni dei parametri strutturali Σ possano ripetersi [Panella 2000, Panella 2001(ter)].

1.6 I sistemi di ragionamento approssimato

Le prestazioni fondamentali di un sistema di modellamento sono, essenzialmente, la sua capacità di generalizzazione, il suo grado di automazione e la sua velocità, intesa come tempo di training. Per quanto riguarda quest'ultima, occorre sottolineare il fatto che il compito che un sistema di modellamento è chiamato a risolvere ha, generalmente, una considerevole complessità computazionale, specialmente quando la dimensione dello spazio di ingresso del processo da modellare è elevata. Da questo punto di vista, ciò che effettivamente rende fruibile una tecnica di modellamento è proprio la sua velocità, soprattutto rispetto alla capacità del cervello umano di risolvere problemi di clustering anche molto complessi sebbene in uno spazio dei dati a dimensione limitata. L'indubbio vantaggio che l'uomo in questi casi ancora possiede sulle macchine è dovuto, come già accennato, al suo modo di pensare e di calcolare ossia alla sua intelligenza o, meglio ancora, alle sue capacità induttive.

I problemi di modellamento sui dati del mondo reale sono spesso caratterizzati da una quantità elevata di pattern su cui operare, non necessitano di un'elaborazione estremamente accurata e bisogna estrarre da essi solo alcune caratteristiche significative (i cluster, ad esempio). Si ha quindi bisogno di sistemi approssimati, non completamente definiti, difficili da modellare e con uno spazio di soluzioni a larga

scala (che quindi lascia adito a diverse interpretazioni soggettive). Il ragionamento umano ed il suo sistema nervoso, nelle procedure di modellamento come quelle di clustering, sfruttano gli unici elementi rilevanti che si hanno a disposizione in questi casi: le informazioni a priori e le conoscenze empiriche sul sistema, oltre ai dati di ingresso e/o di uscita. Il complesso cervello-organi sensoriali, non solo dell'uomo ma anche di altri esseri del mondo animale, è quindi capace di manipolare in modo più efficiente di quanto ancora non riescano a fare le macchine tali imperfette informazioni: per essi si può parlare di *ragionamento approssimato*.

Nell'ambito del calcolo artificiale le prime a svilupparsi sono state le tecniche di modellamento ed i sistemi di calcolo basati sulla logica booleana, sui modelli analitici e sulla ricerca deterministica delle soluzioni; tali tecniche sono spesso riunite sotto il nome di Hard Computing (HC). I calcolatori che continuano a fare uso dell'HC devono essere in presenza di informazioni precise e complete sui sistemi da modellare e possono quindi usare metodi di ragionamento formale, come le dimostrazioni dei teoremi, per associare ai valori logici booleani lo stato o il comportamento di tali sistemi. L'HC è quindi particolarmente adatto in quei problemi computazionali che è possibile formalizzare in modo preciso e che richiedono una grande precisione numerica nel trattamento dell'informazione.

Non si deve però pensare ad una dicotomia esclusiva tra modelli approssimati per il calcolo biologico e modelli precisi per quello artificiale. Con il termine di Soft Computing (SC), secondo la definizione di Zadeh in [Zadeh 1994], si intendono tutte quelle discipline di calcolo che, contrariamente al tradizionale HC, sono tolleranti rispetto all'imprecisione, all'incertezza ed alla parziale verità dei dati a disposizione. Il SC racchiude quindi tutte quelle tecniche di calcolo che tentano di risolvere i complessi problemi del mondo reale, cercando di implementare sui sistemi di calcolo artificiali i metodi naturali di ragionamento approssimato o quantomeno di imitarne i principi di base.

In fig. 1.4 è riportata una possibile schematizzazione di HC e SC [Bonissone 1997]: si noti come, accanto al modo di ragionare che deriva dal tipo di modello adottato, ci sia anche la metodologia di ricerca della soluzione ottimale nello spazio delle possibili soluzioni di un problema. E' evidente che ad una maggiore approssimazione del modello, come nel caso del clustering, corrisponderà uno spazio delle soluzioni

maggiore e quindi la necessità di una ricerca che bilanci la qualità della soluzione con il maggiore costo computazionale derivante dal tempo necessario per provare più soluzioni possibili.

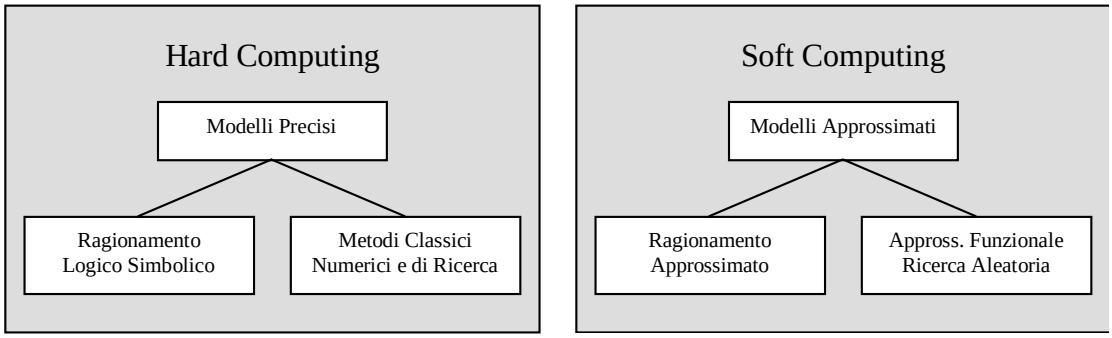


Figura 1.4 - Hard Computing e Soft Computing.

Le componenti essenziali del SC sono quattro: *ragionamento probabilistico* (PR), *logica fuzzy* (FL), *reti neurali* (NN) e *calcolo evoluzionistico* (EC). In fig. 1.5 è riportata la tassonomia essenziale del SC e dei suoi elementi i quali, è bene sottolinearlo, non sono affatto esclusivi l'uno dell'altro bensì ciascuno di essi offre dei metodi complementari di ragionamento e di ricerca della soluzione. Con ciò risulta evidente come il SC, nelle sue espressioni più evolute, utilizzi degli algoritmi ibridi (simbiosi) con diverse combinazioni di queste tecnologie. A titolo di esempio, si possono citare: la logica fuzzy per controllare le reti neurali o gli algoritmi genetici (GA), i quali sono parte del calcolo evoluzionistico; l'uso degli algoritmi genetici per l'apprendimento delle reti neurali o per accordare controllori che usano la logica fuzzy; l'implementazione di controllori in logica fuzzy accordati da reti neurali con opportuni algoritmi. E' importante notare che sistemi ibridi possono esistere anche a cavallo tra HC ed SC, si pensi all'uso della logica booleana come front-end del SC sugli attuali calcolatori, oppure alle tecniche di defuzzificazione o di deprobabilizzazione di cui si parlerà nei capitoli successivi.

Lo scopo di questa tesi è quello di proporre un nuovo paradigma di ottimizzazione strutturale per uno dei più importanti sistemi di ragionamento approssimato: le *reti neurofuzzy*. Nei successivi paragrafi saranno evidenziate le caratteristiche essenziali che rendono le reti neurali e la logica fuzzy particolarmente adatte alla soluzione dei

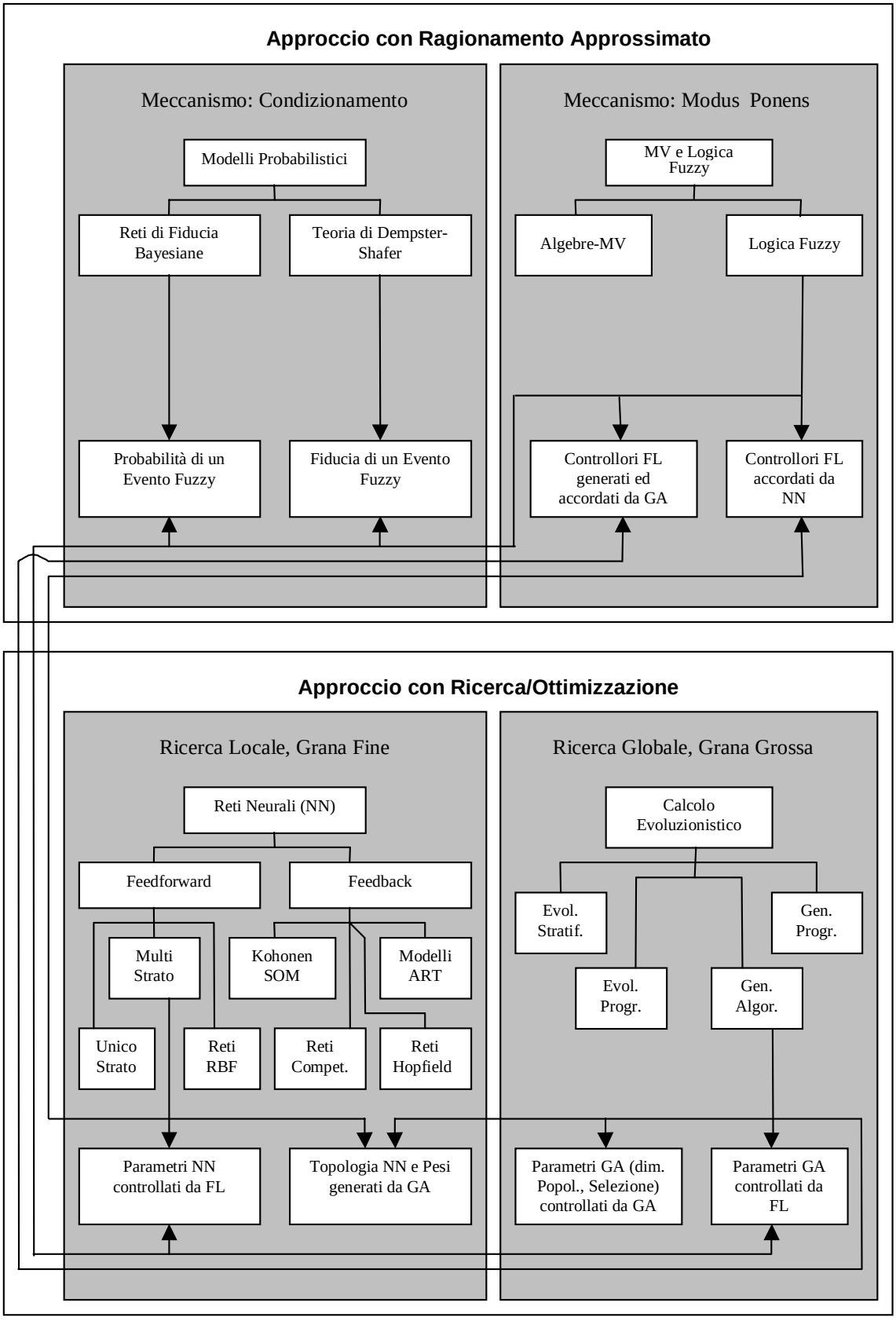


Figura 1.5 - Tassonomia del Soft Computing.

problemi di modellamento sui dati reali. Nel cap. 3 verranno invece introdotti i concetti generali relativi ai meccanismi di simbiosi tra reti neurali e logica fuzzy che permettono di generare le particolari reti neurofuzzy che saranno prese in considerazione.

1.7 La logica fuzzy nei problemi di modellamento induttivo

La pietra miliare nel campo degli studi sulla logica fuzzy è, come già stato detto, un articolo intitolato *"Fuzzy Sets"* pubblicato nel 1965 ad opera di Zadeh [Zadeh 1965]. In tale articolo l'autore elabora una vera e propria algebra degli insiemi fuzzy, comprensiva delle usuali operazioni di composizione degli insiemi tradizionali. Le operazioni sugli insiemi fuzzy sono un'estensione di quelle comunemente usate sugli insiemi tradizionali.

Nell'ambito dei sistemi di modellamento, i vantaggi che è possibile ottenere adottando la logica fuzzy sono essenzialmente i seguenti:

1. la capacità di trattare dati incerti;
2. l'adozione di un robusto e collaudato sistema di inferenza;
3. l'integrazione di conoscenze di tipo linguistico.

Nel seguito saranno discussi questi tre aspetti, mentre nel cap. 3 saranno discussi in dettaglio gli strumenti matematici caratteristici della logica fuzzy.

1.7.1 Incertezza fuzzy ed incertezza statistica

Alle radici della differenza tra logica tradizionale e logica fuzzy vi è quello che Aristotele chiamava "principio del terzo escluso". Nella tradizionale teoria degli insiemi, dato un oggetto ed un qualsiasi insieme, delle due affermazioni una sola è vera: "l'oggetto appartiene all'insieme" oppure "l'oggetto non vi appartiene". Nel caso della logica fuzzy, invece, il confine tra questi due opposti predicati non è così marcato e la verità di uno dei due non implica necessariamente la totale falsità dell'altro; in altre parole, si ammette l'esistenza di un'ulteriore possibilità, intermedia

tra le due, violando così il sopra menzionato principio, postulato cardine della logica classica (non a caso detta “aristotelica”).

Da un punto di vista più formale si può sostenere che data una relazione su un universo di oggetti, è immediatamente individuato su tale universo un insieme indotto dalla relazione stessa (l’insieme contenente tutti gli oggetti che la soddisfano). D’altra parte, la costruzione di un insieme di oggetti determina univocamente su tale universo l’esistenza di una relazione che scaturisce dall’individuazione degli elementi che compongono l’insieme. Un tale insieme è detto *convenzionale* o *crisp*. La polemica filosofico-matematica riguardo al fatto che nasca prima la relazione e da questa scaturisca l’insieme, o viceversa, è molto antica ed ancora irrisolta. Ambedue le scuole di pensiero però concordano necessariamente su uno stesso punto: la corrispondenza tra insiemi e relazioni è biunivoca. Grazie a questa biunivocità, per ogni dato insieme A contenuto in un universo X, una via alternativa per specificarne gli elementi che lo compongono è quella di definire sull’universo X una funzione, la cosiddetta *funzione caratteristica* dell’insieme o *membership function* (MF), a valori nell’insieme binario {0,1}, che valga 1 solo in caso di piena appartenenza:

$$\mu_A : X \to \{0,1\} \quad \text{t.c.} \quad \mu_A(x) = 1 \Leftrightarrow x \in A$$

Per esempio, sia A l’insieme dei numeri reali compresi tra -1 e 1: $X \equiv \Re$, $A = \{x \in \Re : -1 \le x \le 1\}$; in questo caso, la funzione caratteristica avrà espressione:

$$\mu_A(x) = \begin{cases} 1, & \text{se } -1 \le x \le 1 \\ 0, & \text{altrimenti} \end{cases}$$

Nel caso di insiemi fuzzy, invece, le MF degli insiemi non assumono solo i valori 0 ed 1, bensì tutti i valori reali compresi nell’intervallo [0, 1]. Questa è la differenza essenziale tra insiemi crisp e fuzzy e le sue implicazioni sono di portata rilevante: nel caso fuzzy, l’unica maniera per definire un insieme è quella di fornire l’espressione della sua MF. Non sarebbe infatti possibile enumerare gli elementi che lo compongono (nel caso di universo discreto) ovvero specificare un intervallo (nel caso di universo continuo), come è stato fatto per l’insieme A dell’esempio di sopra, per almeno due motivi: primo perché l’insieme fuzzy non è costituito solo dagli elementi che hanno appartenenza 1; inoltre, alcuni tipi di MF non valgono mai zero ed in quel caso l’insieme che una di esse rappresenta si estende, in misura più o meno marcata, su

tutto l'universo (nel senso cioè che non esiste alcun punto dell'universo che *non* appartiene in senso tradizionale all'insieme).

Vediamo un esempio di insieme fuzzy più nel dettaglio: l'insieme B dei numeri reali "prossimi allo zero". Come è intuibile, la specificazione di "essere prossimo a" è abbastanza imprecisa e suscettibile di interpretazioni soggettive. Del resto, le MF in grado di descrivere l'insieme in questione sono infinite e la scelta di una di esse non è univoca; sta a chi ne farà uso determinare quale scegliere tra le tante, in base al contesto ed all'applicazione. Questo fatto costituisce allo stesso tempo un punto di forza e di debolezza della logica fuzzy; l'unicità, tipica degli insiemi tradizionali, viene sacrificata a favore di una maggiore flessibilità e accuratezza di descrizione delle sfumature logiche. Gli unici vincoli che si possono dare a queste funzioni sono quelli di *normalità* (massimo pari ad 1), *monotonicità* (più si è vicini allo zero e più la funzione è grande) e di *simmetria*. Tornando all'esempio di sopra, una possibile scelta per la MF dell'insieme fuzzy B potrebbe essere la seguente:

$$\mu_B(x) = \begin{cases} 1-|x|, & \text{se } |x| \le 1 \\ 0, & \text{altrimenti} \end{cases}$$

In fig. 1.6 vengono riportati gli andamenti delle MF dei due esempi considerati. Si noti come la classe degli insiemi fuzzy racchiuda al suo interno, come caso particolare, anche gli insiemi tradizionali.

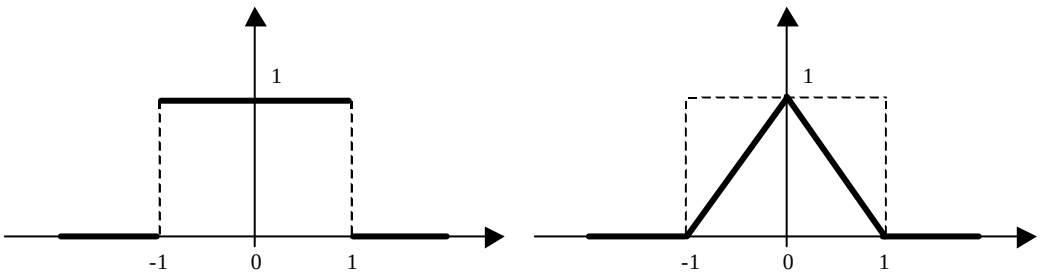


Figura 1.6 – Membership function per un insieme crisp ed uno fuzzy.

Le variabili x appartenenti all'universo X sono dette *variabili fuzzy* quando assumono come valore una *quantità fuzzy* definita, appunto, da un insieme fuzzy. Ogni quantità fuzzy, come ad esempio l'insieme B=B(X) introdotto nell'esempio precedente, è quindi definita tramite una MF $\mu_B(x)$, che esprime il livello di certezza riguardante il valore (fuzzy) B(X) assunto dalla variabile x. Risulta ora evidente come

l'incertezza fuzzy derivi dalla parziale appartenenza di un oggetto dell'universo X ad un certo insieme. Quando si parla di incertezza statistica, invece, i principi di inferenza logica rimangono quelli tradizionali. La probabilità associata ad un evento descriverà la tendenza, o meglio la frequenza, con la quale la variabile aleatoria tende ad appartenere all'insieme legato ad esso (appartenenza 1 ed evento vero) oppure no (appartenenza 0 ed evento falso). Quanto detto mostra che la logica fuzzy e la teoria delle probabilità modellano due tipi di incertezza ben distinti. Le appartenenze fuzzy rappresentano *aderenze di oggetti a proprietà non precisamente definite*, mentre le probabilità forniscono informazioni circa frequenze relative ad eventi.

I molti studi effettuati sul rapporto tra teoria della probabilità e logica fuzzy hanno sempre portato alla stessa conclusione riguardo alla complementarità di queste due teorie; come formalizzato da Zadeh in [Zadeh 1968], è possibile introdurre il concetto di *probabilità di un evento fuzzy*. Infatti, detto X l'universo dei possibili valori che può assumere una variabile aleatoria x, così come un sottoinsieme crisp A in X rappresenta un evento, allo stesso modo è possibile considerare come evento fuzzy un sottoinsieme fuzzy F di X: l'evento non sarà più solo vero o falso a seconda che x cada o no in A, bensì sarà anche vero con un certo grado di verità (al limite zero) pari al valore di appartenenza di x ad F. Ne consegue che, se si indica con f(·) la MF dell'insieme-evento fuzzy F e con p(x) la probabilità della generica variabile x in X, si definisce probabilità dell'evento fuzzy la grandezza:

$$P(F) = \sum_{x \in U} p(x) \cdot f(x)$$

1.7.2 Inferenza induttiva di tipo fuzzy e logiche non esclusive

Si è detto nel par. 1.4 che la caratteristica fondamentale di un sistema di modellamento è la sua capacità di generalizzazione e che questa dipende, essenzialmente, dal tipo di inferenza induttiva adottata, nonché dalla metrica definita sullo spazio di ingresso. Il problema tipico che si presenta in questi casi potrebbe essere il seguente:

> *sia* A *un qualunque insieme in un universo* X *e* B *un sottoinsieme di* A*; supponiamo che ogni elemento di* B *soddisfi un predicato* P*; si vuole indurre la verità del predicato* P *per ogni punto di* A.

Risulta evidente che, affidandosi ad una logica tradizionale (di tipo aristotelico), caratteristica del principio di induzione enunciato al par. 1.4, il predicato potrà essere vero oppure no per i punti di A. Un tale meccanismo di inferenza è privo di flessibilità e robustezza agli eventuali errori che possono essere commessi, cosicché il sistema di modellamento che ne deriva potrebbe essere caratterizzato da un'insoddisfacente capacità di generalizzazione.

Una maggiore flessibilità e robustezza al meccanismo di inferenza potrebbe essere fornita dall'utilizzo di una metrica, definita nello spazio normato X, che misuri la distanza $d_B(a)$ di ogni punto $a \in A$ dall'insieme B. In tali casi, l'inferenza induttiva potrebbe essere basata sulla seguente ipotesi:

> *per ogni elemento* $a \in A$*, il predicato* P *è vero in misura pari ad* $1/d_B(a)$.

L'implementazione di un tale principio risulta molto sensibile alle scelte fatte sul tipo di metrica e, nella pratica, non segue mai la definizione data, se non altro perché il reciproco di $d_B(a)$ potrebbe assumere valori singolari quando il punto a appartiene totalmente all'insieme B.

In questa classe possono rientrare anche quei meccanismi di inferenza che fanno uso di un'impostazione di tipo frequentistico e che, in questo contesto, saranno definiti come meccanismi di *inferenza probabilistica*. In tali casi, l'inferenza induttiva potrebbe essere basata sull'ipotesi che

> *all'insieme* B *viene associata una funzione di probabilità* $p_B(\cdot)$; *formuliamo quindi l'ipotesi che per ogni elemento* $a \in A$*, il predicato* P *sia vero con una probabilità* $p_B(a)$.

La flessibilità di un tale sistema di inferenza potrebbe, in prima istanza, risultare limitata. Infatti, come già accennato in precedenza, il predicato può risultare o del tutto vero o del tutto falso, anche se ciò accade in modo aleatorio sulla base di $p_B(a)$. Ciò significa, ad esempio, che se il predicato fosse effettivamente falso per il punto a, allora ogni 100 decisioni si commetterebbero (in media) $100 \cdot p_B(a)$ errori! Tuttavia,

come sarà evidenziato nel par. 2.5.1 e nei successivi capitoli, le probabilità (o funzioni affini come le densità di probabilità) possono essere utilizzate anche per misurare in modo raffinato la distanza di un punto da un insieme. Il vantaggio consiste nel fatto che esistono degli algoritmi che poggiano su solide basi teoriche e che permettono di stimare in modo efficiente tali funzioni di probabilità.

L'adozione della logica fuzzy consente invece un'implementazione immediata e molto elegante del principio di induzione basato su una distanza tra punto ed insieme. E' infatti possibile introdurre un meccanismo di inferenza induttiva fuzzy del seguente tipo:

> *all'insieme* B *viene associata una funzione di appartenenza* $\mu_B(\cdot)$ *che soddisfi le proprietà di normalità, monotonicità e simmetria; formuliamo quindi l'ipotesi che per ogni elemento* $a \in A$ *, il predicato* P *sia vero in misura pari a* $\mu_B(a)$.

Si noti che, per la proprietà di monotonicità, all'aumentare della distanza dell'elemento $a \in A$ dalla frontiera di B diminuisce il "grado di verità" del predicato P (fig. 1.7).

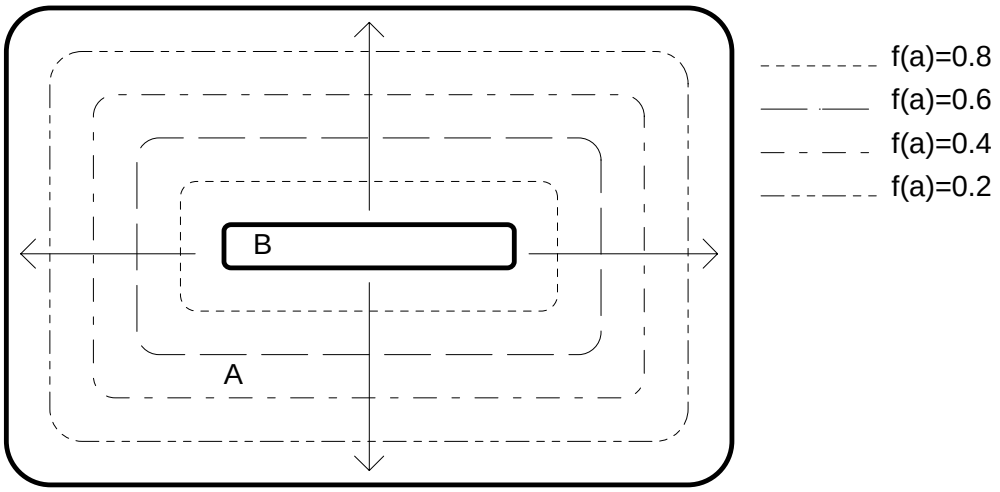


Figura 1.7 - Induzione fuzzy.

In pratica, sfruttando un substrato teorico consolidato, tale principio consente di definire un robusto meccanismo di induzione, simile a quello basato sul concetto distanza di un punto da un insieme in uno spazio normato. In ogni caso, apparire

chiaro quanto sia più conveniente adottare un principio di induzione di tipo fuzzy, piuttosto che affidarsi ad una logica tradizionale di tipo aristotelico. E' importante sottolineare il fatto che per poter definire la proprietà di monotonicità (cui deve necessariamente soddisfare la funzione μ_B) è necessario che l'universo X sia normato.

Si vuole, in conclusione, fornire un'ulteriore definizione per i sistemi di modellamento induttivo. Tali sistemi si diranno *esclusivi* (o *crisp*) se sono basati su una logica di inferenza di tipo classico (aristotelica); viceversa, si diranno *non esclusivi* se utilizzano un'inferenza basata su una metrica. In particolare, i sistemi di modellamento non esclusivi si diranno *probabilistici* se utilizzano un'inferenza probabilistica e *fuzzy* se utilizzano un'inferenza basata sugli insiemi fuzzy.

1.7.3 Integrazione di dati linguistici

Per alcune classi di problemi di modellamento supervisionato è possibile disporre di un campionamento del processo (ossia dati numerici di tipo ingresso-uscita che rappresentano il legame funzionale incognito da identificare) e di *informazioni linguistiche* fornite da esperti che, sulla base delle loro conoscenze sul sistema in esame, ne forniscono una descrizione ingresso-uscita di tipo qualitativo.

I valori fuzzy assunti da x, come ad esempio B(X) nell'esempio di fig. 1.6, sono spesso caratterizzati da etichette linguistiche. In tal modo, le variabili fuzzy possono essere utilizzate in sentenze del tipo "x è B" ("x is B") che esprimono alcune proprietà del processo sotto esame. In tal senso, l'universo X è anche detto *universo linguistico* oppure *universo del discorso*. Sempre con riferimento al caso di fig. 1.6, la variabile fuzzy x potrebbe essere denotata da etichette linguistiche come "molto superiore a dieci", "quasi negativa", "prossima allo zero", e così via. La MF di fig. 1.6 definisce proprio la quantità fuzzy denotata con "prossima allo zero"; in altre parole, una tale MF definisce in modo fuzzy quali sono i valori di x che devono essere considerati quando "si è prossimi allo zero".

Generalmente, la creazione di un dato linguistico è il frutto di una analisi di dati oggettivi e soggettivi e di relazioni più o meno empiriche associate al contesto sotto esame. L'estrazione di regole fuzzy, a partire da un set di dati linguistici, consiste

semplicemente nel tradurre questi ultimi in termini di relazioni di implicazione tra insiemi fuzzy, con funzioni di appartenenza specificate.

Le informazioni di tipo linguistico assumono una notevole importanza qualora l'insieme di allenamento S_{tr} di cui si dispone è incompleto (perché ad esempio non copre tutto lo spazio degli ingressi), ma anche quando non si vuole o non si può attribuire ad S_{tr} la massima affidabilità (perché le misure sono incerte o rumorose). A titolo di esempio, si immagini che l'obiettivo del modellamento sia quello di approssimare una funzione $f(x_1,x_2)$ incognita di cui si conosca, oltre ad alcuni esempi di coppie ingresso-uscita, anche una descrizione approssimativa del tipo:

$$f(x_1,x_2) \begin{cases} \text{intorno a 0.5 per } x_1 \text{ e } x_2 \text{ concordi in segno, e in modulo superiori a} \\ \text{intorno a } -0.5 \text{ per } x_1 \text{ e } x_2 \text{ discordi in segno, e in modulo superiori} \\ \text{circa nulla per } x_1 \text{ circa nulla} \\ \text{circa nulla per } x_2 \text{ circa nulla} \end{cases}$$

Una funzione che ben si accorda con tale descrizione è la seguente:

$$y = \frac{x_1 x_2}{1 + x_1^2 + x_2^2}$$

E' possibile formalizzare la descrizione qualitativa del contesto in termini di regole linguistiche; si ottengono così le seguenti regole:

$R^{(1)}$: se x_1 è superiore a 3 e x_2 è superiore a 3, y è circa 0.5

$R^{(2)}$: se x_1 è inferiore a -3 e x_2 è inferiore a -3, y è circa 0.5

$R^{(3)}$: se x_1 è superiore a 3 e x_2 è inferiore a -3, y è circa -0.5

$R^{(4)}$: se x_1 è inferiore a -3 e x_2 è superiore a 3, y è circa -0.5

$R^{(5)}$: se x_1 è circa nulla e x_2 è qualunque, y è circa nulla

$R^{(6)}$: se x_1 è qualunque e x_2 è circa nulla, y è circa nulla

Di conseguenza, dovranno essere individuate le MF che possono essere associate a ciascuna quantità fuzzy (x_1 è superiore a 3, x_2 è inferiore a -3, y è circa 0.5, ecc.).

Il problema di come utilizzare due classi così disomogenee di dati (linguistici e numerici), per strutturare ed addestrare una rete adattativa di tipo neurale al compito prefissato, ha trovato nei sistemi neurofuzzy una soluzione promettente e foriera di interessanti sviluppi. Due notevoli modelli di approssimazione funzionale, caratterizzati dalla possibilità di integrare regole linguistiche, sono i modelli FBF [Hartman 1990, Kosko 1994, Light 1992, Park 1991, Poggio 1990] ed ANFIS [Jang 1997] che saranno discussi nel cap. 3.

1.8 Le reti neurali come modelli computazionali avanzati

Sebbene la logica fuzzy risulti utile nei procedimenti induttivi capaci di estrarre nuove conoscenze dai dati a disposizione, le capacità induttive prescindono dall'utilizzo o meno della stessa. In realtà la logica fuzzy permette solo di traslare la conoscenza qualitativa di un problema da risolvere in sistemi di ragionamento capaci di effettuare un controllo o un'interpolazione sui dati; essa quindi non possiede le caratteristiche di adattatività e di apprendimento necessarie ai suddetti processi induttivi, piuttosto ne costituisce un valido, ed a volte fondamentale, complemento.

Le reti neurali (artificiali) sono invece le tipiche strutture computazionali adattative in grado di apprendere uno specifico compito mediante una procedura di modellamento data driven. Esse possono essere definite come modelli computazionali (nel senso inteso al par. 1.2) costituiti da "un insieme di processori elementari collegati tra loro secondo una data matrice di connessione".

E' bene premettere che le reti neurali non sono affatto l'unico strumento a disposizione per la progettazione di sistemi di inferenza induttiva. Tuttavia, esse hanno il pregio di poter essere direttamente implementate su sistemi di calcolo paralleli, in modo da ottenere un'implementazione molto più efficiente rispetto ad altre tecniche alternative. Il fatto che una rete neurale nasca, per definizione, come un'opportuna connessione di un insieme di processori elementari rende pressoché immediata la sua interpretazione circuitale. E' possibile pertanto assegnare a ciascun processore un distinto compito su una macchina parallela, oppure realizzare un circuito dedicato.

Ovviamente, neanche questa caratteristica di essere strutture parallele è una prerogativa esclusiva delle reti neurali. E' possibile infatti progettare sistemi di modellamento paralleli, senza alcuna necessità di fare riferimento alle reti neurali, biologiche o artificiali che siano. Esiste tuttavia una valida ragione nell'abbracciare l'approccio connessionista del paradigma neurale. Il sistema di modellamento più efficiente e flessibile che conosciamo è il cervello biologico; evidentemente, il funzionamento di questa autentica meraviglia dell'universo si basa su una struttura costituita da neuroni collegati in reti più o meno articolate. Ovviamente la complessità stessa del cervello va ben oltre i semplici meccanismi di connessione dei neuroni e, nonostante i notevoli progressi compiuti [Kandel 1994], i neurofisiologi sanno davvero ancora molto poco dei principi ultimi su cui si basa il funzionamento del cervello. Non per niente il cervello ha resistito ad ogni tentativo di analisi: si tratta del più complesso circuito conosciuto. Si stima infatti che in un volume di circa 1.5 litri siano stipati 10^{11} neuroni e da 10^{12} a $5 \cdot 10^{12}$ cellule gliali, le quali, oltre ad avere funzioni di supporto, hanno anche il compito di guidare, nel corso dello sviluppo, la migrazione dei neuroni e di dirigere la crescita dei loro assoni. Inoltre, le ramificazioni di ogni assone possono instaurare dei contatti sinaptici con un gran numero di altri neuroni (fino a 1000!).

Evidentemente non esiste alcuna tecnologia in grado di realizzare modelli così massicciamente paralleli in uno spazio paragonabile a quello occupato dal cervello (la stessa *Connection Machine*, con i suoi 65536 processori è ben poca cosa rispetto alla complessità del cervello). Addirittura Penrose [Penrose 1996] sostiene che lo stesso citoscheletro dei neuroni sia responsabile di fenomeni fondamentali per il funzionamento stesso del cervello e che tali fenomeni fisici (di natura quantistica) siano non computabili, e dunque non riproducibili su qualsiasi macchina di calcolo esistente!

Nonostante questo, tale linea di ricerca mantiene una grandissima importanza, tanto per quel che riguarda il progresso nelle conoscenze di neurofisiologia, quanto per lo studio teorico del funzionamento degli algoritmi di calcolo paralleli e per quella nuovissima branca della scienza dell'informazione denominata "Artificial Life". Del resto, il poter replicare almeno alcune delle meravigliose abilità del cervello e dei sistemi sensoriali biologici su un dispositivo artificiale è certamente una delle sfide

più esaltanti che possa intraprendere la futura ricerca scientifica, oltre che un importante traguardo tecnologico.

E' proprio nell'ottica appena delineata che è possibile dare una definizione operativa di rete neurale che, in qualche modo, renda giustizia alle favorevoli caratteristiche possedute da tali modelli. Ciò è tanto più necessario se si considera la crescente tendenza nella letteratura tecnica a definire come rete neurale un qualsiasi sistema di modellamento data driven. In effetti, sarebbe opportuno definire come rete neurale "*un sistema di modellamento data driven caratterizzato da un intrinseco parallelismo della sua struttura ed il cui funzionamento sia caratterizzato, dal punto di vista strutturale e/o funzionale, da un'imitazione delle capacità di inferenza induttiva espresse dai sistemi biologici naturali*".

La struttura più tipica di una rete neurale, forse la prima presentata, è quella del già citato MLP. Questa rete neurale è composta da un insieme di p processori elementari, chiamati *neuroni* in analogia con gli equivalenti biologici, i quali sono organizzati a *strati* e sono connessi tra loro secondo una certa matrice di connessione $\underline{A}$. Un certo sottoinsieme di tali neuroni, di cardinalità n, viene detto *strato di ingresso:* tale strato (in realtà fittizio) ha come unico compito quello di prelevare gli ingressi alla rete e di smistarli a tutti i neuroni dello strato successivo. Un ulteriore sottoinsieme, di cardinalità m<p, viene denominato *strato di uscita*: esso ha il compito di fornire all'esterno, in modo da poterle utilizzare, le m uscite della rete; tutti i rimanenti processori risultano posizionati in uno o più strati intermedi denominati generalmente *strati nascosti*. Ciascuna connessione tra neuroni è in generale caratterizzata da uno o più parametri detti *pesi* appartenenti ad un insieme W. Un altro insieme di parametri significativi, che si denoterà con N, è l'insieme dei parametri che regola il comportamento dei singoli processori elementari. Si considera inoltre un ulteriore collezione F di p funzioni, ciascuna caratterizzante in maniera univoca la relazione ingresso-uscita in corrispondenza al singolo neurone.

Quindi, dal punto di vista dei singoli neuroni di un generico strato s (o, come si dice, della *micro-struttura* della rete), ciascuno di essi realizza una funzione, in generale non lineare, avente n ingressi $\underline{x} = [x_1\ x_2 \ldots x_n]$ provenienti da n neuroni dello strato s-1 ed un'uscita scalare y destinata ai neuroni dello strato s+1; alla i-esima

connessione in ingresso è associato un peso reale $w_i \in W$ che modula in maniera lineare il contributo dell'ingresso corrispondente. La combinazione lineare degli ingressi con i rispettivi pesi, chiamata *attivazione* o *eccitazione* viene "elaborata" da una funzione $f \in F$, detta *funzione di trasferimento* o *di sagomatura* del neurone e viene restituita in uscita. L'eventuale contributo di non linearità fornito dal neurone all'uscita è evidentemente causato dalla presenza della f(·). Complessivamente, l'uscita in corrispondenza al singolo neurone viene calcolata secondo la seguente espressione:

$$y = f(\sum_{i=1}^{n} w_i \cdot x_i)$$

Nella successiva fig. 1.7 sono riportati alcuni degli andamenti usuali delle funzioni di sagomatura. Molte volte il funzionamento del neurone è influenzato anche da un ulteriore parametro di *polarizzazione* $\eta \in N$, che si somma all'attivazione ed ha il compito di traslare l'asse di simmetria della funzione di trasferimento, cioè:

$$y = f(\sum_{i=1}^{n} w_i \cdot x_i + \eta)$$

Il valore f(η) rappresenta l'uscita in corrispondenza ad attivazione nulla del neurone.

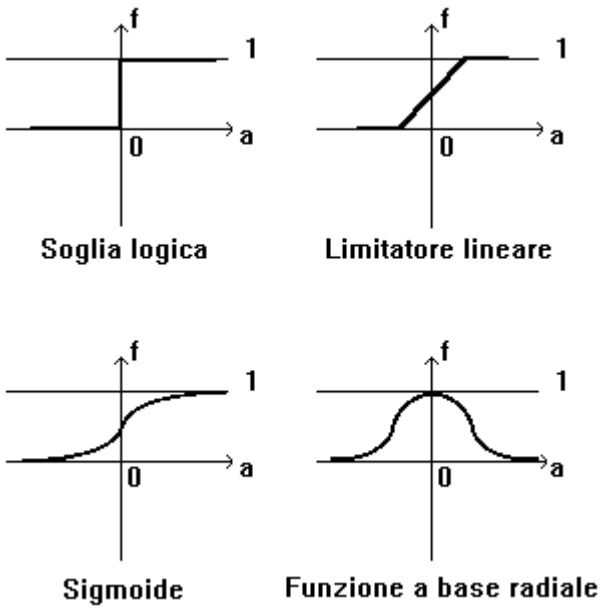


Figura 1.7 - Andamento di alcune tipiche funzioni di trasferimento.

In pratica, il MLP realizza una mappatura (generalmente non lineare) tra lo spazio di ingresso $X \equiv \Re^n$ e lo spazio di uscita $Y \equiv \Re^m$; essa dipende da tutti gli elementi prima menzionati:

$$\underline{y} = G(\underline{x}, W, N, F) \quad \text{con} \quad \underline{x} \in X,\ \underline{y} \in Y$$

La struttura di un sistema di questo tipo è intrinsecamente parallela e ben si presta ad essere implementata su sistemi multiprocessore; tuttavia, nella maggior parte dei casi e tranne in alcune eccezioni (perlopiù nei sistemi dedicati), tali architetture vengono simulate a livello software su strutture monoprocessore e quindi implementate in maniera sequenziale. In tal modo, le reti neurali non beneficiano di quello che è il loro maggior vantaggio rispetto agli strumenti di calcolo più convenzionali usualmente di tipo sequenziale.

Sempre nell'ambito della topologia delle reti è possibile individuare diversi criteri di classificazione delle stesse, uno dei quali fa riferimento alla struttura delle connessioni. Se il grafo di tali connessioni tra neuroni non contiene cicli, allora la corrispondenza $G(\cdot)$ ingresso-uscita sarà di tipo statico: il valore che le uscite della rete assumeranno ad un certo istante t dipenderanno esclusivamente dagli ingressi presentati all'istante $t-\Delta t$, essendo Δt il ritardo di propagazione attraverso la rete stessa (sono le cosiddette reti *feedforward*, in cui l'informazione viaggia in un unico verso, quello ingresso-uscita). Tipico esempio di algoritmo di apprendimento per questa classe di reti è l'algoritmo EBP (Error Back Propagation) [Martinelli 2001]. In contrapposizione, esistono reti capaci di fornire sequenze di valori in uscita in corrispondenza di determinate sequenze di valori di ingresso: il presentarsi di un particolare pattern all'ingresso della rete ad un dato istante non basta da solo a decidere il tipo di uscita fornita; quest'ultima infatti dipenderà anche dalla sua storia passata (ossia dallo stato della rete definito dai pattern presentati in ingresso negli istanti precedenti). Le reti di questo tipo sono dette *dinamiche* o *ricorrenti* e trovano maggiore applicazione nei sistemi adattativi di controllo automatico e nei problemi di predizione.

Da un punto di vista strettamente computazionale, si può riassumere che le reti neurali sono dei sistemi di calcolo capaci di esplorare in modo locale e raffinato (o, come si dice, “a grana fine”) le possibili soluzioni ad un problema di modellamento, in modo da risolverlo tramite una procedura di ottimizzazione più o meno esplicita[2].

[2] Per ottimizzazione esplicita si intende la minimizzazione di una funzione di errore nel caso supervisionato come, ad esempio, accade nella EBP.

Capitolo
2

IL PROBLEMA DELLA RAPPRESENTAZIONE DEI DATI

2.1 Premessa

Il requisito fondamentale per una soddisfacente realizzazione di un sistema di modellamento è la stima dei fattori di base richiesti per rappresentare i dati da analizzare. Gli algoritmi di apprendimento, e la bontà dei risultati da essi prodotti, sono legati al tipo di dati in questione. In questo capitolo saranno fornite le nozioni fondamentali riguardanti i possibili modi in cui i dati possono essere organizzati; inoltre, sarà dato un breve cenno alle tecniche più comuni di trattamento degli stessi, sia in fase di ingresso che in fase di uscita, al fine di migliorare le prestazioni dei sistemi di modellamento. Lo scopo è quello di riunire alcuni argomenti che rivestono un ruolo di primaria importanza, tanto per i problemi di modellamento supervisionati quanto per quelli non supervisionati. Nel primo capitolo si è accennato al fatto che la capacità di generalizzazione di un sistema di modellamento dipende essenzialmente dal tipo di inferenza induttiva adottata. Al fine di definire una robusta modalità di induzione è necessario dotare lo spazio caratteristico del processo di una opportuna metrica. Per questa ragione, la corretta soluzione di un problema di modellamento data driven non può prescindere dall'adozione di un'adeguata funzione di distanza, così come è necessario scegliere opportunamente il tipo di preprocessamento cui sottoporre i dati a disposizione. Nel presente capitolo verranno affrontati tutti questi problemi, anche se la trattazione non potrà e non vorrà essere esaustiva a tale riguardo. Il punto

essenziale sarà mostrare che la distanza ed il preprocessamento dei dati influenzano in modo sostanziale l'inferenza induttiva utilizzata dal sistema di modellamento.

2.2 Ancora sull'uso delle metriche nei principi di induzione

Nel par. 1.7.2 è stato definito un principio di inferenza induttiva su spazi normati e si è visto come l'introduzione della logica fuzzy, e dunque del concetto stesso di funzione di appartenenza, possa formalizzare in modo efficace il suddetto principio di induzione. La scelta della norma, ovvero della membership fuzzy (MF) adottata nello spazio caratteristico del processo, è fondamentale per la corretta applicazione dell'inferenza induttiva. A tale proposito, si consideri un problema di classificazione il cui training set S_{tr} è mostrato in fig. 2.1 tramite uno scatter plot. Ci si chiede a quale delle due classi presenti ("nera" o "grigia") dovrebbe essere assegnato il vettore $\underline{x}$, che si suppone non appartenere ad S_{tr}. La risposta più intuitiva consiste nell'attribuire ad $\underline{x}$ l'etichetta "grigia"; in effetti il cluster "grigio" (cioè l'insieme dei punti etichettati come grigi) è "più vicino" ad $\underline{x}$ di quanto non lo sia il cluster "nero". Si badi bene che si giunge a questa conclusione, seppur in modo intuitivo, tramite un meccanismo di inferenza induttiva, assumendo implicitamente di dotare il piano cartesiano (in cui i punti sono evidentemente definiti) della comune norma euclidea. E' però evidente che se si considerasse una norma differente la decisione presa potrebbe risultare clamorosamente errata. Nonostante la notevole importanza della scelta di una metrica adeguata, questa non è sufficiente per poter definire un principio di induzione.

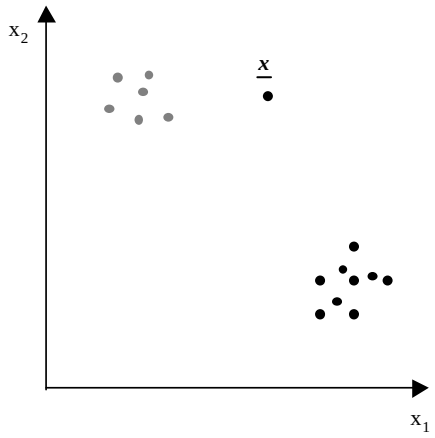


Figura 2.1 - Quale classe dovrebbe essere attribuita al punto $\underline{x}$?

Certamente la capacità di generalizzazione di un sistema di modellamento (sia esso un sistema di classificazione, approssimazione funzionale, ma anche un sistema di modellamento non supervisionato) è una questione di inferenza induttiva, e dunque di distanze. In effetti, il modellamento di un processo P a partire da un suo campionamento consiste nell'assegnare a ciascun punto dello spazio caratteristico di P un'opportuna uscita (un'etichetta, ad esempio, nel caso del problema di classificazione). Da un punto di vista geometrico, ciò significa definire nello spazio di P un certo numero di *regioni d'influenza* (o *di decisione*) e dotare queste ultime di un meccanismo di inferenza induttiva tramite il quale valutare il grado di appartenenza di un generico punto rispetto ad esse. Tali regioni determinano una sorta di *partizionamento* dello spazio d'ingresso o dello spazio caratteristico del processo, a seconda del problema di modellamento affrontato. Tali regioni d'influenza stabiliscono dei *modelli locali*, la combinazione dei quali determina il modello complessivo. In questo modo è possibile determinare modelli globali comunque complessi sulla base di modelli locali relativamente semplici. Tale meccanismo sarà ampiamente discusso nei capitoli successivi, quando si discuterà l'utilizzo delle reti neurofuzzy come sistemi di approssimazione funzionale.

E' comunque chiaro che in ogni problema di modellamento induttivo si ha la necessità di definire ed individuare delle regioni di influenza e che, per ottenere ciò, non è sufficiente stabilire una metrica. In effetti, si rende necessario:

1. definire cosa si intenda per regione di influenza, stabilendo un apposito meccanismo di induzione;
2. individuare numero, forma e dimensioni delle regioni di influenza a partire dall'insieme di campioni del training set.

Evidentemente, il compito descritto al punto 2 è di competenza dell'algoritmo di allenamento, mentre il punto 1 stabilisce in sostanza con quale classe di modelli si intende risolvere il problema. Sebbene la capacità di generalizzazione di un sistema di modellamento dipenda da entrambi i succitati punti, il progetto di un algoritmo di apprendimento è determinato da quanto stabilito circa le regioni di influenza. Pertanto, ai fini di una soddisfacente soluzione di un problema di modellamento induttivo, il primo punto è il più importante.

Si è detto che il principio di induzione dipende dalla particolare funzione adottata per misurare la distanza tra un punto ed un generico insieme. Per poter definire una tale funzione è necessario anzitutto specificare in che modo viene rappresentato un insieme, ossia una regione di influenza dello spazio caratteristico del processo. A titolo di esempio, si consideri l'insieme chiuso e limitato B in fig. 2.2; tale insieme può essere rappresentato tramite un suo punto caratteristico (come il baricentro c), ma anche mediante un inviluppo (il trapezio stesso che lo delimita). Solo dopo aver definito come si intende rappresentare un insieme è possibile stabilire una funzione che misuri la distanza di un punto a da esso e dunque il grado di appartenenza di a all'insieme considerato. Sempre con riferimento alla fig. 2.2, fissata ad esempio la comune norma euclidea, la distanza tra il punto a e l'insieme B assume valori differenti a seconda che si consideri la distanza di a rispetto a c o rispetto all'inviluppo di B, assumendo che in questo ultimo caso essa corrisponda alla distanza minima di a dal luogo geometrico dei punti che delimita B.

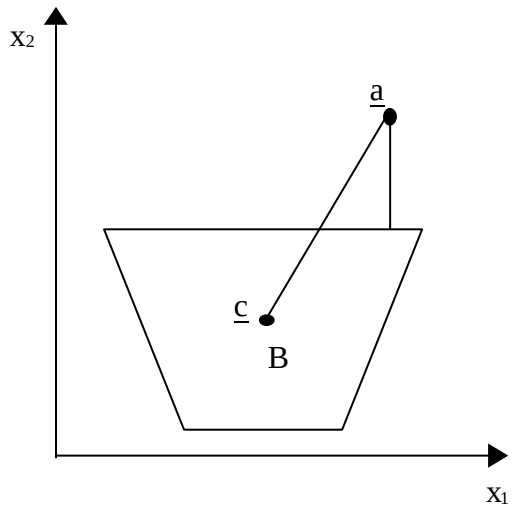


Figura 2.2 - Due possibili definizioni di distanza tra un punto ed un insieme.

Riassumendo quanto detto, si può dire che, al fine di definire un principio di induzione, risulta necessario:

- adottare una metrica nello spazio caratteristico del processo e quindi una funzione distanza punto-punto;
- definire come si intende rappresentare una regione di influenza;

- stabilire una funzione distanza punto-insieme, sulla base di quanto deciso in precedenza.

Con tali premesse è possibile adottare il principio di induzione su spazi normati descritto nel par. 1.7.2. Riguardo agli ultimi due punti, si è già ampiamente discusso su come l'utilizzo di un opportuno meccanismo di inferenza, come quello probabilistico o fuzzy, aiuti a definire in modo diretto la forma delle regioni di influenza ed il tipo di distanza dei punti da esse. Resta quindi da affrontare il problema della definizione di una distanza tra i vari pattern definiti nello spazio in cui il processo è modellato. A tale proposito, saranno introdotte nei paragrafi successivi le varie modalità con cui i pattern possono essere rappresentati in un sistema di modellamento.

2.3 Matrice dei pattern e matrice di affinità

Un insieme di pattern derivanti dal campionamento di un processo è di solito rappresentato da una matrice di dati "grezzi", ossia dati che sono stati nominati e raccolti durante l'osservazione del processo stesso e che non hanno ancora subito alcun tipo di trattamento. Tali dati possono essere disposti secondo due formati standard ben precisi:

- Matrice dei Pattern $\underline{P} \in M_{p,n}(\Re)$;
- Matrice di Affinità $\underline{D} \in M_{p,p}(\Re)$;

dove $M_{a,b}(\Re)$ è lo spazio delle matrici di a righe e b colonne ad elementi reali.

Nel primo caso si parla anche di *dati oggettivi* rappresentati dalle p righe di $\underline{P}$, che altro non sono se non i vari pattern a disposizione[1]. Ciascun pattern è descritto da un vettore di n numeri reali i quali rappresentano il valore delle *caratteristiche* (o *feature*) del pattern stesso. Tali caratteristiche, una per ogni colonna di $\underline{P}$, sono di solito

[1] Si faccia attenzione a non confondere con la matrice dei pattern $\underline{P}$ ciò che fino ad ora è stato indicato come generico processo P.

rappresentate come un sistema di assi ortogonali in uno spazio reale euclideo *E* n-dimensionale detto *spazio dei pattern*; tale spazio risulta normato ed isomorfo allo spazio vettoriale $\Re^n$ delle n-ple reali. Risulta quindi evidente la possibilità di interpretare le feature di un pattern come le coordinate di un punto (il pattern stesso) in uno spazio vettoriale di cui si cercheranno di sfruttare le proprietà geometriche[2].

Il generico elemento $\underline{P}(i,j)$ della matrice dei pattern rappresenta la feature j-esima del pattern i-esimo. Per comprendere meglio cosa si intenda per pattern e relative feature, si può pensare al seguente esempio: si supponga di voler suddividere in gruppi (cluster) i pazienti di un ospedale, in base ai risultati di alcune analisi cui sono stati sottoposti. Ogni paziente può essere considerato come un pattern le cui informazioni possono essere immagazzinate su una riga di $\underline{P}$; le risposte delle analisi effettuate su ciascun paziente rappresentano le informazioni disponibili sull'individuo: i valori di tali risposte vengono immagazzinati, ognuno su una colonna diversa, sulla riga relativa al paziente in questione. Per poter fare una ragionevole suddivisione dei pazienti, sulla base delle informazioni ottenute, è necessario che ogni individuo venga sottoposto allo stesso tipo di analisi e che i risultati delle diverse analisi effettuate vengano immagazzinati sempre nello stesso ordine. In altre parole, la colonna j-esima di $\underline{P}$ contiene i valori di una stessa feature (delle risposte alla stessa analisi) per tutti i pattern (pazienti) analizzati. Le feature sono quindi dei valori che derivano dalle preventive procedure di misura, stima, risposta, o quant'altro, realizzati per ciascun pattern che si vuole analizzare. E' ora anche chiaro il ruolo del clustering quando si vogliono cercare delle *similitudini tra oggetti caratterizzati da un certo numero di misure*: "esso permette l'organizzazione di questi dati multidimensionali laddove falliscono le capacità di percezione ed organizzazione spaziali umane, limitate al massimo a tre dimensioni".

Nel caso di problemi di apprendimento non supervisionato, e quindi di clustering, un cluster può essere definito come un raggruppamento di pattern che presentano delle

[2] In realtà, ricordando quanto detto nel cap. 1, i problemi di modellamento possono essere estesi non solo a quei casi in cui risulta non normato lo spazio di uscita di un processo orientato (classificazione), ma anche a quei casi in cui è non normato il suo spazio di ingresso o lo spazio caratteristico di un processo non orientato. Per semplicità di trattazione nel seguito non si affronterà la discussione sul modellamento di questi processi, i cui dati sono generalmente detti *dati strutturati*.

affinità di carattere logico, fisico, geometrico o altro ancora. Una misura dell'*affinità* tra due generici pattern può essere contenuta dalla matrice di affinità D: il generico elemento D(i,k) è detto *indice di affinità* e rappresenta una misura della similitudine, o della diversità, che intercorre tra i pattern i e k. In altre parole, l'indice di affinità rappresenta una particolare metrica che misura la distanza tra due pattern. Esistono diversi modi per esprimere questo indice, il quale può essere ricavato o dalla matrice dei pattern oppure dai soliti dati grezzi relativi ai vari pattern. Si dice in questo caso che si è in presenza di *dati relazionali* fra tutte le possibili coppie di oggetti che si hanno a disposizione. Si supporrà nel seguito che gli elementi sulla diagonale di D siano tutti uguali, cioè che ciascun pattern sia "uguale a se stesso" sempre nella stessa misura; inoltre si supporrà che la matrice sia simmetrica, cioè che tutte le coppie di pattern abbiano la stessa affinità indipendentemente dall'ordine in cui vengano considerate. Queste proprietà non sono del tutto scontate, in quanto si possono fare esempi di indici che non le soddisfano.

E' interessante notare come il passaggio da matrice di affinità a matrice dei pattern richieda l'utilizzo di metodi generalmente approssimati, con perdita quindi di informazione, come il *multidimensional scaling* o le *proiezioni* che saranno brevemente introdotti nel par. 2.7. Il passaggio inverso è invece molto più semplice: ad esempio, qualsiasi metrica[3] $d : \Re^n \times \Re^n \rightarrow \Re$ che agisca su una coppia di pattern nello spazio normato delle feature è in grado di fornire un indice di affinità. Nel par. 2.5 saranno descritti alcuni indici di affinità, in relazione alla tipologia di dati usati.

In conclusione, si vuole far notare che Bezdek in [Bezdek 1993] suggerisce un'ulteriore distinzione tra i valori che può assumere il generico elemento della matrice di affinità: se i suoi elementi sono solo quelli appartenenti all'insieme binario {0,1} essa si dirà *crisp*; se invece i valori degli elementi appartengono all'intervallo reale chiuso [0, 1] allora D sarà detta *fuzzy* [4]. Rimanendo più vicini alle definizioni date alla fine del par. 1.7.2, nel seguito sceglieremo di attribuire l'aggettivo crisp o

[3] Si ricordi l'isomorfismo supposto tra lo spazio dei pattern *E* e quello delle n-ple reali $\Re^n$.

[4] Non c'è perdita di generalità nel considerare come valore massimo 1, in quanto è sempre possibile un'operazione di normalizzazione.

fuzzy alla matrice di affinità in base al significato logico e non al valore numerico assunto dai suoi elementi.

2.4 Scale e tipi di dato

In [Anderberg 1973] è fornita una classificazione dei dati riassumibile nello schema di fig. 2.3. Questo schema ribadisce il fatto che, a partire dai dati raccolti in modo qualsiasi, è sempre possibile costruire la matrice dei pattern M e/o quella di affinità D; tali matrici saranno nel seguito considerate gli unici ingressi possibili ad un sistema di modellamento.

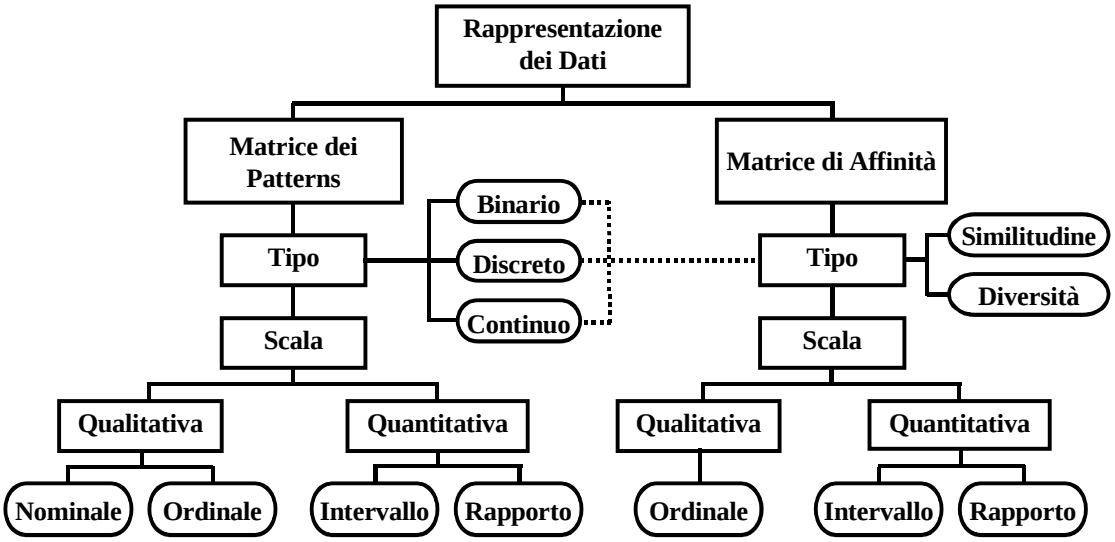


Figura 2.3 - Classificazione dei dati usati nei sistemi di modellamento.

Prendendo in esame il singolo elemento della matrice dei pattern, esso può essere di tre tipi differenti: *binario*, *discreto* o *continuo.* Nel primo caso può assumere soltanto due valori {0, 1}; nel secondo caso può assumere un numero finito di valori (per esempio, i campioni di un segnale vocale possono essere quantizzati a 16 livelli utilizzando 4 bit). Si osservi come il tipo binario è un caso particolare del tipo discreto quando il numero di livelli possibili è due (è utilizzato un solo bit). L'ultimo caso, quello in cui il valore assunto da un elemento della matrice è continuo, può essere interpretato come un punto qualsiasi appartenente ad un determinato intervallo definito lungo l'asse reale legato a quella particolare feature. Si noti che l'elaborazione

su calcolatori digitali richiede sempre una rappresentazione numerica con un numero finito di cifre significative, quindi richiede sempre un tipo di dato discreto come possono essere le misure analogiche processate da un apposito sistema di conversione analogico\digitale (A\D). E' tuttavia comodo pensare ad una rappresentazione continua, sia per generalizzare il discorso a sistemi di calcolo anche analogici come il cervello o le reti neurali, sia perché, nella stragrande maggioranza dei casi, la precisione dei calcolatori è così elevata rispetto a quella richiesta dall'utilizzatore finale da potere ritenere i dati comunque continui.

Anche il valore di un indice di affinità può essere di tipo binario, discreto o continuo. Se si suppone di avere a disposizione un insieme di oggetti partizionati in un certo numero di sottoinsiemi, un indice di affinità di tipo binario può assumere il valore zero nel caso in cui i due pattern considerati non appartengano ad uno stesso sottoinsieme, mentre può assumere il valore uno in caso contrario. Dati n oggetti, un indice di tipo discreto potrebbe essere definito facendogli assumere valori interi nell'intervallo $[1, n\cdot(n-1)/2]$, in cui l'estremo superiore indica il numero massimo di coppie differenti realizzabili con gli n oggetti; si potrebbe pensare di assegnare valori tanto maggiori all'indice quanto più i due oggetti considerati risultano simili tra loro. Un esempio di indice di affinità di tipo continuo è invece dato da una qualsiasi metrica a valori reali definita tra due pattern nel loro spazio reale euclideo.

Il tipo di indice di affinità può essere definito non solo in base alle caratteristiche del valore numerico che esso può assumere, ma anche in base al significato vero e proprio dell'indice. Si parla infatti di *indice di diversità* nel caso in cui il valore numerico dell'indice dia una misura di quanto differiscono due pattern, mentre si parla di *indice di similitudine* nel caso in cui il valore dell'indice misuri quanto i due pattern si somigliano.

Una seconda caratteristica saliente di un generico elemento delle due matrici $\underline{P}$ e $\underline{D}$ è rappresentata dalla *scala* dei dati: essa esprime l'importanza di un valore numerico. La scala dei dati può essere di tipo *qualitativo* (*nominale* e *ordinale*) o *quantitativo* (*intervallo* e *rapporto*). In una scala di tipo qualitativo i valori numerici sono privi di significato quantitativo assoluto: essi vengono usati semplicemente come nomi (ad esempio, "si" e "no" potrebbero essere codificati indifferentemente con 1 e 0 oppure con 50 e 100), per cui non si tratta di una vera e propria scala numerica. In una scala

qualitativa ordinale i singoli valori numerici hanno significato solo se confrontati tra loro: gli insiemi di valori (1, 2, 3) , (10, 20, 30) e (1, 20, 300) sono equivalenti tra loro, questa è la scala in cui il significato numerico quantitativo è il più "debole" rispetto alle altre. Diverso è il caso in cui si voglia attribuire un significato "forte" ai valori numerici, conviene allora utilizzare una scala quantitativa. Una scala quantitativa ad intervallo è utilizzata quando si vuole porre particolare attenzione alla separazione tra due valori numerici e pertanto, per poterla definire, occorre stabilire un'unità di misura e l'intervallo dei valori che essa può assumere. Per poter confrontare due valori è necessario che essi siano espressi con la stessa unità di misura: si pensi alla differenza che intercorre tra 5 ed 8 in una scala da 1 a 10 o in una scala da 0 a 100. La scala quantitativa a rapporto è quella con il significato numerico più forte, proprio perché i numeri hanno un significato assoluto. In essa esiste, oltre ad un'unità di misura, anche uno zero assoluto in modo da dare significato al rapporto tra due valori numerici piuttosto che alla loro separazione: raddoppiare la distanza fra due punti in uno spazio ha sempre lo stesso significato, indipendentemente dall'unità di misura (metri, miglia, millimetri, ecc.) utilizzata.

2.5 Esempi di metriche: gli indici di affinità

Proseguendo il discorso iniziato al par. 2.3, è ora possibile dare una descrizione più dettagliata degli indici di affinità in base alla scala utilizzata nei dati. Un indice di affinità d(i,k) fra i pattern i e k deve soddisfare alle seguenti tre proprietà:

1. $\begin{cases} d(i,i) = 0, \ \forall i \ , & \text{per un indice di diversità} \\ d(i,i) \geq \max_h \{d(i,h), \ \forall i\}, & \text{per un indice di somiglianza} \end{cases}$

2. $d(i,k) = d(k,i), \ \forall i \text{ e } \forall k$ (simmetria);

3. $d(i,k) \geq 0, \ \forall i \text{ e } \forall k$.

2.5.1 Scala a Rapporto

Un modo molto comune per determinare un indice di diversità, a partire dalla matrice dei pattern ed usando una scala a rapporto, è rappresentato dalla metrica del matematico lituano Minkowski; detto $\underline{x}_k = [x_{k1}\ x_{k2}\ ...\ x_{kn}]$ il k-esimo pattern nello spazio n-dimensionale dei pattern, la *metrica di Minkowski di ordine* r è definita come segue:

$$d_r(i,k) = \left(\sum_{c=1}^{n} |x_{ic} - x_{kc}|^r \right)^{1/r}, \text{ con } r > 0 .$$

Queste metriche soddisfano ad altre due proprietà enunciate di seguito:

4. $d(i,k) = 0 \ \Leftrightarrow \ \underline{x}_i = \underline{x}_k$;

5. $d(i,k) \le d(i,m) + d(m,k), \ \forall i, \ \forall k$ e $\forall m$ (relazione triangolare).

In [Gower 1986] è stato dimostrato che per un indice di diversità sono sufficienti soltanto le proprietà 1 e 3; le altre possono essere ricavate da queste due. Definendo particolari valori dell'ordine r si ottengono le seguenti distanze di Minkowski:

- *Norma di Manhattan* (o della *città a blocchi*, o *taxi*):

$$d_1(i,k) = \sum_{c=1}^{n} |x_{ic} - x_{kc}| \quad (r=1)$$

  è chiamata così in quanto rappresenta la distanza che deve coprire un individuo che si muove in una città con strade tra di loro perpendicolari o parallele. Quando le feature sono tutte binarie questa metrica è anche detta *distanza di Hamming* e rappresenta il numero di feature (bit) differenti tra i due pattern.

- *Norma euclidea*:

$$d_2(i,k) = \sqrt{\sum_{c=1}^{n} (x_{ic} - x_{kc})^2} \quad (r=2)$$

è la ben nota norma definita tramite il prodotto scalare standard nello spazio reale euclideo $\Re^n$. E' l'unica distanza invariante rispetto alle rotazioni ed alle traslazioni geometriche dei pattern;

- <u>*Norma di Lagrange* (o *norma superiore*) :</u>

$$d_\infty(i,k) = \lim_{r\to\infty} d_r(i,k) = \max_{c=1,2,...n} \left\{ \left| x_{ic} - x_{kc} \right| \right\} \qquad (r \to \infty)$$

In fig. 2.4 sono mostrati i diversi risultati che si ottengono applicando ad un caso bidimensionale le tre metriche di Minkowski sopra definite:

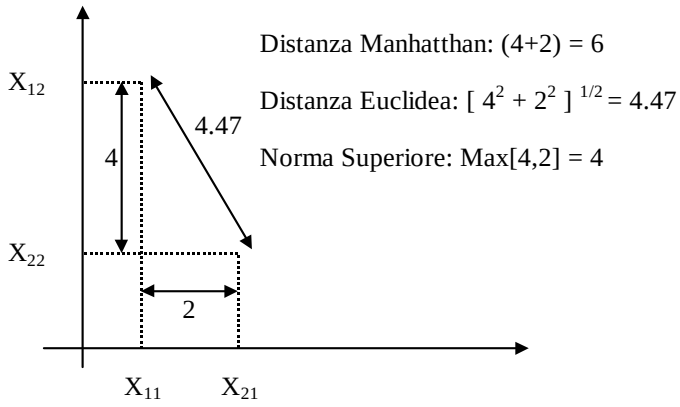


Figura 2.4 - Alcune metriche di Minkowski.

Esistono anche altri tipi di indici di affinità, per uno studio dei quali si rimanda a testi più specifici come [Jain 1988]. Tuttavia, è interessante menzionare anche la metrica introdotta nel clustering in [Everitt 1974]: la *distanza di Mahalanobis.* Dati due pattern $\underline{x}_i$ e $\underline{x}_k$, essa è definita come segue:

$$d(i,k) = (\underline{x}_i - \underline{x}_k)^t \underline{C}^{-1} (\underline{x}_i - \underline{x}_k) \qquad (2.1)$$

dove $\underline{C}$ è la stima della matrice di covarianza dell'insieme di pattern. Questa metrica incorpora le correlazioni fra le diverse feature e normalizza ciascuna di esse a valor medio nullo e varianza unitaria; risulta pertanto utile laddove si voglia ispezionare la struttura di una data matrice di covarianza. Una volta fissato uno dei due pattern al valor medio dell'insieme, e posta d(i,k) ad un valore costante c (ad esempio c=1), il luogo dei punti che soddisfano la (2.1) nello spazio n-dimensionale descrive una

iperellisse orientata secondo le direzioni ortogonali di massima varianza (*componenti principali*) dei pattern. Risulta ora evidente quanto accennato nel par. 1.7.2 riguardo al fatto che una densità di probabilità in realtà può essere usata come metrica per la distanza tra un punto ed un insieme. L'argomento di una densità di probabilità gaussiana $p(\underline{x})$, ad esempio, è proprio costituito dalla 2.1; pertanto $p(\underline{x})$ misura la distanza del punto $\underline{x}$ dal centro (valor medio della gaussiana) mediante opportuna pesatura intorno ad una regione iperellittica (covarianza della gaussiana).

Alcune Curiosità geometriche

Anche le distanze di Manhattan, Euclidea e di Lagrange hanno una semplice interpretazione geometrica che può essere compresa nel caso bidimensionale come luogo di punti in un piano. Infatti, considerato un punto $\underline{x} = [x_1 \;\; x_2]$, si può notare in fig. 2.5 che il luogo dei punti $\underline{x}_i$ che distano dall'origine $\underline{x}_0$ è una quantità costante $d_r(\underline{x}_i, \underline{x}_0)$ rappresentata da:

- una circonferenza di raggio $d_2(\underline{x}_i, \underline{x}_0)$, nel caso $r = 2$;
- il quadrato inscritto nella circonferenza, nel caso $r = 1$;
- il quadrato circoscritto alla circonferenza, nel caso $r \to \infty$.

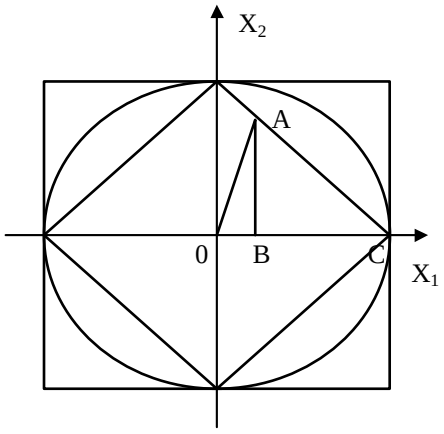


Figura 2.5 - Interpretazione geometrica delle metriche di Minkowski.

Infatti, per ogni punto della circonferenza la distanza euclidea dall'origine è costante e pari al raggio; per i punti del quadrato inscritto nella circonferenza la

distanza dall'origine è costante se si considera la distanza di Manhattan: nel triangolo isoscele *ABC* si ha $\overline{AB} = \overline{BC}$, per cui $\overline{OB} + \overline{BA} = \overline{OC}$; per i punti $\underline{x}_i$ appartenenti al quadrato circoscritto si ha una situazione analoga usando la distanza di Lagrange: $d_\infty(\underline{x}_i, \underline{x}_0) = \max\{|x_{i1} - 0|, |x_{i2} - 0|\} = \text{cost}$.

2.5.2 Scala Nominale

Sarà trattato solo il caso di feature a valori nominali binari, le quali di solito derivano dalla raccolta di dati su soggetti umani come nel caso di interrogazioni a fine statistico. L'interesse è riposto nel fatto che gli indici di affinità ricavati in questo modo possono anche valutare la distanza tra due partizioni di oggetti a seguito, ad esempio, di due differenti processi di clustering.

Per valutare un indice di affinità di questo tipo tra due pattern $\underline{x}_i$ e $\underline{x}_k$, si parte dalla seguente tabella detta, con termine inglese, *contingency table*:

		$\underline{x}_k$	
		1	0
$\underline{x}_i$	1	a_{11}	a_{10}
	0	a_{01}	a_{00}

dove a_{11} è il numero di feature pari ad 1 in entrambi i pattern; a_{10} è il numero di feature che sono 1 in $\underline{x}_i$ e 0 in $\underline{x}_k$; e così via. A questo punto è possibile definire diversi indici di affinità, tre sono quelli più utilizzati:

- *Jaccard*: $d(i,k) = 1 - \dfrac{a_{11}}{a_{11} + a_{10} + a_{01}}$;
- *Simple Matching*: $d(i,k) = 1 - \dfrac{a_{00} + a_{11}}{a_{00} + a_{11} + a_{10} + a_{01}}$;
- *Rand Adjusted*: $d(i,k) = 1 - \dfrac{2(a_{11}a_{00} - a_{10}a_{01})}{2a_{11}a_{00} + (a_{11} + a_{00})(a_{10} + a_{01}) + a_{10}^2 + a_{01}^2}$.

Ne risulta che più il valore dell'indice è prossimo a zero e più i pattern sono uguali, viceversa se l'indice tende ad uno. Inoltre, il denominatore vale zero se e soltanto se i

due pattern sono uguali, per cui non c'è ambiguità nel caso di eventuali singolarità nei valori dell'indice.

Il problema fondamentale nell'utilizzo di questi indici è la loro calibrazione: bisogna trovare un criterio per stabilire, dato un loro valore tra 0 ed 1, se questo rappresenti una grande diversità o una grande similitudine; è necessario quindi trovare una soglia opportuna che non sia necessariamente pari a 0.5. La calibrazione di questi indici può essere ottenuta utilizzando tecniche di tipo statistico che misurino l'aleatorietà, ossia la mancanza di struttura, dei valori dell'indice per un particolare insieme di dati [Jain 1988].

2.6 Il preprocessing dei dati: normalizzazioni

Una volta stabilita una metrica nello spazio caratteristico del processo ed un adeguato principio di induzione, ciò che maggiormente influisce sul risultato di un sistema di modellamento è l'eventuale *preprocessamento* (o *preprocessing*) dei dati. Si supponga che lo spazio caratteristico del processo sia $\Re^n$; in generale, si parla di preprocessamento qualora si operi una trasformazione dei pattern secondo una data funzione (lineare o non lineare) $\varphi : \Re^n \rightarrow \Re^d$. In questo caso il sistema di modellamento agirà, sia nel caso supervisionato che non supervisionato, su uno spazio caratteristico del processo ($\Re^d$) che non coinciderà sicuramente con lo spazio di ingresso del modello; pertanto, quanto eventualmente stabilito in R^n circa la scelta di una metrica e di un principio di induzione non verrà necessariamente conservato in R^d. E' evidente che, ai fini del modellamento, ciò che conta è quanto viene stabilito per lo spazio di ingresso del modello. Questo fatto rimane valido anche se fosse d=n e se $\varphi(\cdot)$ fosse un'applicazione lineare, dal momento che anche in questo caso la funzione di preprocessamento altererebbe la metrica adottata.

L'effetto di distorsione (lineare o non lineare) sulla metrica, dovuto ad una funzione di preprocessing, è utile nei casi in cui l'adozione di un certo sistema di modellamento implichi una scelta implicita (ed obbligata) circa la metrica nello spazio di ingresso del modello. Di fatto è proprio tramite una tecnica di preprocessamento che risulta possibile elevare il grado di flessibilità del sistema di modellamento scelto.

Il preprocessing per eccellenza è sicuramente rappresentato dalla normalizzazione dei dati: una corretta esecuzione di tale operazione risulta essere fondamentale ai fini del raggiungimento di un buon sistema di modellamento. La necessità di una normalizzazione non è suscitata soltanto dal voler limitare il dominio delle coordinate dei dati o dal voler comprimere gli stessi in modo tale da appartenere ad opportuni intervalli (ad esempio al fine di evitare problemi di instabilità numerica). Un'adeguata normalizzazione serve anche per evitare problemi di pesatura implicita delle singole componenti di un pattern. Si verificano effetti di pesatura implicita quando l'intervallo di variazione (*range*) di ciascuna componente del pattern cambia notevolmente di componente in componente, ciò accade frequentemente in presenza di grandezze fisiche o logiche disomogenee o quando due o più campi rappresentano una stessa grandezza fisica espressa con unità di misura differenti.

In assenza di informazioni a priori circa l'importanza relativa dei campi di un pattern, risulta opportuno riportare ciascuna componente a poter variare nel medesimo intervallo dell'asse reale. La normalizzazione di tipo affine consiste proprio nel riportare la variabilità di ogni singola componente nell'intervallo [0, 1], il che significa assumere l'ipercubo unitario come spazio di ingresso del modello.

- Normalizzazione di tipo affine.

  Detti *min* e *max* i valori minimi e massimi dell'insieme da normalizzare, ad ogni dato viene sottratto il valore minimo ed il tutto viene diviso per la differenza tra il massimo e il minimo. Con questa procedura vi sarà sempre un valore (quello massimo) mappato in 1 ed uno (quello minimo) mappato in 0. Indicando con x il generico valore da normalizzare (la generica feature di un pattern), e con y il corrispettivo valore normalizzato, si osserva una relazione del tipo $y = m \cdot x + q$, infatti:

$$y = \frac{(x - min)}{(max - min)} = \frac{x}{(max - min)} - \frac{min}{(max - min)} = m \cdot x + q$$

  Nonostante l'equazione precedente sia quella di una retta, il tipo di normalizzazione non si può considerare lineare poiché ad una combinazione

$x = ax_1 + bx_2$ corrisponde un valore normalizzato y(x) diverso da $ay(x_1) + by(x_2)$.

Il tipo di normalizzazione affine risulta essere poco indicato nei casi in cui si voglia porre particolare attenzione alle caratteristiche geometriche di un certo insieme di dati. Se tutti i suoi punti presentassero un valore molto simile (ad esempio intorno a 0.5) in corrispondenza ad una fissata coordinata, una volta effettuata la normalizzazione affine i punti si troverebbero fortemente dispersi lungo la coordinata in questione. L'esempio di fig. 2.6 mostra lo scatter plot di un insieme di dati, composto presumibilmente da due cluster, i cui punti presentano lungo la coordinata Y valori prossimi a 0.5. Questo insieme è mappato in quello di fig. 2.7 con una normalizzazione affine. E' evidente la dispersione dei punti lungo l'asse delle Y, con una conseguente distorsione della metrica originale: i due cluster non risultano più ben compatti come lo erano prima della normalizzazione.

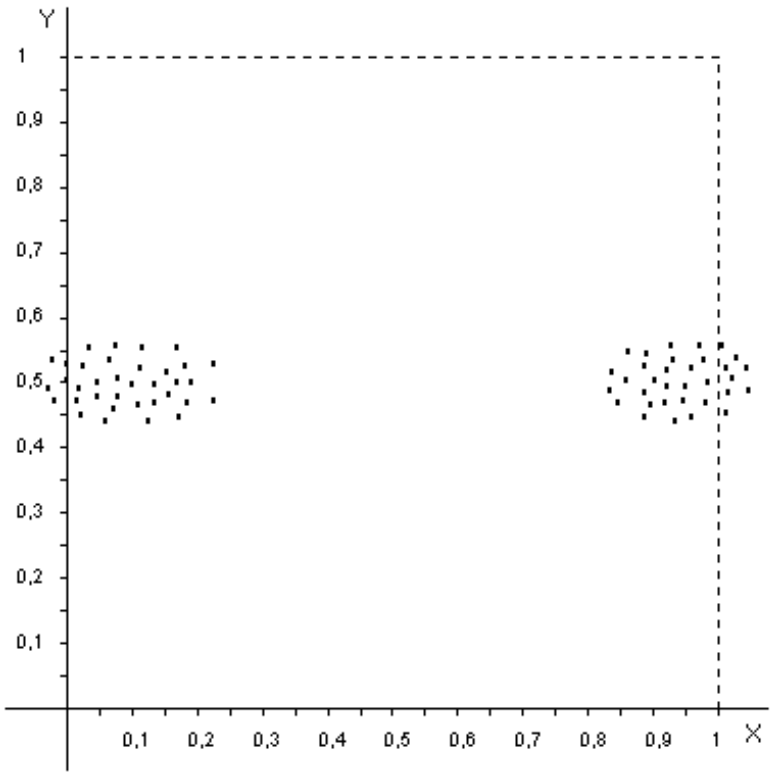


Figura 2.6 - I valori della componente Y sono addensati attorno al valore 0.5.

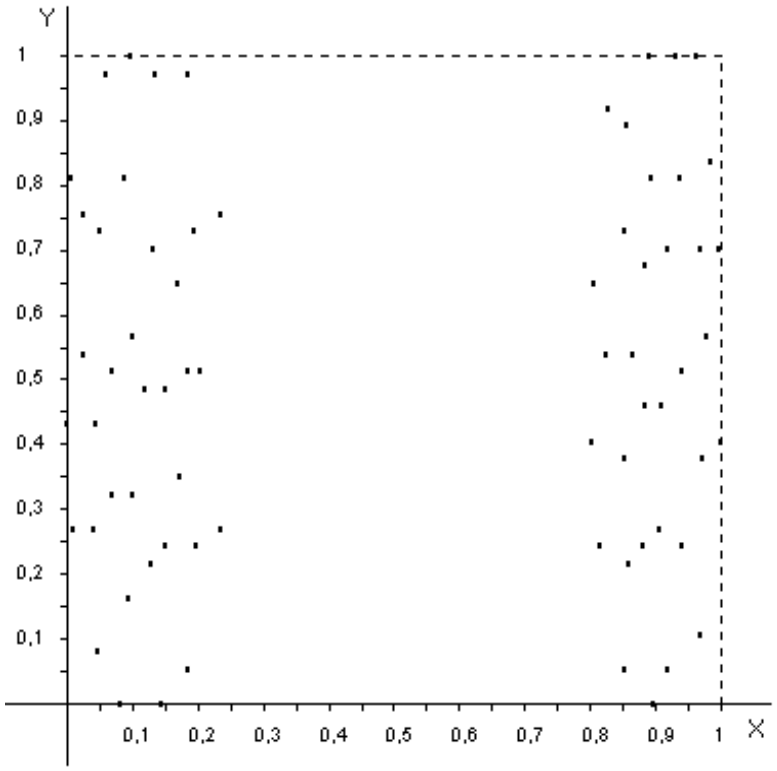


Figura 2.7 - L'insieme di dati di fig. 2.6 normalizzato in modo affine.

- Normalizzazione di tipo geometrico

  Al fine di evitare gli effetti collaterali dovuti alla normalizzazione di tipo affine, qualora non desiderati, è possibile ricorrere ad una normalizzazione di tipo geometrico. Nel caso in cui lo scopo della normalizzazione sia soltanto quello di riportare le coordinate di ciascun pattern all'interno dell'intervallo [0, 1], mantenendo il più possibile inalterate le relazioni geometriche tra i singoli punti, e non considerando il problema delle diverse unità di misura con cui vengono espresse le coordinate, si potrebbe pensare di applicare una normalizzazione affine solo lungo quelle dimensioni in cui vi è almeno un pattern con coordinata maggiore di uno o minore di zero. Mentre da una parte si commette un errore logico, poiché alcune coordinate (normalizzate) diventano numeri puri ed altre conservano la loro unità di misura, dall'altra si riescono a preservare alcune caratteristiche metriche quali la compattezza di un cluster.

Operando una normalizzazione di tipo geometrico sul data set di fig. 2.6 si ottiene il risultato presentato in fig. 2.8: è evidente come non si sia verificata alcuna dispersione dei punti lungo l'asse Y, a causa del fatto che nessuna normalizzazione è stata effettuata su questa componente. Infatti, tutti i pattern hanno in essa valori compresi nell'intervallo [0, 1].

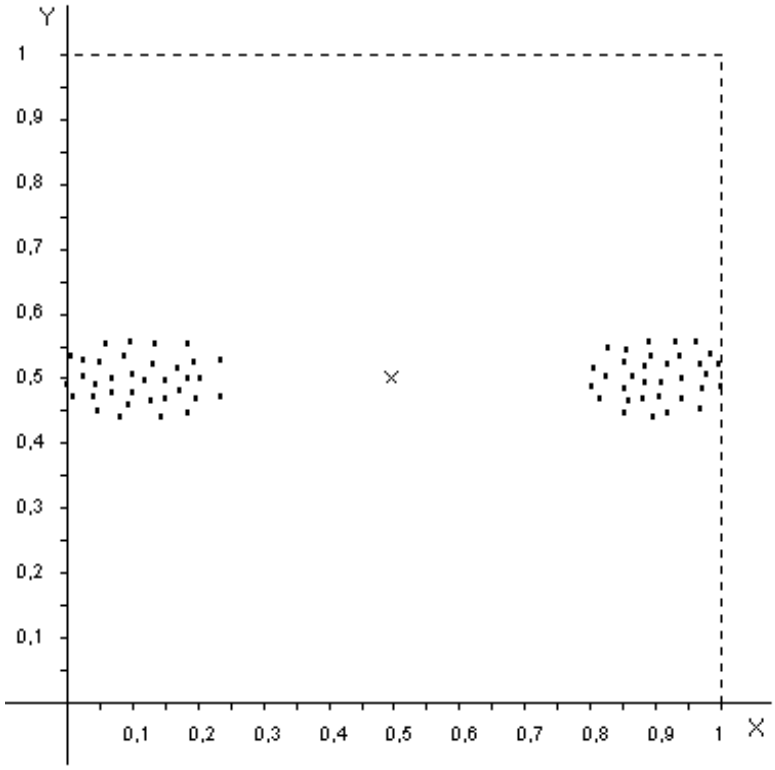


Figura 2.8 - Normalizzazione geometrica dell'insieme di pattern in fig.2.6.

- Normalizzazione di tipo lineare

  Si tratta di rapportare tutti i valori di un dato insieme per il valore massimo tra essi presente; una volta effettuata la normalizzazione i valori del sottoinsieme in questione risultano compresi nel dominio [*min*/*max*, 1], in cui il significato di *min* e *max* è evidente. E' da notare che la relazione del tipo $y=m \cdot x$, con $m=1/max$, tra il dato da normalizzare (x) ed il dato normalizzato (y) è lineare. Questo tipo di normalizzazione gode della proprietà di mantenere intatti i rapporti tra i dati da normalizzare. Per contro, sussiste l'impossibilità di utilizzare tale tipo di normalizzazione su dati di valore

negativo[5], nonché la tendenza ad addensare intorno ad 1 i dati normalizzati nel caso in cui si verifichi $min \approx max$ e $min > 1 \Rightarrow max > 1$.

- Normalizzazione statistica

  L'esempio più classico di normalizzazione statistica è quello che considera i valori dell'insieme da normalizzare come determinazioni di una variabile aleatoria. Detti m_j e s_j^2 rispettivamente il valore medio e la varianza dei dati nella j-esima componente dei pattern, si ha :

$$m_j = \frac{1}{p}\sum_{i=1}^{p} x_{ij} \qquad (2.2)$$

$$s_j^2 = \frac{1}{p}\sum_{i=1}^{p}\left(x_{ij} - m_j\right)^2 \qquad (2.3)$$

  in cui p è il numero di pattern e x_{ij} indica la componente j-esima del pattern i-esimo. Più precisamente, la (2.2) e la (2.3) rappresentano una stima data driven del valor medio e della varianza, rispettivamente. Si rimanda a [Ciccarella 1993] per una trattazione specifica del problema.

  Questa normalizzazione consiste nel traslare e nello scalare i dati in modo tale che essi presentino media nulla e varianza unitaria:

$$\tilde{x}_{ij} = \frac{x_{ij} - m_j}{s_j}$$

  in cui $\tilde{x}_{ij}$ è il nuovo valore di x_{ij} dopo la normalizzazione. Anche la normalizzazione statistica comporta un'alterazione della metrica scelta.

Le normalizzazioni lineare, affine e geometrica comportano la definizione di valori di minimo e massimo in funzione dei quali i dati dovranno essere normalizzati. Da questo punto di vista è necessario fare un'ulteriore distinzione, sulla base di quali valori vengono presi in considerazione per il calcolo di tali valori estremi.

[5] Se $max < 0$, allora è possibile normalizzare dividendo i valori per il suo valore assoluto. In questo modo l'intervallo dei valori normalizzati sarà del tipo [-abs(*min*/*max*), -1].

- *Normalizzazione per componente*. E' la modalità di normalizzazione più frequentemente utilizzata, in cui le coordinate del generico pattern esprimono valori di grandezze fisiche diverse, o relativi ad unità di misura diverse di una stessa grandezza fisica. Serve essenzialmente per evitare di pesare differentemente le singole componenti. Ogni feature (ogni colonna della matrice dei pattern) viene normalizzata in modo del tutto indipendente dalle altre secondo gli estremi:

$$min_j = \min_{i=1..p}\{x_{ij}\}, \quad max_j = \max_{i=1..p}\{x_{ij}\}.$$

- *Normalizzazione per pattern*. Questo tipo di normalizzazione si utilizza nei casi in cui le coordinate del generico pattern i-esimo possono essere interpretate come campioni di una sequenza. In particolare, si vuole discriminare i pattern in base alla loro forma, ossia ai rapporti fra le diverse componenti, piuttosto che in base al valore medio delle componenti in un pattern. Si rende allora necessario operare tante normalizzazioni quanti sono i pattern, adottando i seguenti estremi di normalizzazione:

$$min_i = \min_{j=1..n}\{x_{ij}\}, \quad max_i = \max_{j=1..n}\{x_{ij}\}.$$

- *Normalizzazione totale*. E' corretto operare una normalizzazione totale solo quando tutte le componenti sono omogenee (rappresentano la stessa grandezza espressa nella medesima unità di misura). Si tratta di operare un'unica normalizzazione su tutti gli elementi della matrice dei pattern:

$$min = \min_{\substack{i=1..p \\ j=1..n}}\{x_{ij}\}, \quad max = \max_{\substack{i=1..p \\ j=1..n}}\{x_{ij}\}.$$

2.7 Il preprocessing dei dati: proiezioni

Un algoritmo di proiezione può essere interpretato come una funzione $\varphi : \Re^n \rightarrow \Re^d$, capace di mappare in uno spazio d-dimensionale un insieme di oggetti appartenenti ad uno spazio n-dimensionale con d<n. Una qualsiasi riduzione di dimensione sicuramente altera le caratteristiche di un insieme di punti (l'insieme ottenuto sarà sicuramente diverso da quello di partenza); per questo motivo la mappatura deve essere operata con un certo criterio, nel senso che bisogna cercare di ottenere una configurazione che conservi il più possibile le informazioni contenute nella distribuzione iniziale dei punti.

Il motivo principale che induce ad utilizzare questi algoritmi è la necessità di ottenere una riduzione di dimensione e/o ridondanza dei data set (training o test set). A titolo di esempio, si può ricordare che le prime tecniche di clustering erano proprio basate sulla proiezione in uno spazio di arrivo bidimensionale. In tal modo, si verificava l'opportunità di poter disegnare i pattern in un piano e quindi di poter determinare, mediante ispezione visiva, i cluster stessi.

2.7.1 Proiezioni lineari

Una proiezione lineare esprime le nuove d componenti del data set come una combinazione lineare delle n componenti originali:

$$\underline{y}_i = \underline{H} \cdot \underline{x}_i \ , \quad i=1\ldots p$$

in cui: $\underline{y}_i$ è un vettore colonna a d dimensioni; $\underline{x}_i$ è un vettore colonna ad n dimensioni; $\underline{H}$ è una matrice $M_{d,n}(\Re)$; p è il numero di pattern del data set.

Un esempio di proiezione lineare, che sarà ampiamente utilizzata nel seguito, è rappresentato dalla *trasformata di Karhunen-Loève* basata sull'*analisi alle componenti principali* (PCA) dei dati [Bishop 1995]. Con questa tecnica si opera una riduzione della dimensione dello spazio di ingresso cercando di mantenere il più possibile inalterate le caratteristiche che differenziano un dato dall'altro. Affinché le componenti di un pattern siano significative, esse devono variare sensibilmente da un pattern all'altro: se una feature avesse valori "quasi" uguali in tutti i pattern, questa

non contribuirebbe a determinare le diversità tra i generici punti (non fornirebbe un contributo informativo al problema di modellamento) e pertanto potrebbe essere eliminata in modo complementare a quanto accade nel caso di dipendenza lineare affrontato nel successivo par. 2.8. E' possibile accorgersi del contributo che fornisce una feature andando a considerarne la varianza: le componenti a bassa varianza non portano molte informazioni e possono essere scartate.

Esistono anche altre tecniche di proiezione lineare, la cui descrizione non è di particolare utilità nel contesto di questo lavoro. E' comunque interessante citare alcuni metodi come la *derivazione*, in cui si tenta di scegliere la nuova base nello spazio d-dimensionale con l'obiettivo di minimizzare l'errore quadratico derivante dalla ricostruzione approssimata (n-dimensionale) dei pattern ridotti. Esistono anche metodi come l'*analisi discriminante*, in cui si fa uso di etichette note a priori sulle categorie di appartenenza dei pattern. Questi ultimi sono ovviamente utilizzabili solo nel caso di classificazione supervisionata.

2.7.2 Proiezioni non lineari

Le tecniche di proiezione lineare risultano poco soddisfacenti nei casi in cui i pattern abbiano una disposizione geometrica complessa, come nel caso in cui siano posizionati sulla superficie di un'ellisse. E' stata proprio questa la causa che ha portato allo sviluppo di algoritmi di proiezione non lineare.

La maggior parte di questi algoritmi sono basati sulla ricerca di un massimo (o di un minimo) globale di una funzione obiettivo dipendente da un gran numero di parametri. A causa dell'alto costo computazionale necessario per la ricerca della soluzione ottimale, l'esecuzione di tale algoritmi viene spesso preceduta da tecniche di tipo lineare come la PCA. L'obiettivo principale di un algoritmo di proiezione non lineare è, oltre a quello di mappare i punti su uno spazio di dimensione inferiore, quello di cercare di mantenere inalterata il più possibile la distanza tra due generici pattern.

Un esempio di tecnica di proiezione non lineare è rappresentato dal *Sammon mapping*, il quale tenta di proiettare ciascun punto dello spazio di ingresso (a

dimensione n>2) su un particolare piano bidimensionale sotto il vincolo di mantenere inalterate il più possibile le suddette distanze. Il rispetto di tale vincolo viene realizzato da Sammon andando a minimizzare una funzione costo che dà una misura di quanto differiscano le distanze calcolate nello spazio di partenza e in quello di arrivo per tutte le generiche coppie di punti. Il metodo definisce una procedura iterativa di ottimizzazione della funzione costo di tipo discesa a gradiente, che risulta però computazionalmente molto costosa, molto sensibile alla presenza di minimi locali e quindi alla scelta delle condizioni iniziali.

Esistono anche altri metodi non lineari, che non saranno ulteriormente approfonditi, come le *triangolazioni*. Esse sono meno costose rispetto al metodo di Sammon ma tendono ad ignorare la struttura globale dei dati, tenuta meglio in conto quando si fa uso di algoritmi che minimizzano gli errori quadratici.

2.7.3 Multidimensional scaling

Il Multidimensional Scaling (MDSCAL) è il nome per una tipologia di procedure ed algoritmi che, a partire da una matrice di affinità qualitativa e con dati ordinali, generano configurazioni dei relativi pattern in una, due o tre dimensioni. Gli obiettivi del MDSCAL sono gli stessi delle procedure iterative di proiezione non lineare (che partono però da dati con scala a rapporto) e rappresentano, come anticipato, il passaggio inverso da matrice di affinità a matrice dei pattern. Al fine di rappresentare i dati, il MDSCAL trasla una scala ordinale in un insieme di dimensioni o scale a rapporto: questo è un esempio di ordinamento. Esiste un'ampia letteratura riguardo al MDSCAL; l'algoritmo di base è stato presentato per la prima volta in [Kruskal 1964, Kruskal 1964(bis)].

Quando i dati di ingresso al clustering sono organizzati secondo una siffatta matrice di affinità ordinale, si dice anche che si sta eseguendo un *clustering a coppie* (o *pairwise data clustering*). Se quindi il MDSCAL rappresenta una procedura di visualizzazione di questi dati, il clustering a coppie ne rappresenta la procedura di analisi strutturale. In letteratura sono presenti diversi algoritmi per questo tipo di clustering, i quali cercano di realizzare contemporaneamente questi due obiettivi

usando tecniche e teorie ottimizzanti come il d*eterministic annealing*, la *massima entropia*, e via dicendo.

2.8 Il problema dei dati mancanti

Può accadere che, a valle del campionamento di un processo, alcuni pattern di un data set non presentino tutti i valori delle feature in esame, assumendo così una forma del tipo $\underline{x}_i = [x_{i1}\ ?\ x_{i3}\ ?\ ...\ x_{in}]$. L'incompletezza di un pattern può dipendere da diversi motivi tra i quali i più frequenti sono gli errori di acquisizione e memorizzazione dei dati, la non disponibilità o l'impossibilità da parte del contesto in esame di fornire informazioni, ecc.

Diverse strategie si possono applicare per porre rimedio al problema dei dati mancanti; alcune di esse mirano a stimare (interpolare) le feature che mancano in qualche pattern, altre si limitano a valutare la distanza tra due pattern incompleti. La scelta della tecnica più corretta da applicare dipende dal numero di pattern e di feature nel data set.

In [Dixon 1979] sono state proposte alcune tecniche, ancora oggi del tutto valide, per l'analisi di data set contenenti pattern con dati mancanti. Alcune di queste strategie possono essere riassunte nei seguenti punti:

1. nel caso in cui il numero di pattern incompleti è molto più piccolo rispetto a quello totale nel data set, si potrebbe pensare di eliminare direttamente l'intero pattern incompleto;
2. si supponga che nel pattern $\underline{x}_i$ manchi la coordinata j-esima x_{ij}; il valore di x_{ij} può essere sostituito dalla media delle j-esime coordinate dei K pattern più vicini ad $\underline{x}_i$. Il valore di K viene deciso a priori come funzione del numero di pattern del data set;
3. per definire un indice di affinità tra due pattern incompleti $\underline{x}_i$ e $\underline{x}_k$ è prima necessario stabilire l'espressione della distanza d_j lungo la j-esima coordinata:

$$d_j = \begin{cases} 0 & \text{se } x_{ij} \text{ o } x_{kj} \text{ è mancante} \\ x_{ij} - x_{kj} & \text{altrimenti} \end{cases}$$

L'affinità tra $\underline{x}_i$ e $\underline{x}_k$ può allora essere definita nel seguente modo:

$$d(i,k) = \frac{n}{n - n_0} \sum_{j=1}^{n} d_j^2$$

in cui n è il numero totale di feature ed n_o il numero di quelle mancanti. Si osservi che se $n_0 = 0$ la distanza sopra definita coincide con la norma euclidea;

4. sia $\bar{d}_j$ la distanza media fra tutte le possibili $\frac{p(p-1)}{2}$ coppie di pattern lungo la coordinata j, in cui p è il numero di pattern del data set:

$$\bar{d}_j = \frac{2}{p(p-1)} \sum_{i=2}^{p} \sum_{k=1}^{i-1} \left| x_{ij} - x_{kj} \right|$$

sia d_j la distanza tra due pattern lungo la coordinata j:

$$d_j = \begin{cases} \bar{d}_j & \text{se } x_{ij} \text{ o } x_{kj} \text{ è mancante} \\ x_{ij} - x_{kj} & \text{altrimenti} \end{cases}$$

ora si può definire nel modo seguente l'affinità tra due pattern $\underline{x}_i$ e $\underline{x}_k$:

$$d(i,k) = \sum_{j=1}^{n} d_j^2 .$$

Lo stesso Dixon, basandosi su risultati sperimentali, suggerisce di utilizzare la tecnica esposta al terzo punto.

E' interessante menzionare una procedura che, questa volta volutamente, tende ad eliminare completamente alcune feature del data set. Il fine è quello di ridurre la dimensione dei dati ed è una tipica operazione di preprocessing degli stessi: il criterio utilizzato è quello di scartare le feature (colonne della matrice dei pattern con scala a rapporto) che sono linearmente dipendenti dalle altre. Quando esiste una possibilità del genere significa che si è in presenza di una matrice dei pattern che non è *ben condizionata*, ossia che è stato mal posto il problema di acquisizione dei dati poiché, ad esempio, alcuni tipi di misure non contengono alcuna informazione aggiuntiva rispetto a quella già raccolta nelle altre. Supponendo un numero di pattern maggiore del numero di feature, questo significa ancora che la matrice non è a rango massimo.

Una volta determinato che è necessario, o preferibile, scartare le feature dipendenti, bisogna trovare quali fra esse sono da eliminare; per fare questo si può utilizzare, ad esempio, la stima dei *coefficienti di correlazione* tra le varie feature.

2.9 La struttura dei risultati nel clustering e nella classificazione

L'ultimo paragrafo di questo capitolo è dedicato alle diverse modalità con cui sono rappresentabili le uscite di un sistema di classificazione o di clustering. Dovrebbe essere chiaro, da quanto detto fino a questo punto, che le uscite di tali sistemi di modellamento sono costituite da etichette (*label*) vettoriali che definiscono le "relazioni" che ciascun pattern possiede con le regioni di influenza (cluster o classi) determinate durante l'apprendimento.

Si supponga, a titolo illustrativo[6], che un algoritmo abbia determinato l'esistenza di k cluster, con k eventualmente fissato a priori; a ciascun pattern i-esimo può essere assegnata un'etichetta $\underline{y}_i = [y_{i1}\ y_{i2}\ ...\ y_{ik}]$. Secondo quanto illustrato nel cap. 1, è possibile definire tre tipi di etichette in base alla logica di inferenza induttiva usata dall'algoritmo di clustering:

1. *Etichette Fuzzy*: $N_{fku} = \left\{ \underline{y} \in \Re^k : y_j \in [0 \quad 1],\ j = 1...k \right\}$;

2. *Etichette Probabilistiche*: $N_{fk} = \left\{ \underline{y} \in N_{fku} : \sum_{j=1}^{k} y_j = 1 \right\}$;

3. *Etichette Crisp*: $N_k = \left\{ \underline{y} \in N_{fk} : y_j \in \{0 \quad 1\},\ j = 1...k \right\}$;

dove N_k è l'insieme discreto di punti costituente la base canonica dello spazio reale euclideo $\Re^k$ delle k-ple reali; N_{fk} è l'insieme infinito di punti appartenenti ad un pezzo di iperpiano passante per questi punti ed iscritto nell'ipercubo unitario definito

[6] Il discorso che segue può essere direttamente trasferito al caso della classificazione laddove si sostituisca la parola "cluster" con la parola "classe". Si tenga comunque presente la notevole differenza concettuale tra queste due entità.

dagli stessi; N_{fku} è l'insieme infinito di punti contenuti all'interno dell'ipercubo unitario. In fig. 2.9 sono descritti questi insiemi nel caso di k=3 cluster: si noti come le etichette fuzzy e/o probabilistiche (come quelle denotate con **y**) giacciano sull'iperpiano bidimensionale N_{f3} dei punti la cui somma delle coordinate è 1; mentre quelle fuzzy (come quelle indicate con **z**) non abbiano alcun vincolo se non quello di essere all'interno dell'ipercubo unitario $N_{f3u} = [0 \quad 1]^3$; le tre etichette denotate con **e**, le quali sono al tempo stesso fuzzy, probabilistiche e crisp, costituiscono invece la base canonica $N_3 = (e_1 \quad e_2 \quad e_3)$ di $\Re^3$.

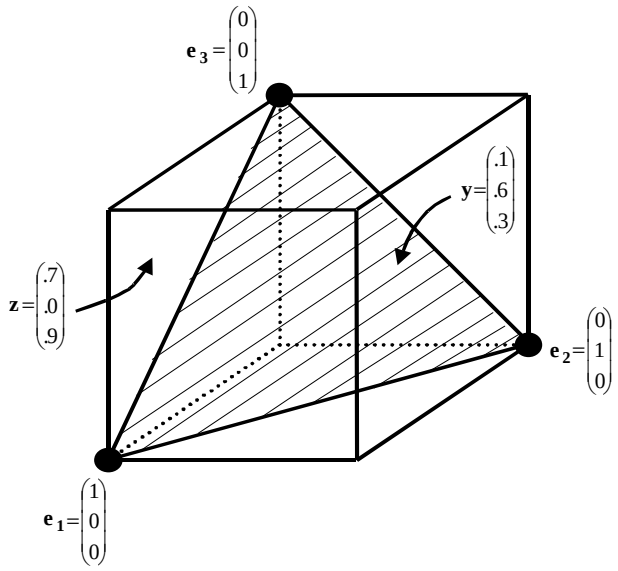


Figura 2.9 - Etichette vettoriali crisp, fuzzy e probabilistiche.

L'interpretazione che si può dare ai diversi tipi di etichette dipende dal tipo di inferenza induttiva e di algoritmo di apprendimento utilizzato: nel caso fuzzy il valore di ciascun elemento y_{ij} delle label non è altro che il valore di membership del pattern i-esimo al cluster j-esimo; nel caso probabilistico l'interpretazione più plausibile di questo valore è quella di probabilità che l'i-esimo pattern appartenga (comunque in modo crisp) al cluster j-esimo, ed in effetti la somma delle probabilità estesa a tutti i cluster è proprio 1; nel caso crisp, infine, il vettore etichetta avrà tutti elementi nulli, tranne uno che avrà valore unitario ed indicherà il cluster a cui il pattern appartiene in modo esclusivo.

Da un punto di vista più formale, si supponga di avere in ingresso al sistema di clustering un insieme di pattern X, i quali derivano dal campionamento di un processo nel suo spazio caratteristico e sono organizzati in uno dei modi visti precedentemente. Si supponga poi che il sistema produca k cluster, per cui ad ogni pattern sarà associata un'etichetta vettoriale di dimensione k; l'insieme delle etichette può essere composto per righe in una matrice $\underline{U} \in M_{p,k}(\Re)$ detta *partizione* di X. La riga i-esima di $\underline{U}$ è quindi l'etichetta assegnata al pattern i-esimo, mentre il generico elemento $\{u_{ij}\}$ di $\underline{U}$ non è altro che il singolo elemento di etichetta assegnato al pattern i-esimo per il cluster j-esimo. Per quanto detto, a seconda del tipo di partizione (crisp, fuzzy o probabilistica), esso deve soddisfare a qualcuna o a tutte delle seguenti condizioni:

$$0 \le u_{ij} \le 1 \quad \forall i, j \qquad (2.4a)$$

$$0 < \sum_{i=1}^{p} u_{ij} < p \quad \forall j \qquad (2.4b)$$

$$\sum_{j=1}^{k} u_{ij} = 1 \quad \forall i \qquad (2.4c)$$

Con queste posizioni, un sistema di clustering (ed analogamente di classificazione) si può definire come una funzione, o mappatura, *C* del seguente tipo:

- *clustering fuzzy:*

  $C : X \to M_{fkpu}$ con

  $M_{fkpu} = \left\{ \underline{U} \in M_{p,k}(\Re) : u_{ij} \text{ soddisdfa alle 2.4(a) e 2.4(b) } \forall i, j \right\}$,

  ossia le righe di M_{fkpu} sono etichette fuzzy appartenenti a N_{fku} ;

- *clustering probabilistico:*

  $C : X \to M_{fkp}$ con

  $M_{fkp} = \left\{ \underline{U} \in M_{fkpu} : u_{ij} \text{ soddisdfa alla 2.4(c) } \forall i, j \right\}$,

  ossia le righe di M_{fkp} sono etichette probabilistiche appartenenti a N_{fk} ;

- *clustering crisp:*

  $C : X \rightarrow M_{kp}$ con

  $M_{kp} = \left\{ \underline{U} \in M_{fkp} : u_{ij} = 0 \text{ o } 1 \quad \forall i, j \right\}$,

  ossia le righe di M_{kp} sono etichette crisp appartenenti a N_k .

Queste definizioni sono le più generali che si possono dare in relazione alla struttura delle uscite di un sistema di modellamento dedicato al clustering o alla classificazione; esse completano le definizioni date al cap. 1. Si noti comunque che, a valle del tipo di risultati fornito dal tipo di logica induttiva utilizzata nell'apprendimento, è sempre possibile cambiare all'occorrenza la tipologia delle uscite mediante opportune procedure di fuzzificazione, defuzzificazione, probabilizzazione, deprobabilizzazione, e così via.

Capitolo
3

PROPRIETÀ DELLE RETI NEUROFUZZY

3.1 Premessa

Le reti neurofuzzy (NeuroFuzzy Networks - NFN) costituiscono uno dei più avanzati sistemi di modellamento attualmente utilizzati nella comunità scientifica. Esse permettono la soluzione di un generico problema di approssimazione funzionale combinando il modellamento induttivo di tipo fuzzy con le caratteristiche strutturali e di apprendimento data driven proprie delle reti neurali. Dovendo costituire un modello fuzzy del processo preso in considerazione, le NFN dovrebbero possedere la proprietà fondamentale di approssimare, con accuratezza illimitata, il legame ingresso-uscita realizzato dal processo stesso. Quando un modello fuzzy soddisfa a tale requisito si dice che esso è un *approssimatore universale*. Le condizioni sufficienti affinché una NFN sia un approssimatore universale sono stabilite dal noto *teorema di Stone-Weierstrass*.

In questo capitolo saranno messe in evidenza alcune caratteristiche che rendono le NFN particolarmente vantaggiose nella soluzione dei problemi di modellamento data driven. Alla discussione verrà dato un taglio particolare, che inquadrerà le NFN nei moderni problemi di trattamento dell'informazione e di elaborazione dei segnali. A tale ambito è tradizionalmente legato anche lo sviluppo della moderna *teoria dei circuiti*. Infatti, la necessità di ottenere un miglioramento delle prestazioni nell'elaborazione dei segnali ha costituito nel passato lo stimolo principale alla ricerca di nuovi aspetti nella teoria dei circuiti. Al giorno d'oggi, l'elaborazione dei segnali sta subendo profonde modificazioni, come manifestato dall'emergente campo

dell'*Intelligent Signal Processing* (ISP) [Kosko 1998]. Di conseguenza, è abbastanza naturale chiedersi se tale evoluzione si stia riflettendo anche nel campo della moderna teoria dei circuiti, visto che le NFN giocano nell'ambito dell'ISP lo stesso ruolo che i circuiti tradizionali giocano nei problemi classici di elaborazione dei segnali.

Certamente esistono più analogie che differenze tra l'approccio neurofuzzy e quello circuitale. Se un circuito è costituito dalla connessione di componenti ideali di pochi tipi diversi, analogamente i componenti di una NFN sono le regole fuzzy in essa utilizzate. Inoltre, il clustering nelle NFN svolge la stessa funzione delle classiche procedure di sintesi circuitale, in quanto mira alla determinazione dei parametri delle regole (ossia dei valori dei componenti). La connessione delle varie regole, realizzata dal meccanismo di inferenza induttiva, determina l'architettura della NFN. E' quindi importante notare come la procedura di sintesi, nel caso delle NFN, deve essere completata anche dall'ottimizzazione della relativa complessità strutturale (determinazione del numero di componenti-regole). E' a tale scopo che saranno proposte le procedure di ottimizzazione strutturale oggetto del lavoro di ricerca presentato in questa tesi.

Dalle architetture delle NFN, e dalle ulteriori connessioni fra esse, potrebbero successivamente essere progettati i processori per l'ISP. Il loro scopo sarebbe quello di svolgere lo stesso ruolo che ha un circuito tradizionale, ossia rappresentare un modello che implementi l'algoritmo richiesto per l'elaborazione del segnale.

3.2 Gli elementi di base delle reti neurofuzzy

Le NFN sono basate su un meccanismo induttivo di tipo fuzzy. Gli elementi di base necessari a questo scopo sono pertanto:

- le variabili fuzzy;
- la conversione tra quantità crisp e quantità fuzzy (conversione crisp-fuzzy);
- le operazioni basate sugli insiemi fuzzy;
- la struttura delle regole fuzzy.

Relativamente a questo contesto, una discussione esauriente sulle variabili fuzzy è stata già affrontata nel par. 1.7. Nel seguito saranno invece discussi gli altri punti, allo scopo di evidenziare le caratteristiche principali delle reti neurofuzzy utilizzate in questo lavoro. Si vuole precisare che la discussione che seguirà non vuole assolutamente essere esaustiva rispetto alle problematiche ed agli strumenti esistenti nel campo della logica fuzzy; per questo si rimanda a testi più specifici, come ad esempio [Jang 1997]. Nel seguito si vogliono invece evidenziare le caratteristiche essenziali della logica fuzzy, soprattutto quelle che risultano particolarmente utili nelle attività di inferenza induttiva richieste da una procedura di modellamento.

3.2.1 Conversioni crisp-fuzzy

La conversione da quantità crisp a quantità fuzzy, e viceversa, è un'operazione spesso necessaria quando si lavora con una rete NFN. Queste due operazioni saranno denotate, rispettivamente, con i termini *fuzzificazione* e *defuzzificazione*. La prima riguarda un valore crisp $x=x_0$, che può convertito in una quantità fuzzy associando ad esso una membership fuzzy (MF) di tipo impulsivo:

$$\mu(x) = \delta(x - x_0) \qquad (3.1)$$

dove $\delta(\cdot)$ è il *delta di Kronecker*, che vale 1 nel caso di argomento nullo e 0 altrove. La risultante quantità fuzzy è anche chiamata, con termine inglese, *singleton*.

Come già accennato nel par. 1.7, alle variabili fuzzy è spesso conveniente associare un'etichetta linguistica. Nel caso di valori crisp l'uso di tali etichette è evidentemente immediato. Se, ad esempio, la variabile fuzzy x fosse la "temperatura", allora l'equazione (3.1) definirebbe la situazione in cui "la temperatura è x_0". Il metodo di fuzzificazione appena descritto è caratterizzato da una semplicità che spesso lo fa preferire ad altri possibili approcci.

La defuzzificazione riguarda invece la conversione di una quantità fuzzy in un valore crisp. Ciò può essere ottenuto con riferimento ad alcuni attributi delle MF come, per esempio, l'utilizzo di un'opportuna soglia. Uno dei metodi di defuzzificazione più comuni ed affermati è il *centro di gravità* (Center Of Gravity - COG) della MF. In questo caso, la quantità crisp è data da:

$$x_o = \frac{\int x\mu(x)dx}{\int \mu(x)dx} \qquad (3.2)$$

3.2.2 Operazioni fuzzy

Le operazioni su variabili fuzzy sono ragionevoli estensioni delle operazioni usate nella tradizionale teoria degli insiemi. Se si limita l'attenzione a due quantità A e B, caratterizzate rispettivamente dalle MF $\mu_A(x)$ e $\mu_B(x)$, il generico risultato di un'operazione fuzzy sarà una quantità fuzzy C, caratterizzata da un'opportuna MF $\mu_C(x)$. Tale operazione combinerà nell'etichetta linguistica associata a C anche le etichette linguistiche che denotano A e B. Sono possibili diverse procedure per implementare gli operatori fuzzy [Jang 1997]. Nel seguito sono presentate le scelte più comuni per gli operatori fuzzy più usati:

- *Unione* (o *t-conorma*): $C = A \cup B$

  $$\mu_C(x) = \mu_A(x) \oplus \mu_B(x) = \max\{\mu_A(x), \mu_B(x)\} \qquad (3.3)$$

  Sempre con riferimento alla temperatura, se A è denotato con la variabile linguistica "alta" e B con "vicina a 200°" allora C potrebbe essere denotato con "alta o vicina a 200°". Come t-conorma, denotata col simbolo '⊕', si userà l'operatore di massimo (3.3).

- *Intersezione* (o *t-norma*): $C = A \cap B$

  $$\mu_C(x) = \mu_A(x) \otimes \mu_B(x) = \begin{cases} \min\{\mu_A(x), \mu_B(x)\} \\ \mu_A(x) \cdot \mu_B(x) \end{cases} \qquad (3.4)$$

  In questo caso, C potrebbe essere denotato con "alta e vicina a 200°". Come t-norma, denotata col simbolo '⊗', useremo l'operatore di minimo oppure il prodotto algebrico, secondo quanto indicato nella (3.4).

- *Complemento*: $C = \overline{A}$

  $$\mu_C(x) = 1 - \mu_A(x) \qquad (3.5)$$

  La temperatura di C potrebbe, in questo caso, essere denotata con "non alta". L'operatore utilizzato nel seguito sarà quello di complemento a 1 riportato nella (3.5).

3.2.3 Regole fuzzy: il modello di Mamdani e di Sugeno del 1° ordine

Le regole rappresentano i componenti di base di una rete NFN. In analogia con l'approccio circuitale all'elaborazione dei segnali e dell'informazione, pochi tipi di componenti sono usati per rappresentare le NFN. Nel seguito descriveremo i due tipi di NFN più frequentemente usate, ciascuna basata su una particolare struttura delle regole fuzzy.

Regole di Mamdani

Una regola di Mamdani è definita dalla seguente struttura (con terminologia inglese):

$$\textit{if } x_1 \text{ is } B_1, \textit{ and } x_2 \text{ is } B_2, \ldots, \textit{ and } x_n \text{ is } B_n \textit{ then } y \textit{ is } C \qquad (3.6)$$

oppure, equivalentemente[1]:

$$\textit{If } \underline{x} \text{ is } \underline{B} \textit{ then } y \text{ is } C$$

La regola può quindi essere divisa in due parti chiamate, rispettivamente, *antecedenti* e *conseguenti* della regola. Gli antecedenti riguardano ciascuno le variabili scalari fuzzy $x_j \in X_j$, j=1…n; il conseguente, in questo caso unico, riguarda la variabile scalare fuzzy $y \in Y$. Le quantità fuzzy $B_j(X_j)$, j=1...n, e C(Y) sono note per mezzo delle rispettive membership fuzzy (MF) $\mu_{B_j}(x_j)$, j=1…n, e $\mu_C(y)$.

La regola di Mamdani può essere considerata come la relazione costitutiva di un particolare componente delle NFN. Esso può essere pensato come un componente unidirezionale statico con n ingressi scalari fuzzy (o equivalentemente un ingesso vettoriale n-dimensionale fuzzy) ed un'uscita scalare fuzzy. I parametri di tale componente sono, evidentemente, quelli che compaiono nelle MF delle quantità fuzzy $B_j(X_j)$, j=1...n, e C(Y). Quando tale componente è inserito in una NFN, esso è attivato dagli ingressi alla rete o dalle uscite di altri suoi componenti (regole). Quando ciascuno degli ingressi assume un valore fuzzy B_j', j=1...n, caratterizzato dalla MF

[1] Senza entrare nel dettaglio, è possibile dimostrare [Jang 1997] che n variabili scalari fuzzy $x_j \in X_j$, j=1…n, possono essere combinate considerando un'unica variabile vettoriale $\underline{x} \in \underline{X}$, $\underline{X} = X_1 \times X_2 \times \ldots \times X_n$, che assuma il valore fuzzy $\underline{B} = B_1 \times B_2 \times \ldots \times B_n$. La MF di tale quantità è generalmente definita da $\mu_{\underline{B}}(\underline{x}) = \mu_{B_1}(x_1) \otimes \mu_{B_2}(x_2) \otimes \ldots \otimes \mu_{B_n}(x_n)$.

$\mu_{B_j'}(x_j)$, l'uscita della regola assumerà un valore fuzzy C' caratterizzato dalla MF $\mu_{C'}(y)$. Quest'ultima è ottenuta come *conseguenza* di un *ragionamento approssimato* (o *Generalized Modus Ponens*), le cui *premesse* sono costituite dalla regola di Mamdani in oggetto e dagli ingessi correnti.

Il ragionamento approssimato rappresenta nell'ambito delle NFN il meccanismo di inferenza induttiva di cui si è ampiamente discusso nei capitoli precedenti. Svariate scelte possono essere fatte riguardo agli operatori (t-norme, t-conorme, composizioni di MF, ecc.), che possono portare a risultati anche molto differenti tra loro [Jang 1997, Mendel 1995]. Nel seguito limiteremo la nostra attenzione al meccanismo di ragionamento approssimato riportato in (3.7):

$$\mu_{C'}(y) = \gamma_{\underline{B}'} \otimes \mu_C(y) \quad (3.7.a) \quad ; \qquad \gamma_{\underline{B}'} = \gamma_{B_1'} \otimes ... \otimes \gamma_{B_n'} \qquad (3.7.b)$$

$$\gamma_{B_j'} = \bigvee_{x_j} \left\{ \mu_{B_j'}(x_j) \otimes \mu_{B_j}(x_j) \right\} \ , \quad j = 1...n \qquad (3.7.c)$$

dove $\gamma_{\underline{B}'}$ è il *grado di affidabilità*, o *grado di verità*, della regola rispetto ai valori correnti B_j', j=1...n, degli ingressi. La t-norma usata nelle equazioni (3.7) è sempre la stessa, anche se ciò non è strettamente necessario. Di solito la scelta possibile per la t-norma cade tra l'operatore di minimo ('min') e quello di prodotto algebrico ('prod'). Come operatore di composizione, denotato col simbolo '$\bigvee$' in (3.7.c), sarà utilizzato l'operatore di massimo ('max').

Sulla base di quanto indicato in (3.7), l'implementazione hardware o software del componente relativo ad una regola di Mamdani può seguire lo schema di fig. 3.1. I blocchi generatori di MF sottolineano la natura fuzzy sia degli ingressi che dell'uscita. Essi non forniscono un preciso valore numerico (crisp); piuttosto, essi possono essere pensati come componenti statici con una trans-caratteristica non lineare che permette di generare tutto l'insieme continuo di valori assunti dalla MF. In fig. 3.1 è indicato in uscita anche il grado di affidabilità della regola in quanto, come vedremo nel seguito, esso è necessario nel caso di meccanismi di ragionamento più complessi.

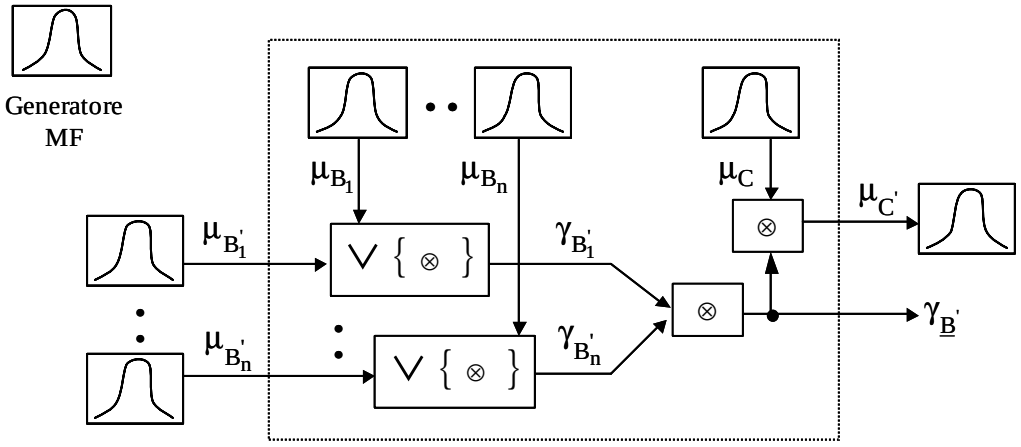


Figura 3.1 – Implementazione di una regola di Mamdani.

Un esempio di ragionamento approssimato è evidenziato in fig. 3.2 nel caso n=2 e di uso della min t-norma. Si ha in questo caso:

$$\mu_{C'}(y) = \min\left\{\gamma_{\underline{B}'},\ \mu_C(y)\right\} \quad (3.8.a) \quad ; \qquad \gamma_{\underline{B}'} = \min\left\{\gamma_{B'_1}, \gamma_{B'_2}\right\} \quad (3.8.b)$$

$$\gamma_{B'_j} = \max_{x_j}\left\{\min\left[\mu_{B'_j}(x_j),\ \mu_{B_j}(x_j)\right]\right\}\ ,\ j = 1, 2 \quad (3.8.c)$$

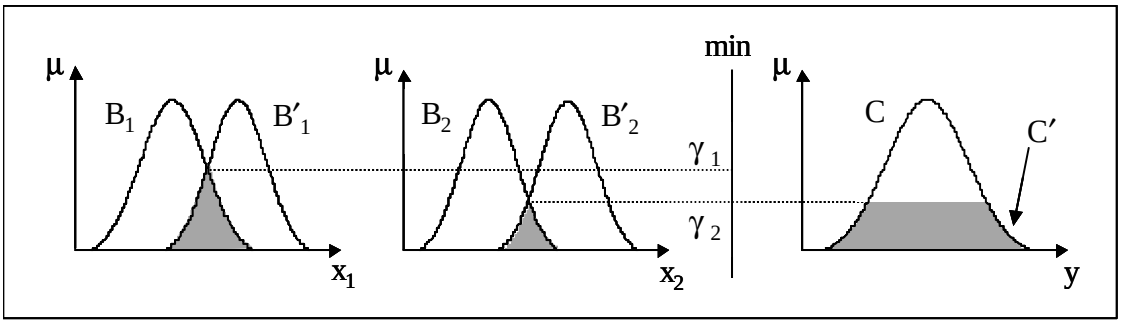


Figura 3.2 - Esempio di ragionamento approssimato nel caso di una regola e due antecedenti.

In molte situazioni pratiche, come quelle legate ai problemi classici di elaborazione dei segnali, l'uscita e gli ingressi di una regola di Mamdani possono, o devono, essere di tipo crisp. L'uscita crisp è ottenuta aggiungendo in cascata al conseguente C' un defuzzificatore come quello indicato in (3.2). Tuttavia, osservando la (3.2) e la (3.7.a), si può osservare che, quando è usata la prod t-norma e/o una $\mu_C(y)$ simmetrica, scompare nell'uscita crisp la dipendenza da $\gamma_{\underline{B}'}$:

$$y_o = \frac{\int y\mu_{C'}(y)dy}{\int \mu_{C'}(y)dy} = \frac{\int y\mu_C(y)dy}{\int \mu_C(y)dy} \qquad (3.9)$$

L'uscita crisp è, in questo caso, indipendente dagli ingressi: essa risulta una caratteristica della regola che può essere calcolata a priori senza la necessità di defuzzificatori.

Nel caso fossero presenti degli ingressi crisp, essi possono essere fuzzificati a monte della regola mediante, ad esempio, un generatore di MF di tipo singleton (3.1). Ciò conduce ad un'interessante proprietà quando tutti gli ingressi sono crisp, ossia $x_j=x_{oj}$, j=1...n. Infatti, il grado di affidabilità $\gamma_{\underline{B}'} = \gamma_{\underline{x}_o}$ potrebbe essere direttamente determinato a partire dalla (3.7.b) e dalla (3.7.c):

$$\gamma_{\underline{x}_o} = \mu_{\underline{B}}(\underline{x}_o) = \begin{cases} \min\limits_j \left\{\mu_{B_j}(x_{oj})\right\} & (3.10.a) \\ \prod\limits_{j=1}^{n} \mu_{B_j}(x_{oj}) & (3.10.b) \end{cases}$$

dove la (3.10.a) è ottenuta usando la min t-norma e la (3.10.b) usando la prod t-norma.

In fig. 3.3 sono proposte due implementazioni equivalenti della regola di Mamdani, opportunamente adattate al caso di ingressi ed uscite crisp. Al fine di semplificare l'architettura di tali componenti, l'uscita crisp è ottenuta dalla (3.9) sotto l'ipotesi, non molto restrittiva, di $\mu_C(y)$ simmetrica. In queste implementazioni non è più necessario utilizzare i generatori di MF nell'intero campo di valori delle rispettive variabili, in quanto è sufficiente valutare il valore della MF solo in corrispondenza dell'ingresso crisp. Un'ulteriore semplificazione è possibile per l'uscita, dove è necessario un semplice buffer per memorizzare il valore caratteristico y_o dato dalla (3.9).

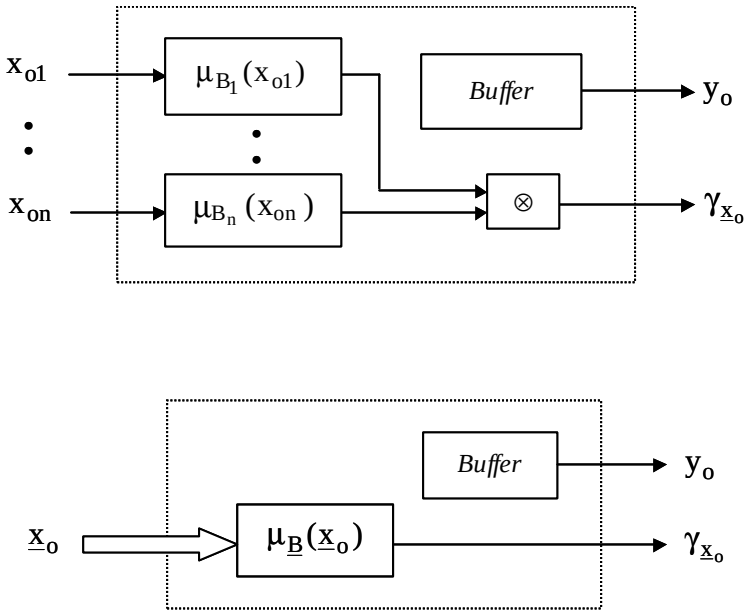


Figura 3.3 - Due implementazioni equivalenti della regola di Mamdani nel caso crisp.

Regole di Sugeno (1° ordine)

Una regola di Sugeno del 1° ordine è costituita dalla seguente relazione (con notazione inglese):

$$\textit{if } x_1 \text{ is } B_1, \textit{ and } x_2 \text{ is } B_2, \ldots, \textit{ and } x_n \text{ is } B_n \textit{ then } y = y_o = \sum_{j=1}^{n} a_j x_{oj} + a_0 \qquad (3.11)$$

o, equivalentemente:

$$\textit{if } \underline{x} \text{ is } \underline{B} \textit{ then } y = y_o = \sum_{j=1}^{n} a_j x_{oj} + a_0$$

La caratteristica essenziale di questo tipo di regole è la natura crisp del conseguente e, al contrario di $\mu_C(y)$ nelle regole di Mamdani, la dipendenza dell'uscita dal valore crisp degli ingressi correnti. Nel seguito considereremo solo conseguenti lineari (regole del 1° ordine), anche se i risultati che si otterranno sono validi per una qualsiasi espressione $y=f(\underline{x}_o)$ adottata per il conseguente (usualmente un polinomio di grado r).

E' possibile osservare che le considerazioni ed i risultati relativi alla (3.7) per le regole di Mamdani restano validi anche nel caso delle regole di Sugeno. Infatti, la

(3.11) è equivalente alla (3.6) laddove si consideri per la MF di uscita $\mu_C(y) = \delta(y - y_o)$, dove y_o è il valore indicato nella (3.11). Da questa osservazione segue che, essendo gli ingressi ad una regola di Sugeno di tipo crisp, il grado di affidabilità $\gamma_{\underline{x}_o}$ della regola può essere calcolato tramite la (3.10). Inoltre, essendo $\mu_C(y)$ un singleton, dalla (3.7.a) segue che $\mu_{C'}(y) = \gamma_{\underline{x}_o}\delta(y - y_0)$ sia nel caso di min t-norma che di prod t-norma. L'evidente simmetria di $\mu_{C'}(y)$ conferma che il suo COG coincide proprio con l'uscita crisp y_o definita nella regola (3.11).

Da quanto appena detto risulta evidente come l'implementazione in una NFN del componente relativo ad una regola di Sugeno è simile a quella della regola di Mamdani nel caso crisp. Due versioni equivalenti del componente di Sugeno sono illustrate in fig. 3.4; contrariamente alla regola di Mamdani, i generatori di MF devono essere usati quando è richiesta un'uscita fuzzy, mentre i defuzzificatori sono necessari quando sono presenti ingressi fuzzy.

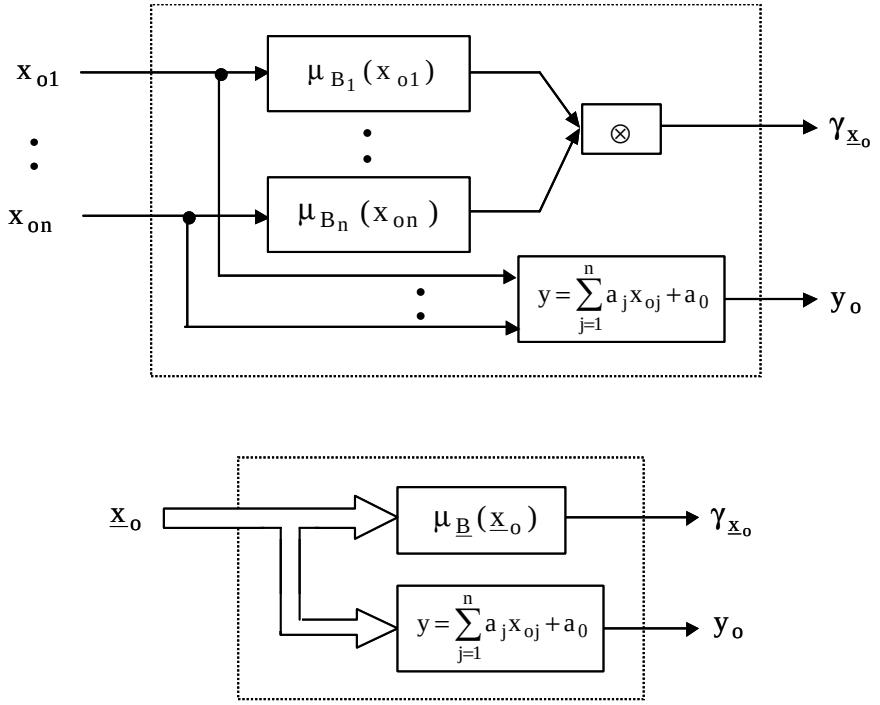


Figura 3.4 - Due implementazioni equivalenti della regola di Sugeno del 1° ordine.

3.3 Architetture delle reti neurofuzzy

L'architettura di una NFN è strettamente legata al ragionamento approssimato utilizzato, ossia al meccanismo di inferenza induttiva che porta alle conclusioni (uscite) sulla base delle premesse (regole ed ingressi). Le opzioni possibili in sede di progetto di una NFN riguardano: il numero di regole; la loro struttura (Mamdani, Sugeno, ecc.); la fuzzificazione/defuzzificazione degli ingressi e/o delle uscite; le operazioni fuzzy utilizzate per la derivazione delle conclusioni di ogni regola (discusse, ad esempio, nel par. 3.2.3) e delle conclusioni complessive. E' da tutte queste opzioni che dipende il particolare modellamento realizzato da una NFN. Nel seguito limiteremo la nostra attenzione ai due più comuni tipi di NFN, le quali sono basate sul tipo di regole analizzate nel paragrafo precedente.

3.3.1 Le reti di Mamdani

Le NFN di Mamdani sono costituite da M regole (3.6):

$$\textit{If } \underline{x} \text{ is } \underline{B}^{(k)} \textit{ then } y \text{ is } C^{(k)}, \quad k=1...M \tag{3.12}$$

In questo contesto considereremo la presenza di <u>tutti</u> gli ingressi x_j, j=1…n, in <u>tutte</u> le regole; tuttavia, il meccanismo di inferenza (3.7) può essere facilmente generalizzato al caso in cui alcuni ingressi non siano presenti in alcune regole [Jang 1997].

Il ragionamento approssimato seguito in questo caso determina l'uscita complessiva della rete $\mu_{C'}(y)$ come composizione delle uscite $\mu_{C'(k)}(y)$, k=1...M, delle singole regole. La composizione di queste uscite, ciascuna ottenuta mediante la (3.7), è ottenuta nella logica fuzzy mediante la t-conorma. Utilizzando l'operatore di massimo si avrà:

$$\mu_{C'}(y) = \mu_{C'(1)}(y) \oplus \mu_{C'(2)}(y) \oplus ... \oplus \mu_{C'(M)}(y) = \max_k \left\{ \mu_{C'(k)}(y) \right\} \tag{3.13}$$

L'implementazione della NFN risultante da tale meccanismo di inferenza è illustrata in fig. 3.5. Un esempio di ragionamento approssimato implementato da tale rete è mostrato in fig. 3.6 nel caso n=2, M=2 e dell'uso di min t-norma.

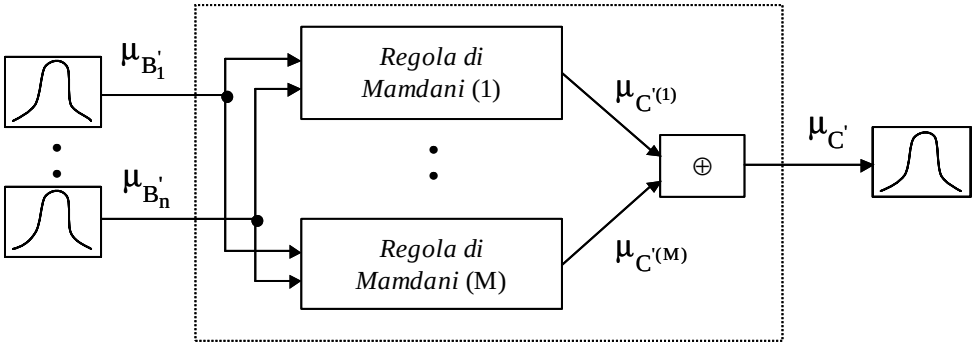


Figura 3.5 – Implementazione di una rete neurofuzzy di Mamdani.

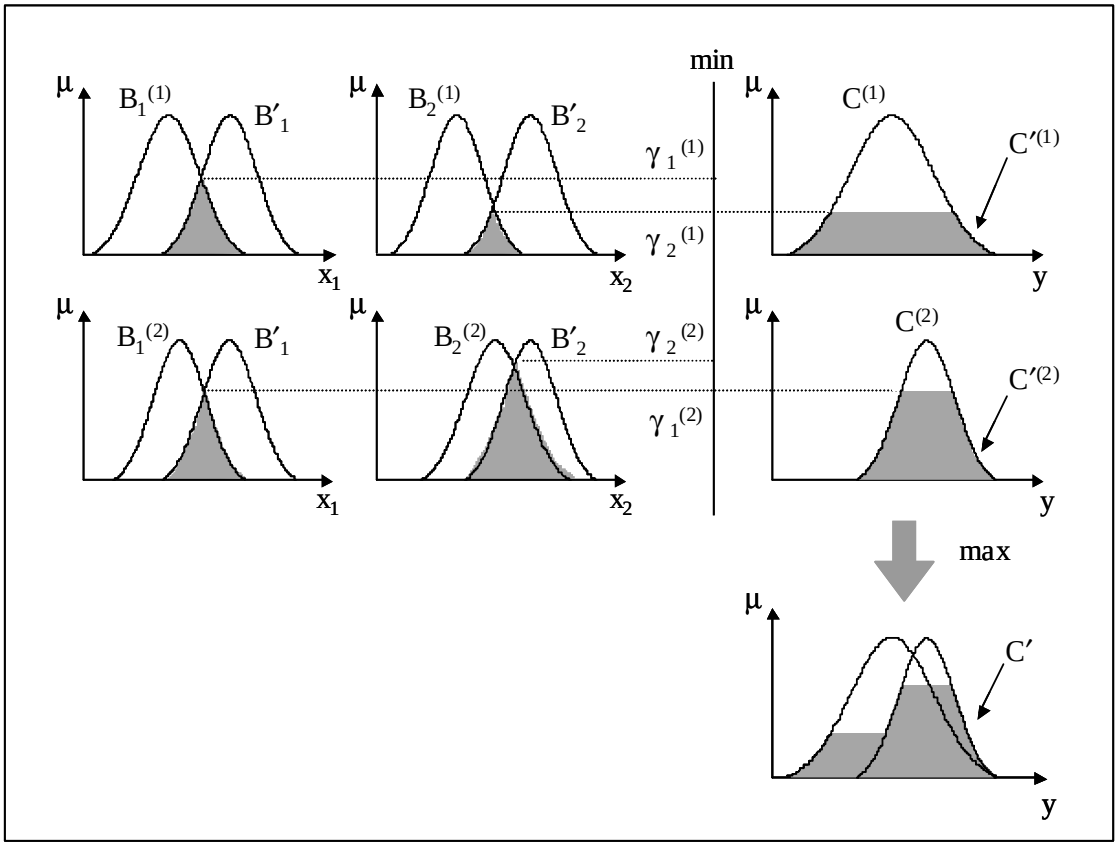


Figura 3.6 – Ragionamento approssimato di una NFN di Mamdani con due regole e due ingressi.

La defuzzificazione dell'uscita può essere ottenuta mediante il COG di $\mu_{C'}(y)$. Tuttavia, si può ottenere un risultato più utile ed interessante se vengono aggiunti ulteriori vincoli al modello. Il primo di essi è la scelta di MF simmetriche per le uscite $\mu_{C^{(k)}}(y)$ delle regole. In questo caso, come già detto, se si usa il COG per determinare l'uscita crisp $y_o^{(k)}$ di ogni regola, quest'ultima risulterà indipendente dagli ingressi e

coinciderà con il centro di simmetria di $\mu_{C^{(k)}}(y)$. Un'ipotesi ragionevole che si può fare è che vengano utilizzate MF simmetriche che decrescano monotonicamente all'allontanarsi dal centro di simmetria della MF, per cui si può assumere[2]:

$$\mu_{C^{(k)}}(y_o^{(k)}) = 1 \qquad (3.14)$$

Il secondo vincolo è imposto dall'uso di un metodo leggermente differente per la normalizzazione dell'uscita complessiva y_o. Quest'ultima è ottenuta mediante la *media dei centri* (Center Average – CA) definita come

$$y_o = \frac{\sum_{k=1}^{M} y_o^{(k)} \mu_{C'^{(k)}}(y_o^{(k)})}{\sum_{k=1}^{M} \mu_{C'^{(k)}}(y_o^{(k)})} \qquad (3.15)$$

Dalla (3.7.a) e dalla (3.14) segue che la (3.15) può essere riscritta come

$$y_o = \frac{\sum_{k=1}^{M} y_o^{(k)} \gamma_{\underline{B}'}^{(k)}}{\sum_{k=1}^{M} \gamma_{\underline{B}'}^{(k)}} \qquad (3.16)$$

dove $\gamma_{\underline{B}'}^{(k)}$ è il grado di affidabilità rispetto agli ingressi della k-esima regola, k=1…M, ottenuto dalla (3.7.b) e dalla (3.7.c) .

La (3.16) evidenzia che, sotto le ipotesi fornite, l'uscita crisp di una NFN di Mamdani può essere pensata come un'espansione, pesata dai termini $y_o^{(k)}$, delle seguenti *funzioni di base*:

$$\alpha_{\underline{B}'}^{(k)} = \frac{\gamma_{\underline{B}'}^{(k)}}{\sum_{q=1}^{M} \gamma_{\underline{B}'}^{(q)}} \quad , \qquad k=1\ldots M \qquad (3.17)$$

Ancora una volta, nel caso di ingressi crisp $\underline{x}=\underline{x}_o$, il grado di affidabilità $\gamma_{\underline{B}'}^{(k)} = \gamma_{\underline{x}_o}^{(k)} = \mu_{\underline{B}^{(k)}}(\underline{x}_o)$ di ciascuna regola può essere determinato direttamente dalle (3.10).

[2] Si sta supponendo l'uso di MF normalizzate aventi almeno un valore di y per il quale esse valgano 1.

In conclusione, si riassumono di seguito tutte le precedenti ipotesi che, quando usate nel ragionamento approssimato, conducono alla particolare NFN di Mamdani di tipo crisp mostrata in fig. 3.7:

- Metodo di fuzzificazione: singleton;
- Defuzzificazione delle singole regole: centro di gravità (COG);
- Defuzzificazione uscita complessiva: media dei centri (CA);
- MF dei conseguenti di ogni regola: simmetriche;
- t-conorma: massimo;
- t-norma: unica, o minimo o prodotto.

Le funzioni $\alpha_{\underline{B}'}^{(k)} = \alpha_{\underline{x}_o}^{(k)}$, k=1…M, sono ottenute in fig. 3.7 dai blocchi di normalizzazione che calcolano la (3.17).

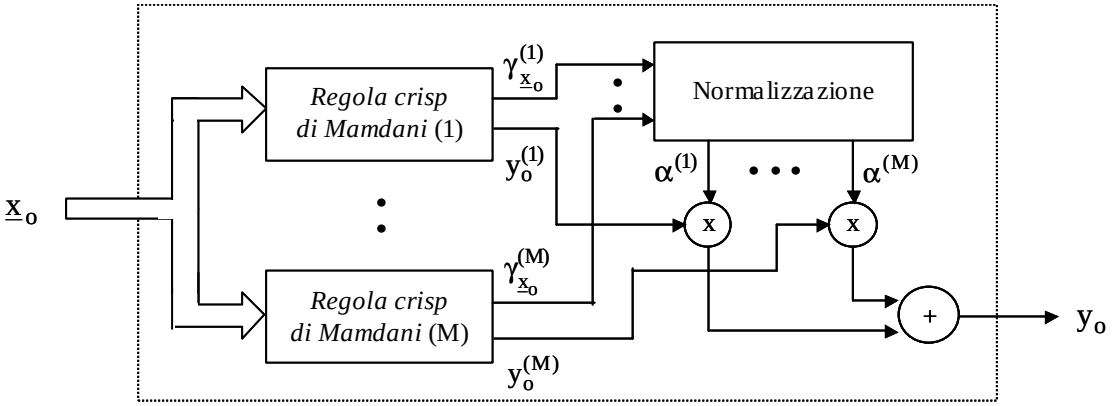


Figura 3.7 – Implementazione di un particolare modello di NFN di Mamdani di tipo crisp.

La rete di Mamdani schematizzata in fig. 3.7 possiede una proprietà fondamentale: essa risulta un approssimatore universale quando le MF sono gaussiane ed è utilizzata la prod t-norma [Wang 1992, Park 1991]. Le funzioni di base risultanti dalla (3.10.b) e dalla (3.17) diventano quindi:

$$\alpha_{\underline{x}_o}^{(k)} = \frac{\prod_{j=1}^{n} \exp\left\{-0.5\left[(x_{oj} - c_{oj}^{(k)})/\sigma_j^{(k)}\right]^2\right\}}{\sum_{q=1}^{M}\prod_{j=1}^{n} \exp\left\{-0.5\left[(x_{oj} - c_{oj}^{(q)})/\sigma_j^{(q)}\right]^2\right\}}, \quad k=1\ldots M \qquad (3.18)$$

dove $c_{oj}^{(k)}$ e $\sigma_j^{(k)}$, j=1…n, k=1…M, sono, rispettivamente, la media e la deviazione standard della MF gaussiana relativa al j-esimo ingresso nella k-esima regola.

3.3.2 Le reti ANFIS (Sugeno del 1° ordine)

Le NFN basate su regole di Sugeno del 1° ordine sono anche dette reti ANFIS (Adaptive NeuroFuzzy Inference System). Esse sono costituite da M regole di Sugeno (3.11):

$$\textit{if } \underline{x} \text{ is } \underline{B}^{(k)} \textit{ then } \quad y = y_o^{(k)} = \sum_{j=1}^{n} a_j^{(k)} x_j + a_0^{(k)}, \quad k=1...M \qquad (3.19)$$

Come per le singole regole di Sugeno, la rete ANFIS lavora con ingressi ed uscite crisp. Essendo $\mu_{C^{(k)}}(y) = \delta(y - y_o^{(k)})$, le MF dei conseguenti soddisfano a tutte le ipotesi precedentemente fatte per le NFN di Mamdani di tipo crisp; in particolare, esse sono simmetriche, il COG è proprio l'uscita crisp $y_o^{(k)}$ della regola ed inoltre è soddisfatta la (3.14). La rete ANFIS ha pertanto la stessa struttura della rete di Mamdani di tipo crisp in fig. 3.7, con i componenti relativi alle regole di Sugeno invece che a quelle di Mamdani. Riassumendo, l'uscita complessiva y_o è ottenuta a seguito delle (3.10), (3.16), (3.17), ossia:

$$y_o = \sum_{k=1}^{M} \alpha_{\underline{x}_o}^{(k)} y_o^{(k)} \qquad (3.20.a)$$

$$y_o^{(k)} = \sum_{j=1}^{n} a_j^{(k)} x_j + a_0^{(k)} \qquad (3.20.b) \quad ; \qquad \alpha_{\underline{x}_o}^{(k)} = \frac{\mu_{\underline{B}^{(k)}}(\underline{x}_o)}{\sum_{q=1}^{M} \mu_{\underline{B}^{(q)}}(\underline{x}_o)} \qquad (3.20.c)$$

Nel prossimo capitolo torneremo ampiamente sull'impiego delle reti ANFIS come sistemi di modellamento neurofuzzy, in quanto è su di esse che è stata incentrata gran parte dell'attività di ricerca illustrata in questa tesi. Per la stessa ragione, i concetti

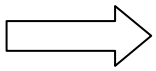
esposti nel seguito di questo capitolo saranno discussi con particolare enfasi per ciò che riguarda le reti ANFIS.

3.4 Metodi di clustering per la sintesi delle reti neurofuzzy quando è disponibile un'informazione numerica

I componenti di una NFN, intesa come sistema hardware o software, sono le regole. Esse sono determinate a partire dalle informazioni riguardanti il processo da modellare. Due sono i tipi di sorgenti informative possibili: un'informazione di tipo linguistico fornita da sistemi esperti, tipicamente soggetti umani, riguardo alle proprietà della funzione ingresso-uscita legata al processo incognito; un'informazione numerica sotto forma di campionamento ingresso-uscita del processo (v. par. 1.3).

L'informazione di tipo linguistico è tipicamente fornita sotto forma di regole di Mamdani (3.6); in tali casi, è evidente come risulti particolarmente vantaggioso ed immediato sintetizzare una NFN di Mamdani sulla base di tali informazioni [Frattale Mascioli 2001]. Nel caso sia necessaria un'elaborazione di tipo crisp da parte della NFN, è sempre possibile determinare una rete di Mamdani di tipo crisp sulla base della procedura vista al par. 3.3.1. L'incorporazione dell'informazione linguistica nella sintesi delle NFN sarà brevemente discussa nel par. 3.5 ed illustrata attraverso un tipico esempio di modellamento.

Nella stragrande maggioranza dei problemi di modellamento ingegneristici è invece quasi sempre presente, a volte in modo esclusivo, un'informazione di tipo numerico. Supponiamo[3] che il processo da modellare realizzi un legame ingresso-uscita $y=f(\underline{x})$, $f:\Re^n \rightarrow \Re$. L'informazione numerica sarà allora presente sotto forma di un insieme di P campioni (data set) di coppie $(\underline{x}_i, y_i)$, i=1…P.



Nota: per semplicità notazionale, d'ora in poi con $\underline{x}_i$ si indicherà la i-esima variabile vettoriale crisp (precedentemente denotata con $\underline{x}_{oi}$) e con x_{ij} la sua j-esima componente scalare. Un discorso analogo vale, ovviamente, anche per $\underline{y}_i$.

[3] Stiamo supponendo, per semplicità, uno spazio di uscita scalare per il processo. Quanto si dirà è comunque facilmente generalizzabile al caso vettoriale.

E' importante notare che, nel caso di dati numerici, è abbastanza naturale pensare alla sintesi di NFN ANFIS oppure di Mamdani nella variante crisp. In tali situazioni risulta spesso superfluo, nella procedura di sintesi, determinare le MF marginali μ_{B_j} per ogni dimensione X_j dello spazio di ingresso; piuttosto, è possibile lavorare sugli interi vettori di ingresso, andando a determinare le MF complessive $\mu_{\underline{B}}$ nell'intero spazio $\underline{X}$. Tale procedura sottintende implicitamente la composizione, ottenuta ad esempio mediante le (3.10), delle MF marginali.

Sono possibili diverse scelte per sintetizzare le NFN sulla base di un data set di P coppie $(\underline{x}_i, y_i)$, i=1…P. La procedura più semplice è basata sull'associazione diretta di ogni regola ad ogni coppia [Mendel 1995, Wang 1994]. Ad esempio, potremmo avere P regole di Mamdani del tipo

$$(\underline{x}_i, y_i) \quad \rightarrow \quad \textit{If } \underline{x} \textit{ is } \underline{B}^{(i)} \textit{ then } y \textit{ is } C^{(i)} \quad i=1...P$$

dove $\mu_{\underline{B}^{(i)}}$ corrisponde alla fuzzificazione (ad esempio singleton) di $\underline{x}_i$ e $\mu_{C^{(k)}}$ alla fuzzificazione di y_i. Se il problema fosse ben posto, i dati numerici dovrebbero contenere un'informazione ridondante riguardo alla funzione da modellare, al fine di compensare gli effetti dovuti ai disturbi o al rumore eventualmente presenti. Il numero di regole dovrebbe quindi essere ridotto da P ad M, così da eliminare quelle che potrebbero risultare identiche o contraddittorie. Tuttavia, nel caso appena illustrato, la procedura di fuzzificazione, così come la riduzione delle regole, potrebbe risultare molto critica ed inefficiente quando il numero P di esempi è elevato. E' quindi intuibile come questo processo di sintesi diretta potrebbe risultare facilmente inadeguato, soprattutto quando la distribuzione dei dati è ampiamente irregolare (e vedremo nel cap. 5 che questo accade spesso in particolari problemi di modellamento come quelli di predizione).

I problemi evidenziati rendono più attrattive quelle procedure di sintesi basate sul clustering del data set $(\underline{x}_i, y_i)$, i=1…P. Il clustering riduce la ridondanza direttamente nello spazio (o negli spazi) in cui esso è applicato, in modo da permettere la determinazione diretta delle regole più significative a partire dai cluster che modellano il data set. Ogni regola, quindi, si manifesterà in tali spazi per mezzo di un insieme strutturato di punti (un cluster, appunto). Sono possibili tre tipi di spazi in cui applicare il clustering: ingresso, uscita ed ingesso-uscita. Questa scelta influisce

pesantemente sulle caratteristiche della procedura di modellamento della NFN e sulle sue conseguenti prestazioni [Guillame 2001]. Il clustering nello spazio di ingresso (uscita) permette la "induzione" delle regole fuzzy mediante la proiezione nello spazio di uscita (ingresso) dei cluster creati nello spazio di ingresso (uscita). Vedremo nei prossimi paragrafi che tale procedura è caratterizzata da una serie di difetti, principalmente dovuti all'assunzione che caratteristiche simili in uno spazio lo siano anche nell'altro. Al contrario, il clustering nello spazio ingesso-uscita tenta di sfruttare pienamente l'informazione presente nei dati al fine di determinare in modo più accurato la struttura del processo da modellare.

Nel seguito sono illustrate le linee guida che caratterizzano le procedure basate sul clustering per la sintesi di una NFN. Riguardo agli algoritmi di clustering tradizionalmente usati per tale scopo, si rimanda a testi più specifici come [Jang 1997, Jain 1988, Bezdek 1993, Davies 1979, Frattale Mascioli 1997, Frattale Mascioli 1999, Frattale Mascioli 2000, Frattale Mascioli 2000(bis), Kohonen 1995, Kohonen 1996, Simpson 1993].

3.4.1 Clustering nello spazio di ingresso

La strategia di clustering adottata in questo caso fa uso dei soli dati nello spazio di ingresso $\underline{X}\equiv\Re^n$. Sia M il numero di cluster $\Gamma_{\underline{X}}^{(k)}$, k=1…M, determinati in tale spazio. Ogni cluster sarà associato ad una regola differente, essendo $\mu_{\underline{B}^{(k)}}(\underline{x})$ la MF complessiva determinata sulla base della struttura del k-esimo cluster. A questo punto entra in gioco un meccanismo di inferenza induttiva, in quanto ogni cluster $\Gamma_{\underline{X}}^{(k)}$ nello spazio di ingresso *induce* un cluster corrispondente $\Gamma_{Y}^{(k)}$ nello spazio di uscita $Y\equiv\Re$. In altre parole, se il pattern $\underline{x}_i$ appartiene[4] a $\Gamma_{\underline{X}}^{(k)}$ allora y_i appartiene a $\Gamma_{Y}^{(k)}$. I coefficienti $a_j^{(k)}$, j=0…n, relativi al conseguente della regola k-esima di una rete

[4] Il concetto di appartenenza di un pattern ad un cluster può essere inteso in modo esclusivo o non esclusivo secondo quanto definito al par. 2.9. Ove necessario, tale concetto sarà nel seguito ulteriormente approfondito.

ANFIS possono essere determinati sulla base di $\Gamma_{\underline{X}}^{(k)}$ e del cluster indotto $\Gamma_{Y}^{(k)}$. Una qualsiasi tecnica ai minimi quadrati può essere usata in questo caso, vedremo nel seguito un esempio a tale riguardo.

La sintesi basata sul clustering nello solo spazio di ingresso dipende in modo critico dalle proprietà di regolarità della funzione ingresso-uscita $y=f(\underline{x})$ legata al processo. Infatti, si è implicitamente assunto che punti vicini nello spazio di ingresso (e quindi appartenenti allo stesso cluster $\Gamma_{\underline{X}}^{(k)}$) siano mappati dal processo in punti vicini nello spazio di uscita (e quindi appartenenti allo stesso cluster indotto $\Gamma_{Y}^{(k)}$). Sfortunatamente, vedremo nei prossimi capitoli che, in molte applicazioni reali, spesso non si può fare affidamento su tale proprietà di regolarità della funzione da modellare. Inoltre, i cluster indotti nello spazio di uscita potrebbero non riflettere la reale struttura di tale funzione, dato che la procedura di clustering non tiene conto dei valori di uscita disponibili nel data set. Si vuole peraltro far notare che un tale approccio non è proprio solo delle NFN. Esistono anche altri paradigmi neurali, come le Radial Basis Function (RBF), che fanno uso di simili procedure di modellamento [Park 1991, Jang 1993].

3.4.2 Clustering nello spazio di uscita

Questa strategia di clustering considera solo il data set nel suo spazio di uscita Y, dove sono determinati M cluster $\Gamma_{Y}^{(k)}$, k=1…M. In modo simile al precedente, ciascuno di essi *induce* il corrispondente cluster $\Gamma_{\underline{X}}^{(k)}$ nello spazio di ingresso: se y_i appartiene a $\Gamma_{Y}^{(k)}$ allora $\underline{x}_i$ appartiene a $\Gamma_{\underline{X}}^{(k)}$. La MF complessiva $\mu_{\underline{B}^{(k)}}(\underline{x})$ nello spazio di ingresso può essere determinata sulla base della struttura del cluster indotto, mentre i coefficienti di uscita $a_j^{(k)}$ di una rete ANFIS potrebbero essere determinati con una tecnica ai minimi quadrati, in modo simile al caso precedente.

L'approccio al clustering nello spazio di uscita permette di determinare solo parzialmente la reale struttura del processo. La NFN che ne deriva potrebbe risultare inaccettabile, vista la possibile presenza di regole contraddittorie aventi antecedenti

simili ma conseguenti differenti. Un tale inconveniente potrebbe essere causato da una limitata risoluzione con cui vengono modellati i dati nello spazio di ingresso. Questa situazione è evidentemente palesata nel seguente esempio. Consideriamo un data set ottenuto dal campionamento uniforme nello spazio di ingresso del seguente processo:

$$y = \begin{cases} 0.01 \cdot |\sin(2x)| & \pi < |x| \le 2\pi \\ 0.8 + 0.01 \cdot \cos(x) & 0 \le |x| \le \pi \end{cases}$$

Il clustering nello spazio di uscita dovrebbe ragionevolmente produrre due cluster, aventi come centroidi[5] i valori in 0.0 e 0.8. A causa del campionamento uniforme e della perfetta simmetria del processo, i due cluster indotti nello spazio di ingresso avranno lo stesso centroide in 0.0. Si è così generata una situazione di "confusione" tra le due regole sintetizzate, che potrebbe essere risolta considerando due cluster distinti nello spazio d'ingresso indotti dallo stesso cluster di uscita relativo al valore 0.0. Per esempio, tali cluster potrebbero avere i centroidi collocati, rispettivamente, in $-\frac{3}{2}\pi$ e $\frac{3}{2}\pi$. Ciò implica quindi la presenza di ben tre cluster nello spazio di ingresso e quindi una risoluzione maggiore richiesta al modellamento in tale spazio.

3.4.3 Clustering nello spazio congiunto ingresso-uscita

Vedremo ampiamente nel seguito che l'uso di tale strategia di clustering è fondamentale per superare i problemi che derivano dai precedenti approcci al clustering. Sono possibili svariate scelte riguardo alla possibile procedura di clustering da operare nello spazio congiunto ingresso-uscita. Illustriamo innanzitutto due tra le più immediate soluzioni.

1. La scelta più semplice consiste nel considerare proprio lo spazio congiunto $\underline{Z}=\underline{X}\times Y$, ossia nel sottoporre al clustering proprio i vettori $\underline{z}_i=[\underline{x}_i\ y_i]$, i=1…P. Il clustering produrrà quindi M cluster $\Gamma_{\underline{Z}}^{(k)}$, k=1…M, nello spazio congiunto $\underline{Z}$;

[5] Il centroide è un punto caratteristico scelto per rappresentare complessivamente la struttura di un cluster. Diverse scelte sono possibili come, ad esempio, la media dei punti contenuti nel cluster stesso.

l'opportuna proiezione di ciascuno di essi determinerà i corrispondenti cluster $\Gamma_{\underline{X}}^{(k)}$ e $\Gamma_{Y}^{(k)}$ nello spazio di ingresso e di uscita, rispettivamente. Al solito, le MF ed i coefficienti di una rete ANFIS possono poi essere determinati come nei casi precedenti.

2. Un'altra tecnica di sintesi nella spazio congiunto potrebbe, ad esempio, utilizzare due procedure di clustering distinte nello spazio di ingresso e di uscita. In questo modo, una coppia $(\underline{x}_i, y_i)$ appartiene al cluster congiunto formato dalla coppia di cluster $\left(\Gamma_{\underline{X}}^{(k)}, \Gamma_{Y}^{(k)}\right)$ se e solo se $\underline{x}_i$ appartiene a $\Gamma_{\underline{X}}^{(k)}$ e *contemporaneamente* y_i appartiene a $\Gamma_{Y}^{(k)}$, ciascuno *separatamente* nei propri spazi.

Anche se definite nello spazio congiunto, le procedure di clustering ora suggerite risultano spesso troppo semplici rispetto al problema di modellamento da risolvere. Nel primo caso, infatti, i pattern appartenenti allo stesso $\Gamma_{\underline{Z}}^{(k)}$ potrebbero non essere vicini né nello spazio di ingresso né in quello di uscita, ottenendo quindi una struttura inadeguata per $\Gamma_{\underline{X}}^{(k)}$ e $\Gamma_{Y}^{(k)}$. Nel secondo caso, invece, l'informazione congiunta viene sfruttata in modo molto limitato, per cui la funzione approssimante realizzata dalla NFN potrebbe risultare irregolare e non riflettere l'effettiva struttura del processo.

Tutti i problemi illustrati sia in questo che nei paragrafi precedenti hanno stimolato lo sviluppo di un nuovo paradigma di sintesi delle reti ANFIS. Tale metodo, basato sul clustering nello spazio congiunto ingresso-uscita, sarà presentato nel successivo capitolo e sarà comparato con le procedure di sintesi più classiche ed affermate nella letteratura tecnica del settore, delle quali sarà comunque fornita una breve illustrazione. La procedura di sintesi proposta si ispira alle tecniche di modellamento fuzzy dette "fuzzy C-regression model" [Hathaway 1993, Kim 1997, Kim 1998]. Mediante tale procedura è possibile determinare dei cluster aventi la struttura di iperpiani, ciascuno dei quali rappresenta la parte conseguente della relativa regola di Sugeno del 1° ordine. Oltre alla procedura di clustering nello spazio congiunto, la procedura di sintesi della rete ANFIS è anche accompagnata dalla soluzione di un

problema di classificazione fuzzy, opportunamente impostato nello spazio di ingresso al fine di determinare le MF degli antecedenti delle regole.

3.5 Incorporazione dell'informazione linguistica nei metodi di sintesi delle reti neurofuzzy

Da quanto detto nei par. 1.7 e 3.2, in relazione alla possibile equivalenza linguistica delle variabili e delle regole fuzzy, risulta evidente come esistano alcuni modelli neurofuzzy, come le reti di Mamdani, che sono particolarmente adatti ad incorporare direttamente un'informazione di tipo linguistico. Quest'ultima, infatti, è fornita da esperti attraverso delle sentenze che solitamente assumono la forma proprio di regole di Mamdani[6]. L'architettura di una rete di Mamdani sintetizzata da un'insieme di regole (3.6) fornite da esperti segue, ovviamente, lo schema di fig. 3.5 o, nel caso crisp, quello di fig. 3.7. Una situazione analoga si potrebbe avere per le reti ANFIS, nel caso in cui le regole fornite dagli esperti fossero solo parzialmente linguistiche. Ciò potrebbe accadere, ad esempio, se ai conseguenti di tipo numerico (i coefficienti $a_j^{(k)}$) fossero affiancati degli antecedenti di tipo linguistico.

Più interessante è il caso in cui siano presenti entrambe le sorgenti informative: quella numerica e quella linguistica. In tal caso, la NFN sarebbe composta da due insiemi di regole: quelle originate dal clustering del data set relativo al campionamento numerico ingresso-uscita del processo; quelle proposte dagli esperti in forma linguistica. E' possibile, in questo caso, *pesare* differentemente le uscite derivanti dai due insiemi di regole, se essi possiedono un differente grado di affidabilità o consistenza. La soluzione più semplice è quella di pesare i due insiemi di regole mediante i fattori α e $(1-\alpha)$, rispettivamente, con $(0 \le \alpha \le 1)$. L'architettura della NFN risultante potrebbe quindi essere quella illustrata in fig. 3.8. Le M_{num} regole della rete di Mamdani di tipo crisp sono legate all'informazione numerica e quindi determinate con uno dei metodi di clustering prima discussi. Le M_{lin} regole della rete

[6] A volte l'informazione non può essere direttamente traslata in regole fuzzy come quelle di Mamdani [Abu-Mustafa 1995]. Sono queste le situazioni in cui il processo deduttivo seguito dagli esperti per determinare le regole potrebbe essere stato alquanto complicato.

di Mamdani classica (cioè con ingressi ed uscite fuzzy) sono invece quelle derivanti dall'informazione linguistica.

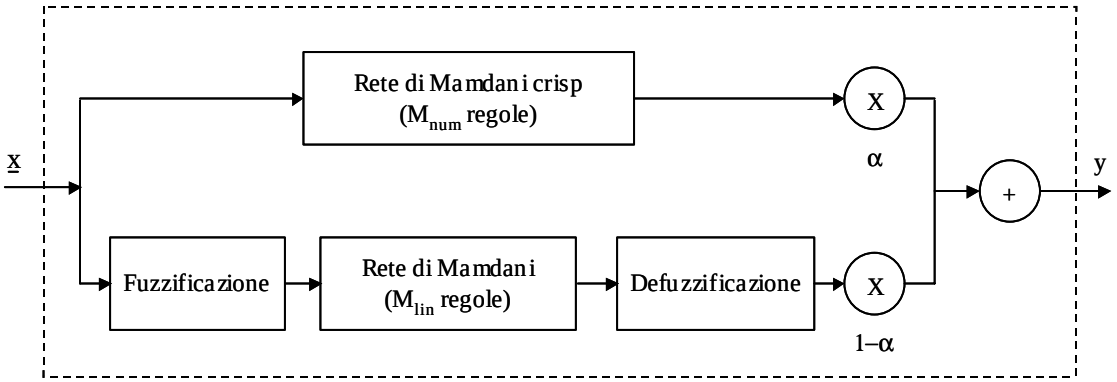


Figura 3.8 - Rete di Mamdani crisp ottenuta da informazioni sia numeriche che linguistiche.

Dalle precedenti considerazioni si può notare come le regole linguistiche rappresentino un'informazione equivalente a quella di un numero opportuno di esempi numerici. La possibilità di aggiungere, o addirittura sostituire, dati numerici con quelli linguistici è molto importante negli attuali problemi di modellamento. Infatti, i dati numerici sono spesso difficili e/o costosi da ottenere (si pensi, ad esempio, al costo che ha l'acquisizione di serie storiche legate all'andamento di titoli finanziari); inoltre, essi sono sempre contaminati da rumore in modo più o meno marcato, a seconda del sistema di campionamento utilizzato.

Vediamo ora un tipico esempio di come la procedura di sintesi coadiuvata dalle informazioni linguistiche possa rendere una rete NFN, in particolare una rete di Mamdani, più efficiente e robusta rispetto ai problemi ora evidenziati. Si tratta di approssimare la legge di controllo di una palla su una trave (fig. 3.9); la trave è capace di ruotare nel piano verticale applicando una coppia al centro di rotazione, la palla è libera di scorrere lungo la trave. Il controllo u(t) è esercitato in termini di accelerazione $\ddot{\theta}$ dell'angolo d'inclinazione della trave. Il problema è quindi quello di progettare u(t) in modo che la palla converga asintoticamente verso l'origine (r=0) a prescindere dall'inclinazione della trave e dalla posizione iniziale della palla. Il sistema è non lineare ed è descritto da quattro variabili di stato:

$$x_1 = r \ , \quad x_2 = \dot{r} \ , \quad x_3 = \theta \ , \quad x_4 = \dot{\theta}$$

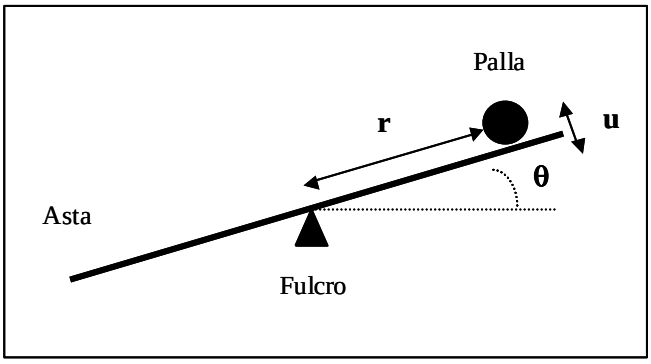


Figura 3.9 – Il sistema fisico su cui modellare la legge di controllo.

La legge di controllo che ottiene il risultato voluto può essere facilmente determinata per via analitica. In [Wang 1992] è dimostrato che lo stesso risultato è ottenibile anche usando un numero limitato di esempi numerici e quattro regole linguistiche. In questo caso noi considereremo anche un data set numerico contaminato da rumore, in modo da rendere la situazione più realistica. L'informazione disponibile è quindi costituita da:

- un data set numerico DS1 contenente 100 esempi della legge di controllo u(t), determinati analiticamente come in [Wang 1992];
- un data set rumoroso DS2, ottenuto aggiungendo a DS1 rumore additivo gaussiano a media nulla e varianza pari a 0.5;
- quattro regole linguistiche fuzzy, riportate in tab. 3.1 con terminologia inglese, le cui MF sono mostrate in tab. 3.2.

Tabella 3.1 – Informazione linguistica relativa alla legge di controllo.

If r is *positivo* and $\dot{r}$ is *vicino a zero* and θ is *positivo* and $\dot{\theta}$ is *vicino a zero* then u is *negativo*
If r is *negativo* and $\dot{r}$ is *vicino a zero* and θ is *negativo* and $\dot{\theta}$ is *vicino a zero* then u is *positivo*
If r is *positivo* and $\dot{r}$ is *vicino a zero* and θ is *negativo* and $\dot{\theta}$ is *vicino a zero* then u is *grande positivo*
If r is *negativo* and $\dot{r}$ is *vicino a zero* and θ is *positivo* and $\dot{\theta}$ is *vicino a zero* then u is *grande negativo*

Tabella 3.2 – MF utilizzate per le variabili linguistiche in tab. 3.1.

Varabile fuzzy	Quantità fuzzy	Membership fuzzy
$x_1 = r(t)$	*positivo*	$\exp[-0.5\ (\min((x_1 - 3)/2,0))^2]$
	negativo	$\exp[-0.5\ (\max((x_1 + 3)/2,0))^2]$
$x_2 = \dot{r}(t)$	*vicino a zero*	$\exp[-0.5\ (x_2)^2]$
$x_3 = \theta(t)$	*positivo*	$\exp[-0.5\ (\min((x_3 - 0.6)/0.4,0))^2]$
	negativo	$\exp[-0.5\ (\max((x_3 + 0.6)/0.4,0))^2]$
$x_4 = \dot{\theta}(t)$	*vicino a zero*	$\exp[-0.5\ (x_4)^2]$
$u = u(t)$	*positivo*	$\exp[-0.5\ (u - 0.5)^2]$
	negativo	$\exp[-0.5\ (u + 0.5)^2]$
	grande positivo	$\exp[-0.5\ (u - 2)^2]$
	grande negativo	$\exp[-0.5\ (u + 2)^2]$

La sintesi della NFN è ottenuta in cinque situazioni differenti, al fine di comprendere meglio le caratteristiche essenziali della procedura di apprendimento. In particolare, si vuole evidenziare come l'informazione linguistica e l'ottimizzazione strutturale della rete siano di particolare aiuto a determinare una NFN dalle prestazioni adeguate anche in presenza di dati numerici rumorosi. Il discorso sull'ottimizzazione strutturale, già introdotta al par. 1.5, sarà ampiamente ripreso nel prossimo capitolo dove, in particolare nel par. 4.2, saranno discussi anche i dettagli delle procedure di clustering utilizzate per questo esempio.

Caso 1: si utilizza solo DS1. La procedura di sintesi utilizza il clustering senza ottimizzazione strutturale del numero di regole; il valore risultante è pari a M=20. La NFN che ne deriva modella in modo corretto la legge di controllo; questo risultato è illustrato in fig. 3.10. In particolare, in fig. 3.10(a) sono mostrate le traiettorie della palla sotto l'azione della legge di controllo ricavata per via analitica e per quattro differenti condizioni iniziali; in fig. 3.10(b) sono mostrate le traiettorie ottenute con la NFN sintetizzata.

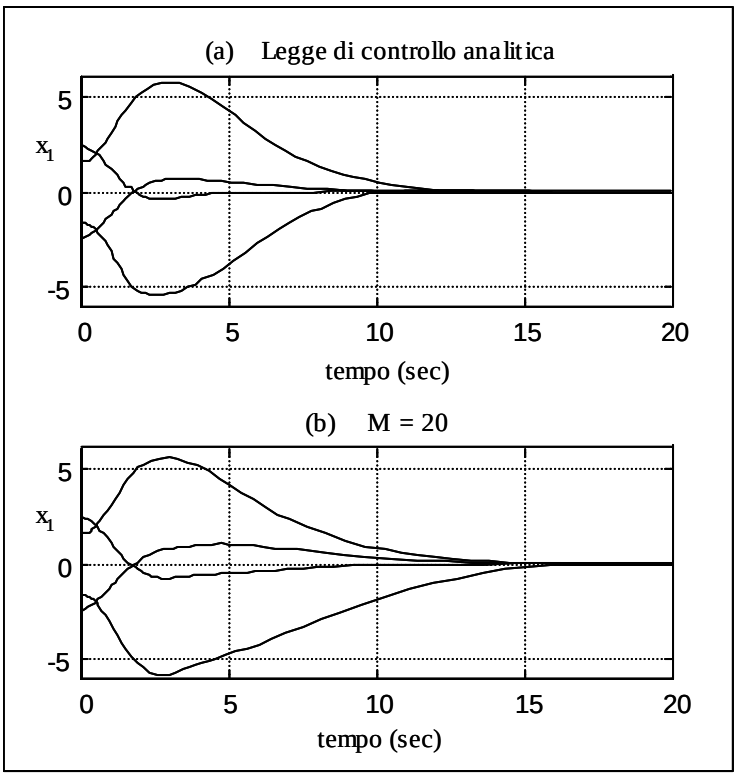


Figura 3.10 - Caso 1: (a) traiettorie analitiche della palla per quattro differenti condizioni iniziali; (b) traiettorie derivanti dalla rete NFN sintetizzata.

Caso 2: simile al caso precedente ma con DS2 al posto di DS1. La NFN risultante non è capace di modellare correttamente la legge di controllo. Le traiettorie delle palla sono in questo caso divergenti, come mostrato in fig. 3.11. Il rumore nei dati è dunque causa di una perdita di informazione tale che quella recuperabile dalla procedura di sintesi non è sufficiente a raggiungere lo scopo prefissato.

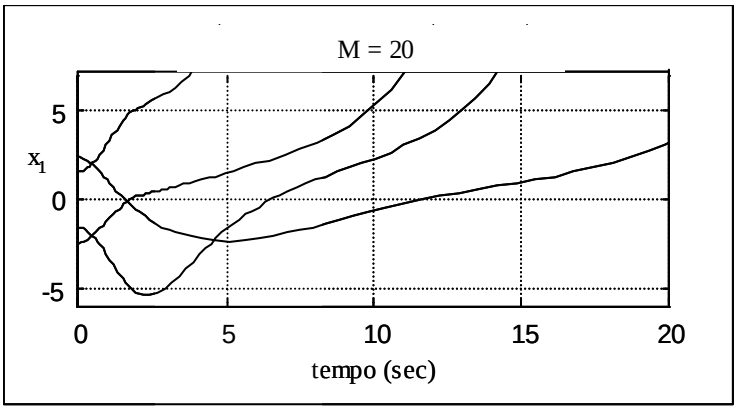


Figura 3.11 - Caso 2: come nel caso 1 ma in presenza di rumore nel data set.

Caso 3: come nel caso 2 ma con l'aggiunta dell'informazione linguistica. La NFN risultante riesce a modellare correttamente il processo di controllo con valori di α nell'intervallo [0.2, 0.5]. In questo campo di valori l'informazione linguistica è più importante di quella numerica e quindi ne compensa il contenuto rumoroso. E' interessante notare che più è alto il valore di α, più è "dolce" la traiettoria della palla. In fig. 3.12(a) sono mostrate le traiettorie per α=0.5, valore per il quale si ottiene la massima dolcezza delle traiettorie.

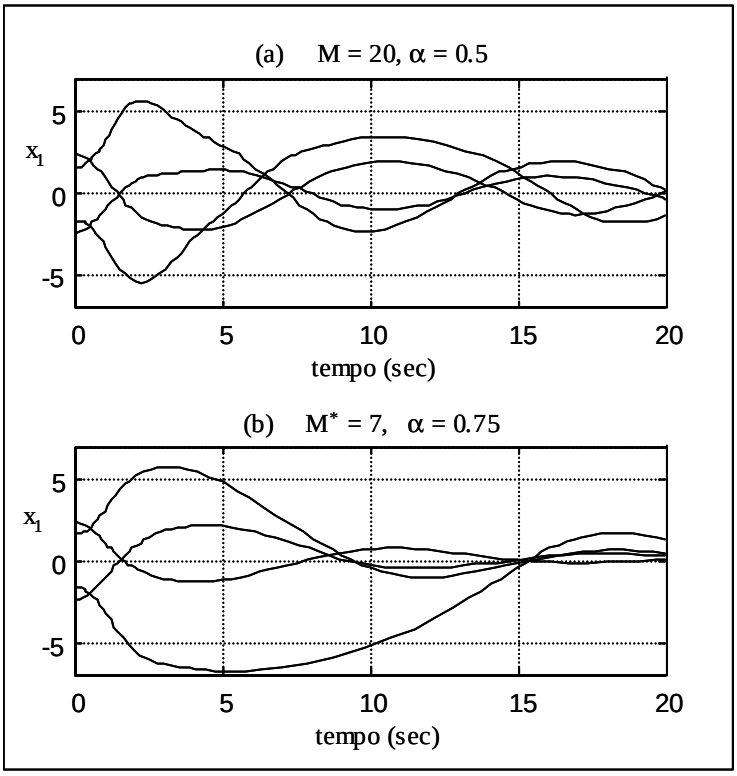


Figura 3.12 – Effetto dell'informazione linguistica: (a) aggiunta al caso 2; (b) aggiunta al caso 4.

Caso 4: come nel caso 2 ma con l'ottimizzazione strutturale (di tipo costruttivo) del numero di regole. Il numero ottimo di regole risulta in questo caso pari a $M^*=7$. Solo una traiettoria della palla è divergente, come mostrato in fig. 3.13. Comparando questi risultati con quelli mostrati in fig. 3.11, si nota un miglioramento dei risultati rispetto al caso 2 in cui non c'era l'ottimizzazione strutturale.

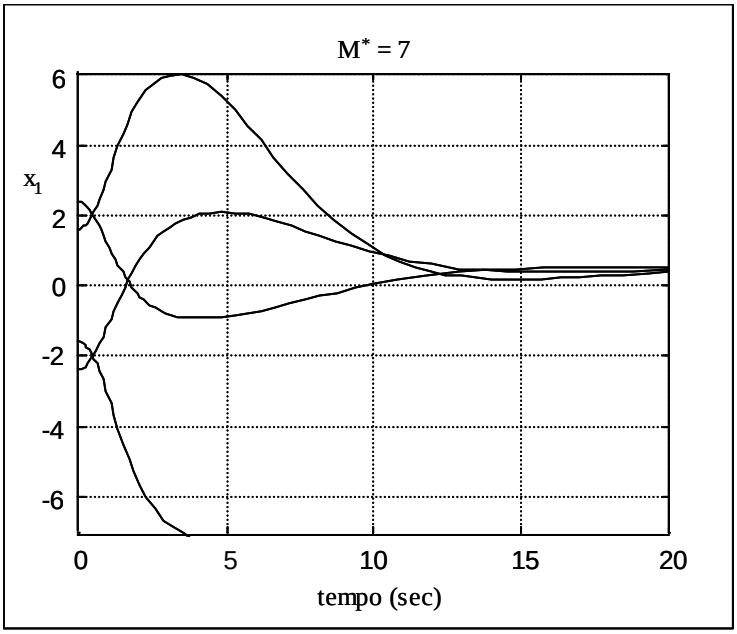


Figura 3.13 - Caso 4: come nel caso 2 ma con l'ottimizzazione strutturale della NFN.

Caso 5: come nel caso 4 ma con l'aggiunta dell'informazione linguistica. Il campo di valori α per cui la legge di controllo è correttamente approssimata è questa volta molto più ampio rispetto al caso 3. Ciò significa che l'ottimizzazione strutturale è comunque di aiuto a filtrare il contenuto rumoroso presente nei dati. In fig. 3.12(b) sono mostrati i risultati per $\alpha=0.75$.

Capitolo
4

UN NUOVO APPROCCIO ALLA SINTESI DELLE RETI ANFIS

4.1 Premessa

In questo capitolo saranno approfonditi i concetti relativi alla sintesi delle reti ANFIS. Sin dalla loro introduzione, le reti ANFIS hanno ricevuto un crescente interesse nella comunità scientifica, grazie all'efficacia mostrata nel risolvere importanti problemi di modellamento come quelli di classificazione, pattern recognition, controllo di sistemi dinamici, e così via. Riassumendo quanto detto nel capitolo precedente, le reti ANFIS fanno uso di un apprendimento data driven supervisionato, in modo da approssimare la relazione ingresso-uscita (in generale non lineare) realizzata da un processo incognito. Le reti ANFIS sono quindi particolarmente adatte a risolvere un problema di approssimazione funzionale, che è poi il problema principale da risolvere in molti dei campi ingegneristici. Infatti, sotto particolari ipotesi [Kosko 1994(bis)], si può dimostrare che esse sono degli approssimatori universali, nel senso definito al par. 3.1.

L'approssimazione della funzione incognita $y=f(\underline{x})$, $f:\Re^n\rightarrow\Re$, è ottenuta nelle reti ANFIS mediante un sistema di inferenza fuzzy costituito da M regole di Sugeno del 1° ordine. La k-esima regola ha la forma (con terminologia inglese):

$$\textit{if } x_1 \text{ is } B_1^{(k)},\ \textit{and } x_2 \text{ is } B_2^{(k)},\ldots,\ \textit{and } x_n \text{ is } B_n^{(k)}\ \textit{then}\ \ y^{(k)} = \sum_{j=1}^{n} a_j^{(k)} x_j + a_0^{(k)} \qquad (4.1.a)$$

dove $\underline{x} = [x_1\, x_2 \ldots x_n]^t$ è il pattern di ingresso (il simbolo 't' denota la trasposizione) ed $y^{(k)}$ è l'uscita crisp associata alla regola. Quest'ultima è caratterizzata dalle funzioni di appartenenza (MF) $\mu_{B_j^{(k)}}(x_j)$ delle variabili di ingresso scalari fuzzy $B_j^{(k)}$, j=1…n, e dai coefficienti $a_j^{(k)}$, j=0…n, dell'uscita crisp. Sono possibili diverse alternative riguardo al tipo di fuzzificazione degli ingressi, alla composizione delle variabili fuzzy e dal modo in cui le M regole sono combinate [Jang 1997]. Con le scelte usuali, già discusse nel cap. 3, una rappresentazione più compatta della regola di Sugeno può diventare la seguente:

$$\textit{if } \underline{x} \text{ is } \underline{B}^{(k)} \textit{ then } \quad y^{(k)} = \sum_{j=1}^{n} a_j^{(k)} x_j + a_0^{(k)} \qquad (4.1.b)$$

dove $\underline{B}^{(k)}$ è la variabile vettoriale fuzzy complessiva e $\mu_{\underline{B}^{(k)}}(\underline{x})$ la relativa MF. Le scelte usuali sui vari operatori fuzzy conducono quindi al seguente sistema di inferenza:

$$\tilde{y} = \sum_{k=1}^{M} \frac{\mu_{\underline{B}}{}^{(k)}(\underline{x}) \cdot y^{(k)}}{\sum_{q=1}^{M} \mu_{\underline{B}}{}^{(q)}(\underline{x})} \qquad (4.2)$$

dove $\tilde{y}$ è l'uscita stimata che approssima $y=f(\underline{x})$.

Come detto nel cap. 3, le informazioni relative al processo da stimare sono spesso fornite per mezzo di un campionamento ingresso-uscita del processo stesso. Si ha a disposizione un training set di esempi costituito da P coppie $(\underline{x}_i, y_i)$, i=1…P. In questa situazione, la sintesi delle reti ANFIS è tipicamente basata sull'utilizzo di un sistema di clustering che operi sul training set. Nel par. 3.4 sono già stati ampiamente discussi i possibili tipi di approcci alla sintesi basata sul clustering; a tale proposito, in questo capitolo verrà presentata una nuova procedura di sintesi delle reti ANFIS. La procedura è basata sull'utilizzo del clustering nello spazio congiunto ingesso-uscita.

Lo scopo di questa nuova procedura di sintesi è quello di ottenere una migliore capacità di generalizzazione della rete risultante, ossia una migliore corrispondenza tra il modello sintetizzato e l'effettiva distribuzione dei dati. Ciò può essere ottenuto solo se l'architettura è costituita da un opportuno numero di regole, vale a dire che è

necessaria un'ottimizzazione strutturale della rete stessa. La determinazione del numero ottimo di regole può risultare molto critica, soprattutto perché la rete neurale potrebbe andare in *overfitting* a causa di dati rumorosi o mal condizionati. Sarà quindi proposta una tecnica di ottimizzazione strutturale di tipo costruttivo che permetterà la determinazione automatica del numero ottimo di regole.

4.2 Metodi tradizionali di sintesi basata sul clustering

In letteratura sono state presentate numerose alternative riguardo ai metodi basati sul clustering per la sintesi di reti ANFIS. Ciascuna di esse è caratterizzata da un differente livello di complessità, a seconda dell'approccio seguito per l'ottimizzazione dei risultati. Le scelte da fare sono relative al tipo di MF, ai processi di fuzzificazione e defuzzificazione, all'algoritmo di clustering, allo spazio in cui esso è applicato, e così via. L'utilizzo di simulatori software che permettano la valutazione comparativa di tali procedure diventa quindi indispensabile, soprattutto per validare i risultati della propria attività di ricerca. In fig. 4.1 è mostrata la schermata principale di uno degli strumenti software implementati a tale scopo durante il corso di dottorato.

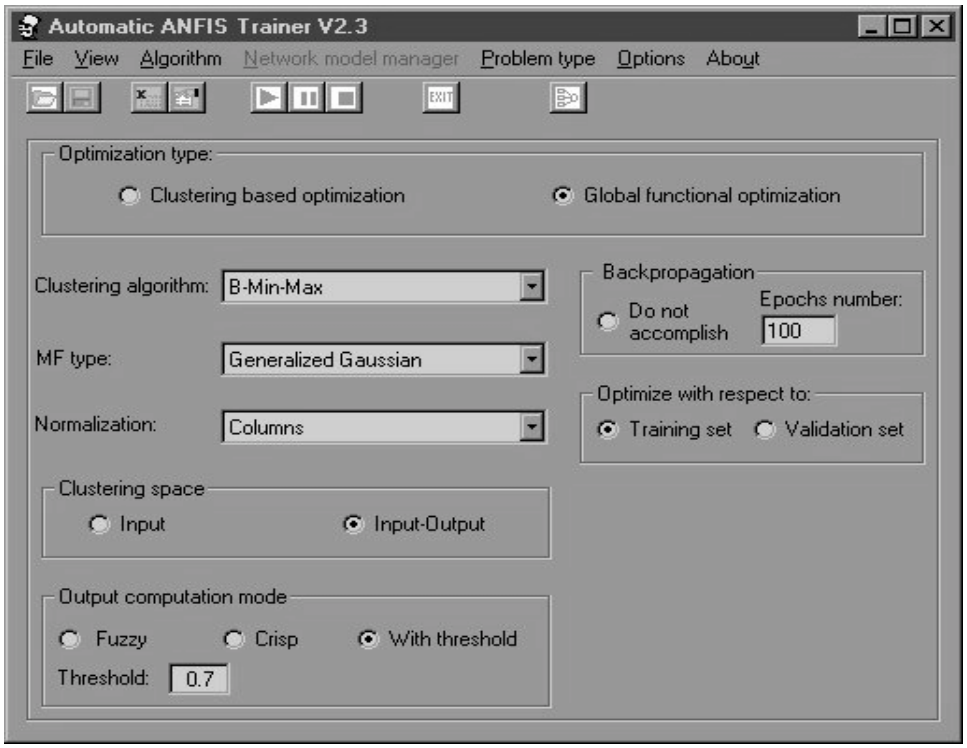


Figura 4.1 – Uno degli strumenti software realizzati per la sintesi di reti ANFIS.

Nel seguito di questo paragrafo verranno presentati due dei metodi di sintesi più classici e rappresentativi, entrambi legati ai principi generali della teoria dell'apprendimento nelle reti neurali [Haykin 1999].

4.2.1 Ottimizzazione dei soli risultati del clustering

Da quanto detto al cap. 1, dovrebbe essere chiaro che l'obiettivo finale dell'apprendimento è sempre la massimizzazione della capacità di generalizzazione di una sistema di modellamento (la rete ANFIS, in questo caso). Tuttavia, uno degli approcci classici in letteratura è basato sull'ottimizzazione dei soli risultati del clustering. Il criterio di ottimizzazione seguito in questo caso può essere legato, ad esempio, alla compattezza ed alla separabilità dei cluster stessi. Successivamente, ad ogni cluster è associata una regola secondo quanto illustrato al par. 3.4.1. Lo schema di sintesi è quindi il seguente:

- clustering del data set nello spazio di ingresso $\underline{X}$ ed ottimizzazione del numero di cluster sulla base di un opportuno criterio (ad esempio, massimizzazione della compattezza-separabilità tra cluster). Siano $\Gamma_{\underline{X}}^{(k)}$, k=1…M, i cluster così ottenuti, ciascuno dei quali è associato ad una regola distinta;
- calcolo delle MF degli antecedenti $\underline{B}^{(k)}(\underline{X})$ di ciascuna regola, sulla base della densità dei pattern contenuti in ogni cluster;
- calcolo dei coefficienti $a_j^{(k)}$, j=0…n, dei conseguenti di ciascuna regola, ottenuti risolvendo un problema ai minimi quadrati. In particolare, è necessario risolvere il seguente sistema di equazioni lineari:

$$y_t = \sum_{j=1}^{n} a_j^{(k)} x_{tj} + a_0^{(k)} \qquad (4.3)$$

  in cui le equazioni considerate sono quelle relative alle coppie $(\underline{x}_t, y_t)$ per le quali $\underline{x}_t$ appartiene a $\Gamma_{\underline{X}}^{(k)}$;
- implementazione della rete ANFIS utilizzando come componenti le regole precedentemente ottenute.

4.2.2 Ottimizzazione della capacità di generalizzazione

Questa seconda alternativa è basata sulla diretta ottimizzazione della capacità di generalizzazione. Essa si basa sulla minimizzazione di un funzionale di costo che tenga conto sia dell'adeguatezza del modello rispetto ai campioni presenti nel training set, sia della complessità del modello risultante. Tale funzionale deve pertanto risultare direttamente legato alla capacità di generalizzazione della rete neurale, anche se è quasi sempre difficile da esprimere correttamente in forma analitica. Spesso si usa una procedura alternativa detta di *cross-validation*, in cui il valore del funzionale viene sostituito dalla valutazione della performance rispetto ai campioni presenti in un insieme, chiamato *validation set*, che contiene dati non utilizzati durante l'apprendimento. Come accennato al par. 1.5, l'ottimizzazione strutturale della rete ANFIS può essere ottenuta con un approccio o costruttivo o di pruning, entrambi comunque basati sullo schema di fig. 4.2 che è riassunto qui di seguito:

- divisione del data set (DS) in due sottoinsiemi: il training set (TS) ed il validation set (VS);
- calcolo del numero ottimo di cluster, e perciò della struttura ottima della rete ANFIS, usando il seguente algoritmo di ricerca unimodale:
  - incremento[1], ad ogni passo della procedura, di un parametro φ che controlla l'algoritmo di clustering applicato al TS nello spazio di ingresso;
  - ogni cluster $\Gamma_{\underline{X}}^{(k)}$ nello spazio di ingresso induce un sistema di equazioni lineari (4.3) che deve essere risolto, ai minimi quadrati, per ottenere i coefficienti $a_j^{(k)}$, j=0…n, della relativa regola;
  - la capacità di generalizzazione della rete ANFIS, ottenuta in corrispondenza del valore corrente di φ, è misurata dalla performance della rete sul VS;

[1] La scelta di incrementare (o di diminuire) il parametro è puramente convenzionale. Infatti, l'approccio è costruttivo se ad ogni incremento (diminuzione) del parametro corrisponde sempre un incremento delle regole. Discorso opposto vale ovviamente per gli algoritmi di pruning.

- Il valore ottimo di φ, ottenuto in corrispondenza della migliore performance sul VS, determina l'architettura ottima (ossia il numero di regole) della rete ANFIS. Tutti i parametri presenti nella rete possono successivamente essere raffinati usando l'intero DS. Una procedura particolarmente adatta a tale scopo consiste nell'uso alternato di due procedure di raffinamento: la prima riguardante i parametri delle MF di ingresso usando, ad esempio, tecniche basate sulla *discesa a gradiente*; la seconda riguardante i coefficienti dei conseguenti usando, ad esempio, tecniche basate sui minimi quadrati. Alcuni tra i metodi più noti sono stati presentati, valutati e comparati in [Guély 1993, Jang 1997, Chen 1999, Guillame 2001].

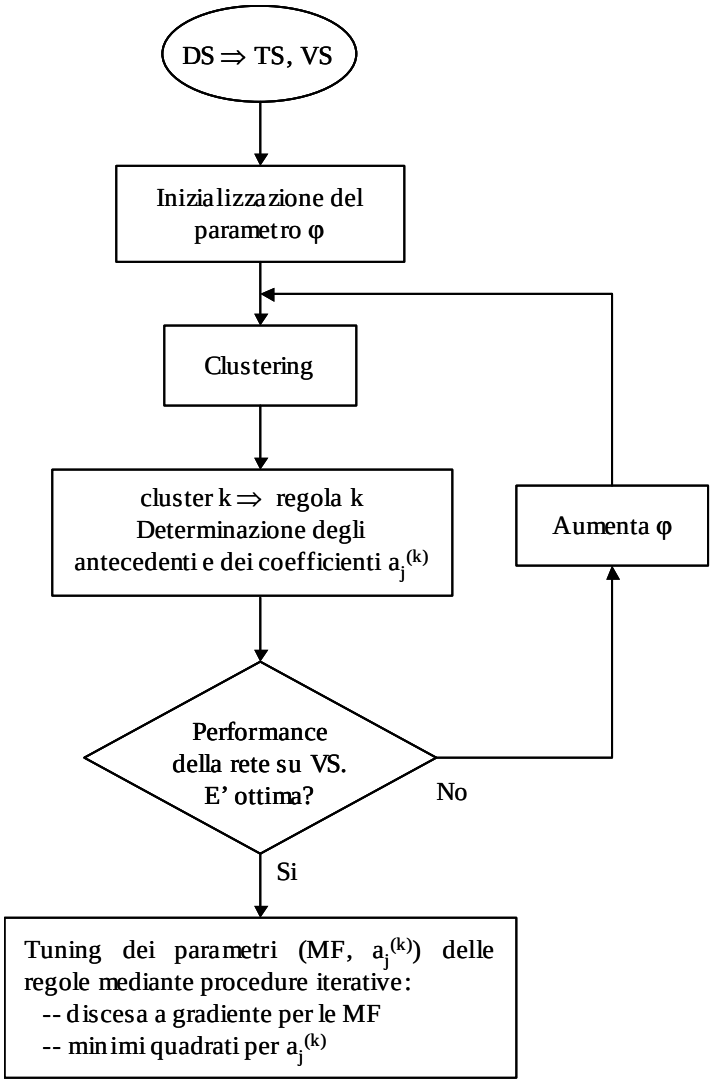


Figura 4.2 – Una possibile procedura di ottimizzazione strutturale per la sintesi di reti ANFIS.

4.3 Sintesi di reti ANFIS mediante "clustering per iperpiani"

La procedura di clustering che si vuole proporre in questo paragrafo è orientata alla determinazione diretta della struttura della funzione incognita f($\underline{x}$). A tale proposito, è importante ricordare che una rete ANFIS può essere considerata un modello di regressione *localmente lineare*, ossia lineare in diverse regioni dello spazio di ingresso. La funzione f($\underline{x}$) è quindi approssimata attraverso M iperpiani, ciascuno relativo ad una regola.

Come sarà più evidente nel seguito, la procedura di clustering è definita nello spazio congiunto ingresso-uscita. Il prototipo del k-esimo cluster, associato alla k-esima regola, sarà quindi definito nello spazio congiunto e sarà rappresentato proprio dai coefficienti $a_j^{(k)}$, j=0…n, che determinano l'uscita lineare (l'iperpiano) relativa al conseguente della k-esima regola, k=1…M. Al fine di determinare questi prototipi, è di seguito proposta la tecnica di ottimizzazione alternata che sarà definita come clustering nello *spazio degli iperpiani* [Panella 2001(ter)]:

- ✓ **Inizializzazione.** Fissato un valore di M, gli iperpiani sono inizializzati seguendo un criterio opportuno. In questo caso, si è scelta un'inizializzazione basata sul noto algoritmo di clustering C-Means[2] [Forgy 1965, Selim 1984, Jain 1988] applicato al solo spazio di ingresso. In particolare, siano $\Gamma_{\underline{X}}^{(k)}$, k=1…M, i cluster determinati dal C-Means nello spazio di ingresso; la corrispondenza tra ciascuno dei cluster e l'iperpiano della regola corrispondente è basata sul seguente criterio induttivo: "se un pattern di ingresso $\underline{x}_i$, i=1…P, appartiene al cluster $\Gamma_{\underline{X}}^{(q)}$, 1≤q≤M, allora la coppia ingresso-uscita ($\underline{x}_i$, y_i) appartiene all'iperpiano corrispondente A_q".

- ✓ **Passo 1.** I coefficienti di ogni iperpiano sono aggiornati utilizzando le coppie ad esso assegnate (o nell'inizializzazione o nel successivo passo 2). Per il k-

[2] Per ragioni storiche, spesso l'algoritmo C-Means è anche noto come algoritmo K-Means.

esimo iperpiano, k=1…M, deve essere risolto un sistema di equazioni lineari del tipo (4.3), dove l'indice 't' è applicato alle sole coppie assegnate al k-esimo iperpiano. Una qualsiasi tecnica ai minimi quadrati può essere usata per risolvere questo sistema di equazioni.

- ✓ **Passo 2.** Ogni coppia ($\underline{x}_i$, y_i) del training set è assegnata all'iperpiano aggiornato A_q, con q tale che:

$$d_i = \left| y_i - \left(\sum_{j=1}^{n} a_j^{(q)} x_{ij} + a_0^{(q)} \right) \right| = \min_{k=1...M} \left| y_i - \left(\sum_{j=1}^{n} a_j^{(k)} x_{ij} + a_0^{(k)} \right) \right| \qquad (4.4)$$

- ✓ **Criterio di stop.** Come criterio di convergenza dell'algoritmo si è scelta la quantità $\Theta = \left| D - D^{(old)} \right|$, dove D è l'errore di approssimazione corrente definito da:

$$D = \sum_{i=1}^{P} d_i \qquad (4.5)$$

e $D^{(old)}$ è l'errore di approssimazione calcolato nell'iterazione precedente. L'algoritmo iterativo si ferma se Θ è minore di una soglia prestabilita θ, altrimenti ritorna al passo 1. Nel seguito si utilizzerà un valore di riferimento θ=0.001.

Il precedente algoritmo permette la determinazione dei soli conseguenti lineari delle regole di Sugeno. Tuttavia, come accennato al par. 3.4.3, nello spazio di ingresso potrebbero esistere insiemi ben distinti di pattern associabili ad uno stesso iperpiano. Pertanto, il calcolo delle MF degli antecedenti non è così banale, dato che i cluster indotti da uno stesso iperpiano nello spazio di ingresso potrebbero essere ben più di uno. Una situazione tipica è quella illustrata in fig. 4.3, dove lo stesso iperpiano riesce ad approssimare in modo adeguato la funzione in due distinte regioni dello spazio di ingresso. Ciò è ulteriormente evidenziato in fig. 4.4, dove è mostrata la proiezione di questa funzione su uno dei piani coordinati.

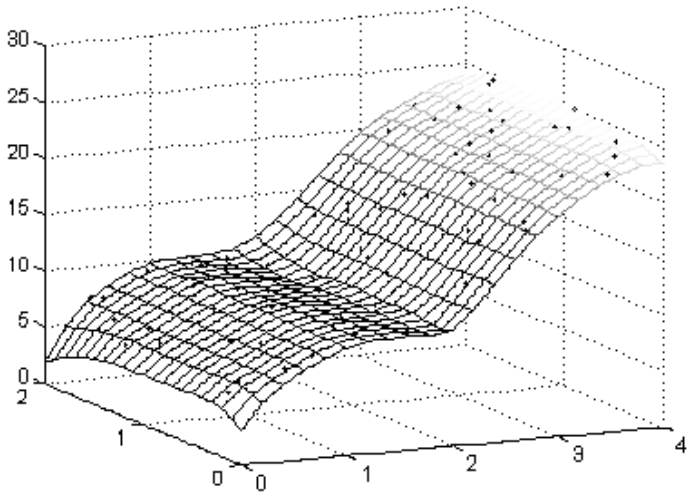


Figura 4.3 – Funzione approssimabile con lo stesso iperpiano in due regioni distinte dello spazio di ingresso.

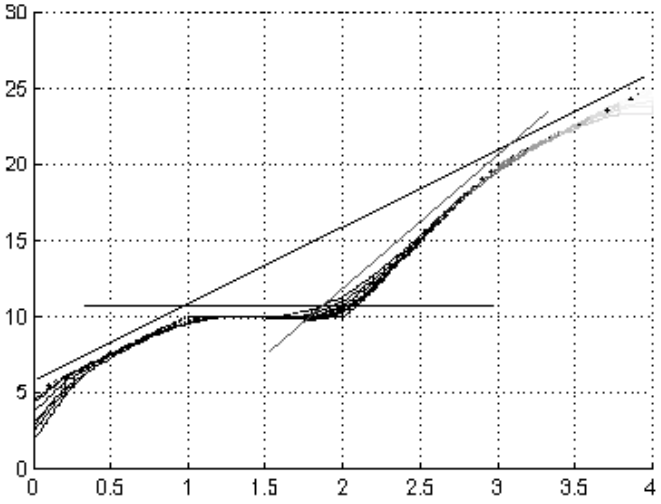
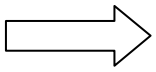


Figura 4.4 - Proiezione della funzione in fig. 4.3.

La soluzione a tutto ciò può essere ottenuta impostando un opportuno problema di classificazione nello spazio di ingresso: ogni pattern $\underline{x}_i$, i=1...P, può essere etichettato con un intero q (etichetta di classe), $1 \le q \le M$, rappresentante l'iperpiano al quale il pattern è stato assegnato nell'ultima iterazione (specificatamente al passo 2) della precedente procedura di clustering. Una volta impostato il problema, la sua soluzione può essere ottenuta utilizzando un qualsiasi sistema di classificazione fuzzy. Nel seguito si illustrerà l'uso dei classificatori a risoluzione adattativa (Adaptive

Resolution Classifier – ARC), appartenenti alla ben nota famiglia dei classificatori Min-Max di Simpson.



Nota: i classificatori ARC sono stati oggetto di una ricerca che ha richiesto un impegno concorrente e sinergico a quello richiesto per ottenere i risultati presentati in questa tesi. Tuttavia, per ragioni di sintesi, non si entrerà nel dettaglio dell'attività svolta in tale ambito, per la quale si rimanda ai riferimenti bibliografici che saranno citati nel seguito della discussione.

La combinazione del clustering per iperpiani e della classificazione nello spazio di ingresso permette di sintetizzare una rete ANFIS con un fissato numero M di regole. A tale algoritmo ci si riferirà nel seguito col nome di *sintesi mediante clustering per iperpiani* (Hyperplane Clustering Synthesis - HCS). Prima di introdurre la procedura di ottimizzazione strutturale dell'algoritmo HCS, nel prossimo paragrafo verranno brevemente illustrati i concetti essenziali relativi ai classificatori ARC.

4.4 Uso dei classificatori ARC per il calcolo degli antecedenti

La tecnica di classificazione Min-Max consiste nella copertura di un insieme di pattern del training set attraverso degli iperparallelepipedi (Hyperbox – HB) con le facce parallele agli assi coordinati del sistema di riferimento dei dati. Conseguentemente, un hyperbox H_r è completamente definito da due punti nello spazio dei dati: i suoi vertici di minimo ("min") e di massimo ("max"), $\underline{v}_r$ and $\underline{w}_r$, rispettivamente. Un hyperbox può quindi essere considerato come una struttura crisp su cui è possibile adattare delle opportune MF. Nel seguito si utilizzerà la MF originale proposta da Simpson in [Simpson 1992], in cui la pendenza al di fuori dell'hyperbox è associata ad un parametro γ:

$$\mu_{H_r}(\underline{x}) = \frac{1}{n}\sum_{j=1}^{n}[1 - f(x_j - w_{rj}) - f(v_{rj} - x_j)] \; ; \qquad f(z) = \begin{cases} 1 & , \; \gamma \cdot z > 1 \\ \gamma \cdot z & , \; 0 \le \gamma \cdot z \le 1 \\ 0 & , \; \gamma \cdot z < 0 \end{cases} \qquad (4.6)$$

dove $\underline{v}_r = [v_{r1}\, v_{r2} \ldots v_{rn}]^t$ e $\underline{w}_r = [w_{r1}\, w_{r2} \ldots w_{rn}]^t$. Nel seguito si utilizzerà il valore di riferimento γ=0.5.

Ogni HB di un classificatore Min-Max è associato ad una classe fra quelle che etichettano i pattern del training set. Ovviamente, ad una stessa classe possono essere associati svariati HB. Nel nostro caso, ciascuna classe corrisponde proprio ad uno dei cluster definiti nello spazio congiunto ingresso-uscita (un iperpiano) e perciò è associata ad una regola della rete ANFIS. In altre parole, ogni HB rappresenterà uno dei diversi cluster che possono essere indotti nello spazio di ingresso da un iperpiano. La MF complessiva $\mu_{\underline{B}^{(k)}}(\underline{x})$ della k-esima regola, k=1…M, può essere determinata sulla base degli operatori di composizione usualmente adottati nelle reti neurofuzzy di tipo Min-Max [Jang 1997, Simpson 1992]. Ad esempio, se $H_1^{(q)}$, $H_2^{(q)}$,…, $H_R^{(q)}$ sono gli HB associati alla classe (regola) q, e $\mu_{H_1^{(q)}}(\underline{x})$, $\mu_{H_2^{(q)}}(\underline{x})$,…, $\mu_{H_R^{(q)}}(\underline{x})$ sono le corrispondenti MF, allora si avrà:

$$\mu_{\underline{B}^{(q)}}(\underline{x}) = \max\left\{\mu_{H_1^{(q)}}(\underline{x}),\ \mu_{H_2^{(q)}}(\underline{x}), \ldots,\ \mu_{H_R^{(q)}}(\underline{x})\right\} \qquad (4.7)$$

Sia l'algoritmo originale di Simpson che quasi tutte le sue versioni successivamente sviluppate sono comunque caratterizzate dai seguenti difetti:

- eccessiva dipendenza dall'ordine di presentazione dei pattern del training set;
- risoluzione costante, nell'intero spazio di ingresso, della copertura dei pattern effettuata dagli HB;

Al fine di superare questi inconvenienti, in [Rizzi 1998] sono stati presentati i classificatori ARC, in cui l'operazione di base per ottenere una risoluzione variabile è il *taglio* dell'hyperbox. Il taglio consiste nella divisione dell'HB in due parti attraverso un iperpiano perpendicolare agli assi coordinati e passante per un punto opportuno. Lo scopo del taglio è quello di generare HB che coprano un insieme di pattern associati ad un'unica classe. Questi HB saranno nel seguito chiamati *puri*, in contrasto con gli altri chiamati *ibridi*. In accordo con i principi della teoria dell'apprendimento, gli HB puri associati ad una stessa classe sono fusi se l'HB che ne consegue non genera sovrapposizioni né con HB ibridi né con HB puri di un'altra classe. Inoltre, gli HB

derivati da un'operazione di taglio sono ridotti alla minima misura che contenga comunque tutti i suoi pattern. Una descrizione più dettagliata dell'algoritmo di apprendimento ARC è presente in [Rizzi 2002]; esistono anche delle varianti più raffinate, peraltro già presentate in [Rizzi 2000, Rizzi 2001]. Quest'ultime mirano a migliorare la velocità della procedura di apprendimento e la corrispondenza tra gli HB e le effettive strutture presenti nei dati.

La sintesi dei classificatori ARC è ottimizzata strutturalmente; durante l'apprendimento l'algoritmo genera una successione di classificatori Min-Max caratterizzati da una crescente complessità (numero di neuroni nello strato nascosto) e da un'accuratezza sempre migliore nel classificare i pattern del training set. Secondo quanto detto al par. 1.5, il classificatore ottimo è ottenuto in corrispondenza del minimo di un funzionale di costo definito da:

$$F_{ARC} = (1-\lambda_{ARC})E_{ARC} + \lambda_{ARC}C_{ARC} \qquad (4.8)$$

dove E_{ARC} è la percentuale di esempi del training set per i quali si verifica un *errore di classificazione*; C_{ARC} è la percentuale di HB rispetto alla cardinalità P del training set; λ_{ARC} è un peso con $0 \le \lambda_{ARC} \le 1$. Il valore di questo peso non è critico, dato che i risultati sono lievemente influenzati da una sua variazione in un largo intervallo intorno a $\lambda_{ARC}=0.5$. Il flow-chart dell'algoritmo di apprendimento ARC è mostrato in fig. 4.5.

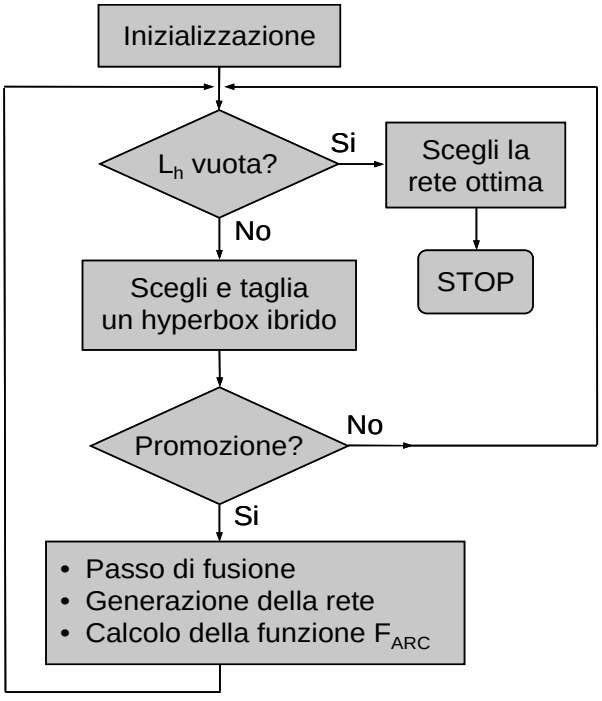


Figura 4.5 - Flow-chart dell'algoritmo di apprendimento ARC.

Come dimostrato in [Rizzi 2002], le prestazioni dell'algoritmo ARC superano, in termini di capacità di generalizzazione e di costo computazionale, sia quelle dell'algoritmo originale di Simpson che quelle di alcune sue versioni ottimizzate [Frattale Mascioli 1998, Frattale Mascioli 2000]. Inoltre, i classificatori Min-Max allenati dall'algoritmo ARC sono indipendenti dall'ordine di presentazione. Per tutti questi motivi, i classificatori ARC saranno utilizzati nelle procedure di sintesi HCS per il calcolo delle MF di ingresso delle reti ANFIS.

4.5 Ottimizzazione costruttiva dell'algoritmo HCS

Come già accennato nel cap. 3, la sintesi di una rete neurofuzzy consiste in realtà in un duplice processo: da una parte è necessario ottimizzare un certo insieme di parametri (MF e coefficienti di uscita); dall'altra è necessario stimare in modo accurato il numero di regole che garantiscano le migliori prestazioni in termini di capacità di generalizzazione. Ciò significa effettuare un'ottimizzazione strutturale della rete che può seguire principi simili a quelli indicati nel paragrafo precedente. In particolare, come misura della performance sul training set si utilizzerà l'*errore quadratico medio* (MSE):

$$E = \frac{1}{P}\sum_{i=1}^{P}\left(y_i - \tilde{y}_i\right)^2 \qquad (4.9)$$

dove $\tilde{y}_i$ è l'uscita della rete ANFIS in corrispondenza dell'i-esimo pattern $\underline{x}_i$, i=1...P, determinata con la (4.2). Dalle molte analisi preliminari svolte a tale riguardo, si è giunti a conclusione che è conveniente utilizzare un funzionale di costo come quello proposto in (4.8):

$$F(M) = (1-\lambda)\frac{E(M) - E_{min}}{E_{max} - E_{min}} + \lambda\frac{M}{P} \qquad (4.10)$$

dove E_{min} ed E_{max} sono, rispettivamente, il valore minimo e massimo della performance, ottenuti durante la procedura di ottimizzazione in corrispondenza di diversi valori di M; λ è un peso per il quale valgono le stesse considerazioni fatte per λ_{ARC} a proposito della (4.8).

La procedura di ottimizzazione strutturale proposta per la minimizzazione della (4.10) è di tipo costruttivo; essa sarà denotata nel seguito con OHCS (HCS Ottimizzata – Optimized HCS). Il valore di M è incrementato progressivamente da 1 ad M_{max}, dove M_{max} è una frazione della cardinalità del training set che rappresenta la massima complessità consentita alla rete. In corrispondenza ad ogni valore di M sono generate diverse reti ANFIS utilizzando la procedura HCS. Ciò è reso necessario dal fatto che la procedura HCS fa affidamento su un'inizializzazione casuale degli iperpiani, in quanto tale è l'inizializzazione di solito utilizzata per l'algoritmo C-Means. Ne consegue che differenti inizializzazioni possono condurre a reti ANFIS anche molto diverse tra loro, pur conservando lo stesso numero di regole. La ragione di questa differenza tra i risultati risiede nel fatto che l'algoritmo HCS può restare facilmente intrappolato nei minimi locali della (4.5), come del resto accade per ogni altro algoritmo di minimizzazione alternata.

Ricapitolando, nella procedura OHCS sono necessarie differenti inizializzazioni per uno stesso valore di M, al fine di ottenere una soddisfacente capacità di generalizzazione della rete ANFIS. Se T sono le inizializzazioni effettuate per ogni valore di M, allora la procedura OHCS genererà $T \cdot M_{max}$ reti, tra le quali l'ottima sarà scelta sulla base del minimo valore della (4.10). Il flow-chart dell'algoritmo OHCS è riportato in fig. 4.6.

Sfruttando quanto già proposto in [Panella 2000, Panella 2001], esiste anche la possibilità di sviluppare ulteriormente l'ottimizzazione costruttiva OHCS, cercando di diminuire il suo costo computazionale e di migliorare l'accuratezza della soluzione finale. L'idea è quella di evitare la necessità di effettuare diverse inizializzazioni per ogni valore di M, cioè di eliminare ogni aleatorietà nell'inizializzazione dell'algoritmo HCS. Ciò potrebbe essere fatto utilizzando una procedura costruttiva gerarchica dove, per ogni valore di M, alla fine dell'algoritmo HCS è opportunamente selezionata una regola da eliminare (sulla base, ad esempio, dell'errore locale da essa prodotto). Questa regola potrebbe essere usata per generare delle *regole figlie* ossia per inizializzare, in modo opportuno e deterministico, la successiva esecuzione dell'algoritmo HCS in cui il numero di regole è aumentato.

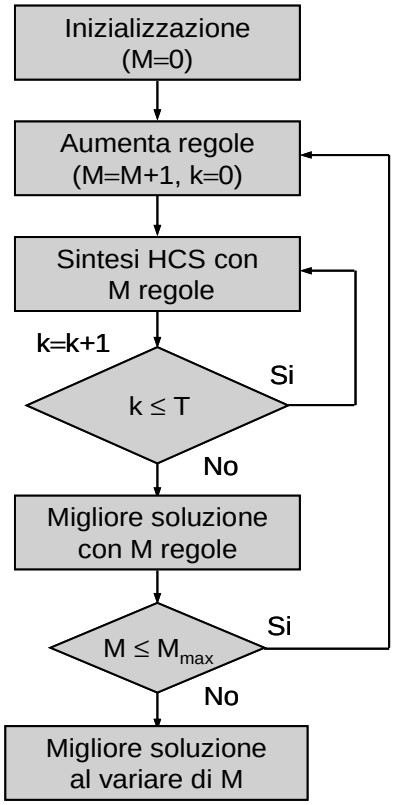


Figura 4.6 - Flow-chart della procedura di ottimizzazione OHCS.

Analogamente a quanto detto nel par. 4.2.2, oltre alla procedura OHCS fin qui descritta sono possibili anche diversi metodi (discesa a gradiente, minimi quadrati, ecc.) per raffinare la stima dei parametri della rete ANFIS. Tali metodi mirano ad ottenere una migliore corrispondenza tra gli iperpiani e l'effettiva distribuzione dei dati, così come una maggiore accuratezza nell'approssimazione della funzione incognita. In conclusione di paragrafo si vuole proporre una semplice procedura di questo tipo, la cui efficacia è stata sperimentalmente verificata.

Come illustrato nel par. 4.3, la sintesi HCS consiste nel clustering per iperpiani e nella classificazione fuzzy dello spazio di ingresso. Il primo conduce al partizionamento dei dati ingresso-uscita ed al calcolo dei coefficienti $a_j^{(k)}$, j=0…n, k=1...M, dei conseguenti. La successiva classificazione modella lo spazio di ingresso con degli hyperbox, i quali determinano con le loro MF gli antecedenti delle regole fuzzy.

Invece di fermarsi ai coefficienti $a_j^{(k)}$ così calcolati, è possibile aggiornarne la stima (e quindi aggiornare gli iperpiani) al fine di ottenere un'approssimazione più accurata della funzione. Una semplice procedura è quella che aggiorna i coefficienti $a_j^{(k)}$ in modo fuzzy; ciò è possibile andando a risolvere un sistema di equazioni lineari che sia anche basato sulle MF (4.7), le quali legano ciascun pattern ad una regola (iperpiano). Questo sistema di equazioni lineari può essere ottenuto minimizzando l'errore quadratico medio (4.9), cioè ponendo a zero tutte le sue derivate rispetto ad $a_j^{(k)}$, j=0…n, k=1...M. Dalle (4.2) e (4.9) si ottiene il seguente sistema di equazioni lineari, in cui le incognite sono i coefficienti $a_j^{(k)}$ da aggiornare:

$$< y_i \xi_{ir}^{(q)} >_i = \sum_{k=1}^{M} \sum_{j=0}^{n} a_j^{(k)} < \xi_{ij}^{(k)} \xi_{ir}^{(q)} >_i \,, \quad r=0...n, \quad q=1...M \qquad (4.11)$$

dove $< \cdot >_i$ denota la media sui P pattern del training set ed inoltre

$$\xi_{ij}^{(k)} = \begin{cases} \dfrac{\mu_{\underline{B}}^{(k)}(\underline{x}_i) \cdot x_{ij}}{\sum_{q=1}^{M} \mu_{\underline{B}}^{(q)}(\underline{x}_i)}, \; 1 \le j \le n \\ \\ \dfrac{\mu_{\underline{B}}^{(k)}(\underline{x}_i)}{\sum_{q=1}^{M} \mu_{\underline{B}}^{(q)}(\underline{x}_i)}, \; j = 0 \end{cases}, \quad k=1...M, \quad i=1...P \qquad (4.12)$$

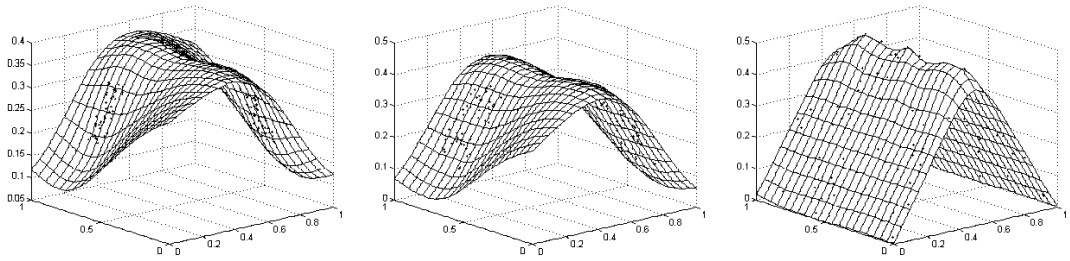
4.6 Risultati sperimentali

La validità della procedura di sintesi proposta è confermata dall'ingente quantità di prove svolte a tale proposito; nel seguito ne verranno illustrati i risultati più significativi. Innanzitutto, nel seguente esperimento sarà analizzata la capacità dell'algoritmo HCS di determinare la reale struttura del processo $y=f(\underline{x})$.

Consideriamo una funzione bidimensionale, $f:\Re^2 \to \Re$, costituita da due piani:

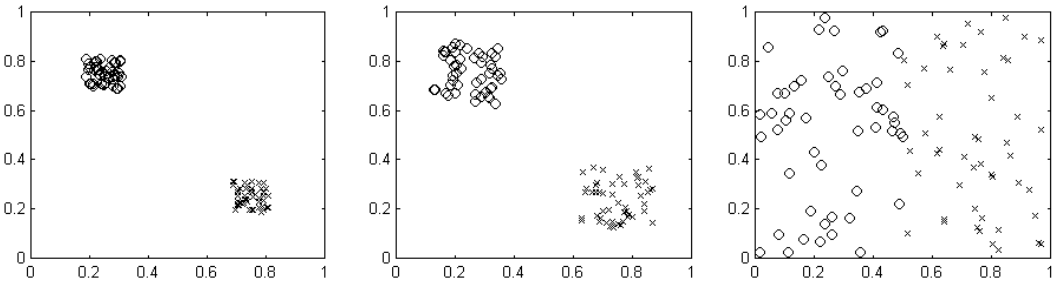
$$y = 0.5 - |x_1 - 0.5|, \quad \underline{x} = (x_1, x_2), \quad x_1, x_2 \in [0,1] \qquad (4.13)$$

La procedura di apprendimento è eseguita utilizzando tre diversi training set, ottenuti dal campionamento ingresso-uscita della (4.13). Ogni training set è costituito da 50 pattern distribuiti nell'intorno di un piano e da 50 pattern distribuiti nell'intorno dell'altro piano. Le procedure di campionamento utilizzate sono tali che i pattern tendono ad avere una distribuzione nello spazio di ingresso sempre più diffusa ed uniforme. I pattern appartenenti ad ogni training set sono mostrati nelle figg. 4.7(a-c), rispettivamente, insieme alla superficie che rappresenta la funzione realizzata dalla risultante rete ANFIS[3]. Evidentemente, ciascuna di queste funzioni approssima bene la (4.13); ciò è ulteriormente confermato dall'osservazione dei training set nello spazio di ingresso, rispettivamente mostrati nelle figg. 4.8(a-c). Ogni pattern è simboleggiato da 'x' oppure da 'o', a seconda del piano a cui è stato assegnato dal classificatore ARC. Ebbene, i pattern sono sempre assegnati al piano corretto, perfino nella situazione di fig. 4.8(c) nella quale essi risultano distribuiti in modo quasi uniforme nello spazio di ingresso. E' facile notare come, in questo ultimo caso, una procedura di clustering che operi solo nello spazio di ingresso non giungerebbe a tale risultato.



(a) (b) (c)

Figura 4.7 – Funzioni di uscita prodotte dalle reti ANFIS per approssimare la (4.13).



(a) (b) (c)

Figura 4.8 - Proiezione nello spazio di ingresso dei training set di fig. 4.7.

[3] Ogni rete ANFIS è ottenuta applicando al rispettivo training set la procedura HCS con M=2.

L'efficacia della procedura OHCS è valutata nel seguito comparando le sue prestazioni con quelle dei più noti paradigmi neurali. In particolare, sarà svolta un'analisi comparativa rispetto al sistema di modellamento neurofuzzy denominato "Self-Organized Fuzzy Rule Generation" (SOFRG) [Rojas 2000] ed alle più note Radial Basis Function (RBF) e Multi-Layer Perceptron (MLP). Si riportano in tab. 4.1 i risultati, in termini di MSE, relativi ai problemi di approssimazione funzionale esaminati in [Rojas 2000]:

1) $f_1(\underline{x}) = \dfrac{C_a}{1+e^{-C_b x_2}} + \Theta(x_1)$;

2) $f_2(\underline{x}) = C_c \sin^2\left(2\pi\sqrt{\dfrac{(5-x_1)^2+(5-x_2)^2}{10}}\right)$;

3) $f_3(\underline{x}) = C_d \dfrac{(5-x_2)^2}{3\cdot(5-x_1)^2+(5-x_2)^2}$;

dove $\underline{x} = (x_1, x_2)$, $x_1, x_2 \in [0,10]$, mentre $\Theta(x_1)$ è una piccola perturbazione che agisce sulla funzione del caso 1. Per ogni funzione è stato usato un campionamento con distribuzione spirale nello spazio di ingresso, al fine di creare training set costituiti da 400 esempi. In tal modo i training set non possiedono particolari strutture (cluster) nello spazio di ingresso, per cui è possibile apprezzare in modo significativo le capacità di sintesi basate sul clustering nello spazio congiunto. Come già menzionato, è stato usato un valore pari a 0.5 sia per λ che per λ_{ARC}. Il valore di M_{max} è stato posto pari a 40, cioè al 10% della cardinalità dei training set. Dai risultati riportati in tab. 4.1 si deduce che la sintesi OHCS è capace di garantire un'accuratezza che è alla stessa altezza dei più noti sistemi di modellamento neurali. In tab. 4.1 è riportato anche il numero ottimo di regole (M_{OHCS}) determinato dalla procedura OHCS.

Tabella 4.1 – MSE nel caso di differenti problemi di approssimazione funzionale.

Funzione	M_{OHCS}	OHCS	SOFRG	RBF	MLP
$f_1(\underline{x})$	44	$2.81\cdot10^{-2}$	$4.9\cdot10^{-2}$	$3.1\cdot10^{-2}$	$2.6\cdot10^{-2}$
$f_2(\underline{x})$	120	$5.77\cdot10^{-3}$	$3.8\cdot10^{-2}$	$9.8\cdot10^{-2}$	$1.73\cdot10^{-1}$
$f_3(\underline{x})$	64	$9.64\cdot10^{-3}$	$1.2\cdot10^{-2}$	$1.9\cdot10^{-2}$	$2.1\cdot10^{-2}$

La caratteristica più importante richiesta ad una rete neurale è quella di ottenere un'adeguata capacità di generalizzazione, cioè un'approssimazione soddisfacente quando ad essa è applicato un test set qualsiasi mai utilizzato durante l'apprendimento. Nel seguito si valuteranno le prestazioni in termini di capacità di generalizzazione, comparando le reti ANFIS ottenute mediante la procedura OHCS con quelle ottenute utilizzando alcune recenti, ed altrettanto innovative, tecniche di sintesi. In particolare, sarà considerato il metodo proposto in [Chen 1999] che combina un algoritmo adattativo di discesa a gradiente (Resilient Propagation - RPROP) ed i minimi quadrati ricorsivi (Recursive Least-Squares Error - RLSE). In [Chen 1999] è ampiamente discusso anche il confronto del suddetto metodo con quello più tradizionale che combina un algoritmo di discesa a gradiente (GD) con la RLSE.

A titolo di esempio, si riportano i risultati relativi al modellamento della seguente funzione:

$$f_4(\underline{x}) = 1 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5}$$

ottenuti utilizzando, come in [Chen 1999], un training set costituito da 216 campioni ed un test set di 125 campioni. I risultati sono mostrati in tab. 4.2 in termini di radice dell'errore quadratico medio (Root Mean Squared Error – RMSE) ed insieme al numero di regole contenute nelle rispettive reti ANFIS.

Tabella 4.2 – Confronto tra le sintesi OHCS, RPROP+RLSE e GD+RLSE.

$f_4(\underline{x})$	OHCS	RPROP+RLSE	GD+RLSE
Regole (M)	6	9	9
Training set	$1.16 \cdot 10^{-2}$	$2.58 \cdot 10^{-3}$	$2.93 \cdot 10^{-2}$
Test set	$1.34 \cdot 10^{-1}$	$2.18 \cdot 10^{-1}$	$2.59 \cdot 10^{-1}$

Diverse ottimizzazioni basate sulla discesa a gradiente sono state presentate nella letteratura tecnica per la sintesi di una rete neurofuzzy. Questo approccio è semplice, permette un calcolo facile e veloce dei parametri delle regole e, usualmente, garantisce una rapida convergenza. Tuttavia, il suo difetto principale è l'inevitabile intrappolamento nei minimi locali della funzione che si vuole minimizzare. Un confronto approfondito tra le varie tecniche di discesa a gradiente per la sintesi di reti

ANFIS è stato proposto in [Guély 1993]. La vicinanza di quel contesto al presente lavoro rende particolarmente appropriato il confronto della procedura OHCS con due tra le più efficaci tecniche di discesa a gradiente considerate in [Guély 1993]: la "Symmetric Triangular MF Sugeno rules" (TS-A) e la "Centred TS-A" (C-TS-A). Il training set ed il test set sono ottenuti utilizzando un campionamento casuale della seguente funzione:

$$f_5(\underline{x}) = \frac{(3e^{2x_1} + 2e^{-4x_2})}{170} \quad , \quad x_1, x_2 \in [-1,1]$$

La cardinalità di entrambi gli insiemi è P=250. Si riportano in tab. 4.3 il numero di regole usate in ciascuna delle reti ANFIS e l'errore di approssimazione, definito in questo caso dalla seguente quantità:

$$Q = \frac{1}{2}\sum_{i=1}^{P}(y_i - y'_i)^2$$

dove y'_i è l'uscita generata dalle reti. Anche in questo caso è possibile rendersi conto dell'efficacia della procedura proposta, osservando i risultati più significativi che sono quelli relativi al test set.

Tabella 4.3 - Confronto tra la sintesi OHCS ed alcune tecniche basate sulla discesa a gradiente.

$f_5(\underline{x})$	OHCS	TS-A	C-TS-A
Regole (M)	35	16	16
Training set	$3.5 \cdot 10^{-4}$	$1.3 \cdot 10^{-3}$	$9 \cdot 10^{-4}$
Test set	$1.1 \cdot 10^{-3}$	$4.3 \cdot 10^{-3}$	$4.7 \cdot 10^{-3}$

4.7 Analisi dei risultati

L'uso delle reti ANFIS per risolvere problemi di approssimazione funzionale è stato ampiamente sperimentato e validato nella letteratura tecnica del settore. L'utilizzo del clustering rende particolarmente efficaci le procedure di sintesi di reti ANFIS quando si dispone di esempi numerici della funzione da approssimare. Tuttavia, alcune di queste procedure sono caratterizzare da gravi problemi, anche in

funzione dello spazio dei dati in cui il clustering è applicato: ingresso, uscita e congiunto ingresso-uscita.

Gran parte del lavoro di ricerca presentato in questa tesi mirava proprio al superamento di questi difetti. Si è giunti di conseguenza alla definizione di un nuovo sistema di modellamento ANFIS, l'apprendimento del quale è basato sul clustering per iperpiani. Questo nuovo approccio ribalta l'usuale impostazione del problema di modellamento, basata sulla determinazione degli antecedenti di ogni regola con un clustering nello spazio di ingresso e, successivamente, sulla stima dei conseguenti con minimi quadrati e discesa a gradiente. Il clustering per iperpiani permette invece la determinazione diretta della struttura della funzione, ossia degli iperpiani associati ai conseguenti lineari di ogni regola di Sugeno del 1° ordine. Una volta che questo cruciale compito è stato assolto, è poi possibile determinare gli antecedenti di ogni regola risolvendo un opportuno problema di classificazione definito nello spazio di ingresso. A tale riguardo, in questo capitolo è stato illustrato l'uso dei classificatori fuzzy Min-Max a risoluzione adattativa.

La procedura di sintesi HCS corrisponde ad un'ottimizzazione parametrica delle reti ANFIS, essa può essere utilizzata per un qualsiasi numero di regole fissato a priori. Perciò, come per ogni altro algoritmo di apprendimento supervisionato, è necessaria anche un'ottimizzazione strutturale del sistema, dal momento che quest'ultima è pressoché obbligatoria al fine di ottenere una soddisfacente capacità di generalizzazione della risultante rete neurofuzzy. A tale proposito è stata concepita la procedura di ottimizzazione strutturale OHCS; si tratta di una procedura costruttiva che incrementa in modo progressivo il numero di regole nell'architettura ANFIS, allo scopo di determinarne automaticamente il numero ottimale.

L'efficacia di tale metodo è stata ampiamente verificata durante il lavoro di dottorato. I risultati presentati in questo capitolo sono evidentemente di stimolo a sostenere un ulteriore sviluppo della sintesi basata sull'approccio HCS/OHCS, soprattutto se si considerano le migliori prestazioni che essa garantisce rispetto ad altri sistemi di modellamento neurali ed alle stesse reti ANFIS ottenute con altri algoritmi di apprendimento.

Capitolo
5

STUDIO DI UN CASO: PREDIZIONE DI SEQUENZE CAOTICHE

5.1 Premessa

La letteratura tecnica sta mostrando un sempre più crescente interesse alla soluzione dei problemi di predizione con mezzi innovativi e più efficaci [Abarbanel 1996, Masulli 1999]. La soluzione di tali problemi consente infatti di gestire in modo più efficiente le risorse umane, economiche ed ambientali a disposizione. L'approccio generale alla soluzione di un problema di predizione è basato sulla soluzione di un opportuno problema di approssimazione funzionale. Consideriamo un generico predittore, utilizzato per predire una sequenza campionata s(n), ed un problema di approssimazione funzionale $y = f(\underline{x})$, $f:\Re^N \rightarrow \Re$. Uno degli approcci più semplici possibili è quello basato sul modello lineare autoregressivo (AR), dove ogni vettore di ingresso $\underline{x}_n$ è costituito da N esempi consecutivi della sequenza s(n) mentre l'uscita è costituita dal campione da predire a distanza m:

$$\begin{aligned} &\underline{x}_n = [s(n)\ s(n-1)\ \ldots\ s(n-N+1)]\ ,\quad y_n = s(n+m) \\ &f_{AR}(\cdot) = -\sum_{j=1}^{N} \lambda_j x_{nj} \ \Rightarrow\ \hat{s}(n+m) = -\sum_{j=1}^{N} \lambda_j s(n-j+1) \end{aligned} \qquad (5.1)$$

dove $\hat{s}(n+m)$ indica la stima del valore vero s(n+m) della sequenza. I coefficienti λ_j, j=1…N, della funzione $f_{AR}(\cdot)$ possono essere determinati sulla base delle proprietà statistiche globali della sequenza s(n) come, ad esempio, la sua funzione di autocorrelazione [Papoulis 1991].

Le prestazioni di un predittore dipendono da quanto è accurato il modellamento dei sistemi fisici incogniti che generano le sequenze da predire. Quelle relative all'osservazione dei fenomeni naturali spesso possiedono un comportamento caotico che è peraltro tipico della maggior parte dei sistemi fisici reali. L'uso del predittore lineare (5.1) potrebbe rivelarsi ampiamente inadeguato in tali situazioni, dato che esso è basato su un semplice modello di approssimazione lineare e su un metodo rudimentale di costruzione dei vettori di ingresso.

Nel caso di sequenze caotiche, s(n) può essere considerata come l'uscita di un sistema autonomo osservabile solo tramite questa uscita che, d'ora in poi, verrà denotata con y(n). In questo capitolo verrà dato un breve cenno ai risultati relativi alla teoria della *ricostruzione dinamica*, che suggerisce come utilizzare in modo ottimale i campioni della sequenza osservata per impostare il problema di approssimazione funzionale. A causa dell'intrinseca non linearità e non stazionarietà del sistema caotico, la funzione approssimante f(·) dovrà essere scelta con cura, utilizzando un modello certamente non lineare ed una procedura di sintesi data driven. Poiché le reti ANFIS soddisfano a tali requisiti, nel seguito di questo capitolo verranno analizzate le prestazioni delle reti ANFIS sintetizzate con la procedura OHCS illustrata nel cap. 4. Verranno presi in considerazione per tale scopo alcuni problemi di predizione di particolare interesse pratico.

5.2 Teoria della ricostruzione dinamica

La ragione principale per cui si usa la ricostruzione dinamica è quella di dare un "senso fisico" alla serie temporali osservabili da un sistema caotico, by-passando così il bisogno di una conoscenza matematica dettagliata delle dinamiche che soggiacciono al suo interno. Tale sistema è caratterizzato da una grandezza di stato $\underline{x}(t) = [x_1(t)\ x_2(t)\ ...\ x_N(t)]$ definita in uno spazio (*spazio di stato*) avente un'opportuna dimensione N. La rappresentazione analitica del sistema S sconosciuto è, nel caso autonomo e tempo continuo, data da

$$\begin{cases} \underline{x}'(t) = F[\underline{x}(t)] \\ y(t) = H[\underline{x}(t)] \end{cases} \qquad (5.2)$$

in cui $\underline{x}(t)$, F[·] e H[·] sono ignote. L'unica informazione che abbiamo è rappresentata dalla grandezza misurata y(t), la quale è generalmente una grandezza scalare. Eventualmente, y(t) potrebbe anche essere mascherata da rumore dovuto a diverse cause. Nel seguito supporremo, per semplicità, che il rumore non sia presente.

L'andamento della y(t) dipende dall'evoluzione dinamica del sistema, cioè dalla traiettoria dello stato $\underline{x}(t)$ nello spazio di stato. Tale evoluzione può essere di vari tipi; quello che interessa è la situazione che si stabilisce dopo lo smorzamento del transitorio. A tale riguardo, si possono avere i seguenti andamenti :

1. arrivo ad uno stato finale stabile;
2. andamento periodico (ciclo limite);
3. andamento non-periodico contenuto in una regione limitata.

L' ultimo caso è il più interessante perché comune a molte situazioni reali. L'oggetto che viene disegnato dalle traiettorie non-periodiche è un ente geometrico che prende il nome di *attrattore*. Tipicamente, gli attrattori di sistemi caotici hanno una dimensione di valore non intero, detta anche *dimensione frattale*.

La grandezza osservata y(t) è chiaramente legata in modo diretto all'evoluzione dello stato $\underline{x}(t)$ che, d'altra parte, non è osservabile. Esiste tuttavia la possibilità di ottenere una specie di replica di quello che succede nello spazio di stato, utilizzando uno spazio individuato a partire dalla sola y(t). Tale spazio prende il nome di *spazio di ricostruzione* (SR), la cui esistenza è garantita dal *teorema di Takens* [Takens 1981, Haykin 1995] meglio noto come *"teorema dell'embedding"*. Il teorema di Takens permette di stabilire un'equivalenza tra il sistema dinamico non lineare originale e il sistema ricostruito nell'ambito di un cambiamento uniforme di coordinate. La base di questo teorema è il legame che esiste tra y(t) e $\underline{x}(t)$ nella (5.2). Nei sistemi caotici ciò comporta che le coordinate dello spazio SR sono legate a quelle dello spazio di stato tramite trasformazioni non lineari. Usando trasformazioni arbitrarie di y(t) si perverrebbe in ogni caso ad una rappresentazione significativa dello spazio di stato originale. Ciò comporterebbe una semplice modifica del legame tra y(t) e $\underline{x}(t)$, mentre l'oggetto ricostruito conserverebbe le caratteristiche dinamiche peculiari di quello originale.

La trasformazione più naturale è quella che fa uso di y(t) e delle sue derivate successive. Tale scelta va però modificata per tenere conto dell'effetto devastante che avrebbe l'eventuale presenza del rumore. Infatti, l'operazione di derivazione rispetto al tempo costituisce un filtraggio passa-alto, che esalterebbe l'entità del rumore presente nelle coordinate di SR. Un modo semplice per ovviare a questo inconveniente, conservando la scelta naturale citata, è quello di usare y(t) ed un numero di suoi campioni ritardati. L'uso di campioni ritardati di y(t) equivale a mettere in gioco le sue derivate successive senza gli inconvenienti legati all'eventuale presenza del rumore.

Il teorema dell'embedding è basato su una teoria matematica molto complessa, l'esposizione della quale esula dallo scopo di questo lavoro. Per una visione del teorema in termini più pratici, possiamo invece considerare un sistema dinamico sconosciuto che evolve nel tempo discreto. L'evoluzione dello stato è in questo caso descritta dalla seguente equazione non lineare alle differenze:

$$\underline{x}(n+1) = F[\underline{x}(n)] \qquad (5.3)$$

dove $\underline{x}(n)$ è il vettore di stato D-dimensionale del sistema all'istante $n\tau_s$. Nel seguito si supporrà un periodo di campionamento τ_s normalizzato all'unità. La serie temporale y(n), osservabile all'uscita del sistema, può essere definita in termini del vettore di stato $\underline{x}(n)$ come segue:

$$y(n) = H[\underline{x}(n)] + w(n) \qquad (5.4)$$

dove w(n) denota l'eventuale rumore supposto additivo. Il rumore w(n) può anche essere usato per tenere conto degli effetti combinati dovuti alle imperfezioni ed alle imprecisioni sull'uscita osservata y(n).

Secondo il teorema di Takens, la struttura geometrica degli attrattori di un sistema caotico può essere "dispiegata" ("sbrogliata") utilizzando le osservazioni y(n); in altre parole, la traiettoria dello stato del sistema nello SR può essere ottenuta osservando il vettore

$$\underline{y}_R(n) = [y(n)\ \ y(n-T)\ \ \ldots\ \ y(n-(D-1)T)] \qquad (5.5)$$

essendo T un intero positivo chiamato *ritardo di embedding normalizzato* (ciascuna delle componenti del vettore $\underline{y}_R(n)$ è separata dalla precedente e dalla successiva da

un intervallo di tempo pari a T volte il tempo di campionamento τ_s). L'ipotesi fondamentale del teorema è che la dimensione D del vettore ricostruito $\underline{y}_R(n)$ sia maggiore di 2d, avendo indicato con d la dimensione effettiva dello spazio di stato del sistema. La condizione D > 2d è una condizione sufficiente ma non necessaria. La procedura che permette di ricostruire l'evoluzione dello stato di un sistema caotico in uno SR è chiamata *embedding* ed è sinteticamente illustrata in fig. 5.1; il valore intero D utilizzato per la ricostruzione dinamica è chiamato *dimensione di embedding*.

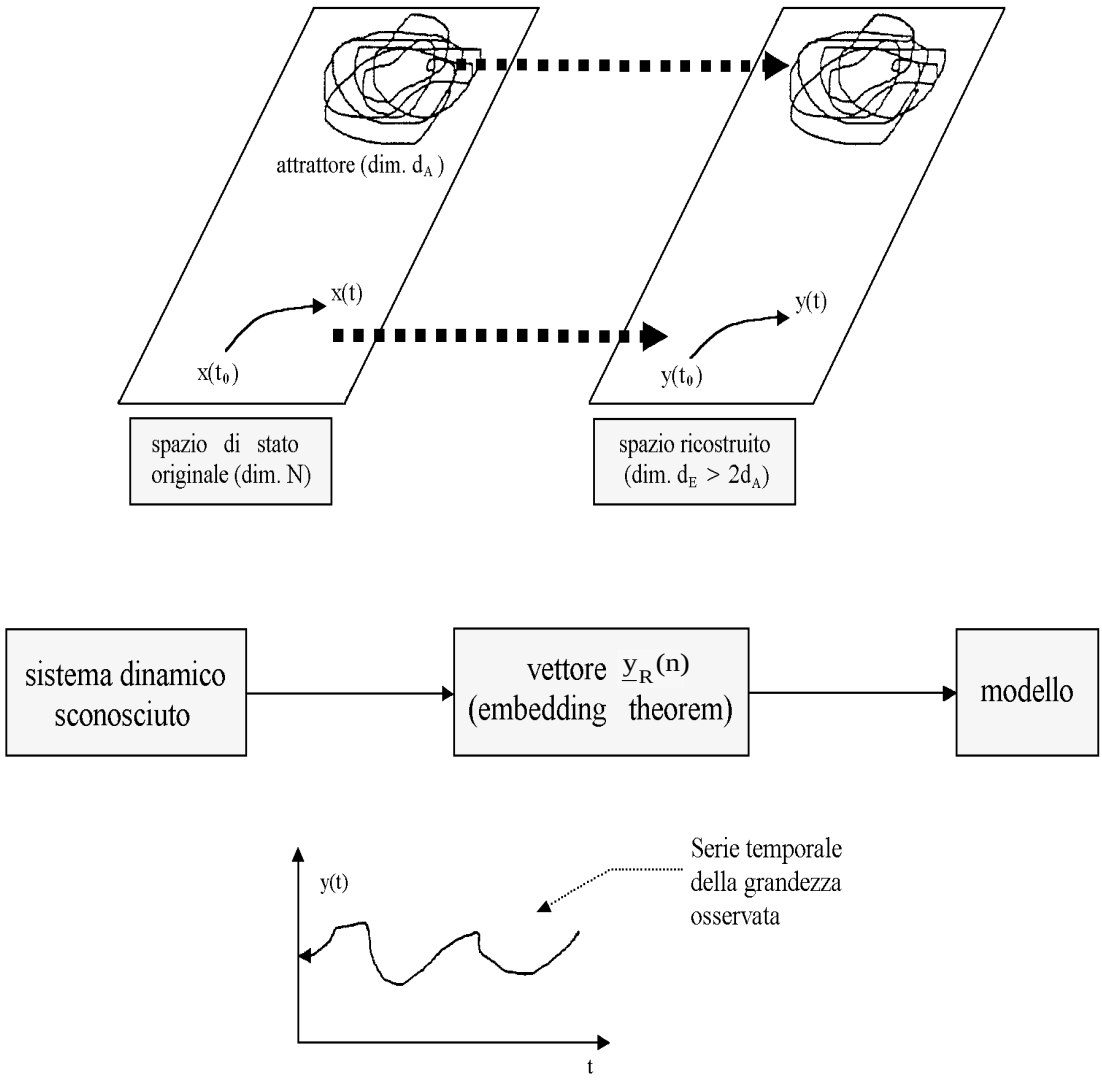


Figura 5.1 – Schema di principio della procedura di embedding.

Il teorema dell'embedding ha un'implicazione importante: l'evoluzione dei punti $\underline{y}_R(n) \Rightarrow \underline{y}_R(n+1)$ nello spazio di ricostruzione segue quella dell'attrattore incognito $\underline{x}(n) \Rightarrow \underline{x}(n+1)$ nello spazio di stato originale. Ne consegue che molte proprietà importanti del vettore di stato non osservabile $\underline{x}(n)$ sono riprodotte senza ambiguità da $\underline{y}_R(n)$ nello spazio di ricostruzione. Dallo studio del legame $\underline{y}_R(n) \Rightarrow \underline{y}_R(n+1)$ si può quindi determinare quella funzione non lineare che modella il sistema dinamico in oggetto e che ci permette di predire i campioni futuri sulla base di quelli passati. E' quindi evidente che stiamo risolvendo un problema di predizione della sequenza caotica y(n), ma anche l'identificazione del sistema caotico stesso. Ciò è ottenuto attraverso la soluzione del problema di approssimazione funzionale relativo alla suddetta relazione tra i campioni $\underline{y}_R(n)$. Questo concetto sarà ampiamente approfondito nei successivi paragrafi, in quanto è alla base dell'applicazione dei sistemi di modellamento neurofuzzy qui proposti per la soluzione dei problemi di predizione.

5.2.1 Giustificazione qualitativa del teorema di Takens

In base alle rappresentazioni considerate in precedenza per i sistemi dinamici, ed in particolare quelle nel tempo discreto (5.3) e (5.4), risulta evidente la possibilità di predire lo stato successivo $\underline{x}(n+1)$ a partire da quello attuale $\underline{x}(n)$. Tuttavia, se è vero che è possibile conoscere il valore dell'uscita y(n) quando è noto lo stato $\underline{x}(n)$, l'inverso però non vale. Infatti, $\underline{x}(n)$ è un vettore costituito da più componenti mentre noi, all'istante n-esimo, ne conosciamo una sola: l'uscita y(n). A questo punto entra in gioco il teorema di Takens, che suggerisce un modo semplice per ovviare a questo inconveniente sostituendo le grandezze mancanti con le versioni ritardate dell'unica grandezza disponibile y(n). In tal modo, il vettore $\underline{y}_R(n)$ nella (5.5) diviene in corrispondenza biunivoca con $\underline{x}(n)$ e perciò il legame $\underline{x}(n) \Rightarrow \underline{x}(n+1)$ è equivalentemente rappresentato dal legame $\underline{y}_R(n) \Rightarrow \underline{y}_R(n+1)$.

5.2.2 Estensione al caso di sistemi non-autonomi

La teoria dell'embedding si riferisce ai soli sistemi autonomi, tuttavia si può pensare, con la dovuta cautela, alla possibilità di un'eventuale estensione anche a sistemi che non siano tali. Nel caso non-autonomo il sistema è generalmente rappresentato dal seguente sistema di equazioni:

$$\begin{cases} \underline{x}'(t) = F[\underline{x}(t), \underline{u}(t)] \\ y(t) = H[\underline{x}(t), \underline{u}(t)] \end{cases}$$

dove $\underline{u}(t)$ rappresenta l'eccitazione esterna al sistema. Nella situazione in cui $\underline{u}(t)$ sia un'eccitazione specifica, e ci riferiremo solo a questa, essa può essere considerata alla stregua di un parametro del sistema. Di conseguenza, il sistema può essere trattato come se fosse autonomo.

Sulla base di queste ipotesi, relativamente alla costruzione di un predittore, la teoria valida nel caso autonomo è estendibile anche al caso non autonomo. Tuttavia, il predittore risultante non è un modello valido del sistema in quanto dipende anche dall'eccitazione. La trattazione attuale, quindi, non riguarda il modellamento del sistema, ma la possibilità di predire una grandezza. A tale scopo, è quindi possibile estendere quanto detto per un sistema autonomo anche al caso di sistemi non-autonomi.

5.2.3 Scelta dei parametri di embedding

Per applicare con successo la teoria della ricostruzione dinamica abbiamo bisogno di stime affidabili sia della dimensione di embedding D che del ritardo normalizzato T. Nel seguito daremo solo un breve cenno ai metodi più comuni che possono essere utilizzati, anche perché sono oramai disponibili svariati strumenti software che risultano di grande aiuto nel calcolo dei parametri di embedding.

Sfortunatamente, il teorema dell'embedding non ha nulla da dire sulla scelta del ritardo di embedding normalizzato. Infatti, il teorema permette l'uso di qualsiasi ritardo T per quanto sia infinitamente lungo l'intervallo temporale della serie di campioni a disposizione. Ad ogni modo, nella pratica dobbiamo sempre lavorare con insiemi di dati osservati di lunghezza finita. Il metodo corretto per scegliere T è

riconoscere che il ritardo di embedding normalizzato sia abbastanza grande da fare risultare y(n) e y(n-T) essenzialmente indipendenti l'uno dall'altro per servire da coordinate dello stesso punto nello SR, ma non così indipendenti da essere completamente scorrelate tra di loro (e quindi appartenere a due punti diversi nello SR). A tale proposito è possibile usare un qualsiasi metodo per la misura di indipendenza tra due variabili. Uno dei criteri più affermati è quello della *Mutua Informazione di Shannon* (Average Mutual Information - AMI), che suggerisce di scegliere come valore di T il primo minimo dell'AMI [Fraser 1986].

Riguardo alla dimensione di embedding, è interessante notare che la condizione fornita dal teorema di Takens (D>2d), quando soddisfatta, rende possibile nello SR l'eliminazione delle intersezioni di un'orbita dell'attrattore con se stessa. Tali intersezioni deriverebbero dalla proiezione di quell'orbita in uno spazio a dimensione troppo piccola che, ovviamente, risulterebbe meno adeguata ai fini della risoluzione di un problema di modellamento (predizione) sull'attrattore.

Una delle caratteristiche importanti di un attrattore è che generalmente esso risulta essere un oggetto compatto nello spazio di stato. Pertanto, punti di una sua orbita possono avere dei "vicini" che giacciono su orbite differenti. Ma se si è considerata una dimensione di embedding troppo piccola per "racchiudere" l'attrattore, non tutti i punti che giacciono vicini l'un l'altro possono considerarsi realmente tali. Alcuni di questi punti, infatti, appaiono vicini non a causa dell'evoluzione dinamica del sistema, ma perché la struttura geometrica dell'attrattore è stata proiettata in uno spazio a dimensione troppo piccola. E' essenziale allora scegliere una dimensione D tale da garantire che le traiettorie di $\underline{y}_R(n)$ non presentino intersezioni o "vicinanze" per effetti di proiezione, le quali darebbero luogo ad ambiguità in fase di predizione (nel punto comune a due traiettorie non sappiamo qual è il punto successivo). Questa situazione è ben illustrata in fig. 5.2.

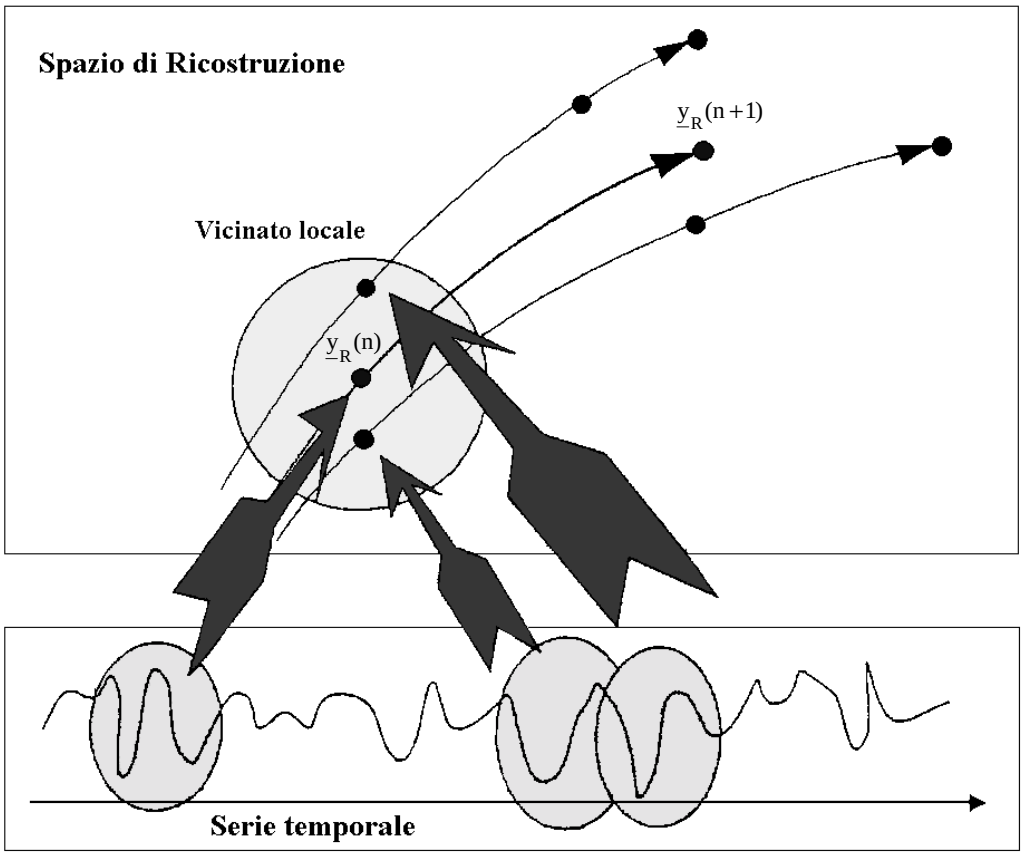


Figura 5.2 - Determinazione dei punti nello SR a partire dalla serie dei campioni osservati.

La dimensione di embedding può essere minore di 2d, in quanto tale condizione è solo sufficiente. Un criterio naturale per scoprire errori nella procedura di embedding si basa sulla constatazione che, nel passaggio della dimensione di embedding da D a D+1, l'incremento della distanza (opportunamente definita) tra $\underline{y}_R(n)$ e l'eventuale falso vicino $\underline{y}_R^{(f)}(n)$ è relativamente grande. La tecnica che si usa è quindi la seguente:

> *fissato un valore D per la dimensione dello spazio di ricostruzione, si determina per ogni punto dell'attrattore ricostruito in quello spazio il punto più vicino e si verifica se, incrementando di uno il valore di D, il punto determinato in precedenza risulta essere ancora "vicino".*

La percentuale dei falsi vicini, man mano che si sale con la dimensione dello SR, va in generale diminuendo; ciò accade perché diminuiscono i "ripiegamenti" dovuti alle proiezioni dell'attrattore su uno spazio a dimensione inferiore. Intuitivamente, la scelta

corretta per la dimensione di embedding è quella in corrispondenza della quale è minima la percentuale dei falsi vicini. Il procedimento appena descritto prende appunto il nome di *"metodo dei falsi vicini"* [Abarbanel 1993, Kennel 1992].

In linea con il discorso sin qui affrontato, è importante avere un'idea sull'origine dei vicini di un punto; si hanno quattro casi:

- *vicini veri*: sono vicini per effetto della dinamica del sistema;
- *vicini falsi*: sono vicini per effetto di una proiezione da uno spazio di dimensione maggiore;
- *vicini da sovracampionamento*: derivano da un intervallo di campionamento troppo piccolo, possono essere facilmente eliminati;
- *vicini per quantizzazione*: il numero di bit usati non è sufficiente, tali vicini sono difficili da eliminare.

E' interessante notare che l'aumento progressivo della dimensione di embedding, nel metodo dei falsi vicini, può non portare all'annullamento del numero di questi ultimi. Ne possono rimanere un certo numero, il che è in genere indicativo della presenza di rumore.

5.3 Le reti neurofuzzy come sistemi di modellamento per la ricostruzione dinamica

Come abbiamo visto nel paragrafo precedente, il teorema di Takens garantisce, qualora le sue condizioni siano rispettate, l'esistenza di una funzione di mapping che trasforma lo stato ricostruito corrente $\underline{y}_R(n)$ nello stato successivo $\underline{y}_R(n+1)$. La mappa predittiva $f : \Re^D \rightarrow \Re$ può essere espressa nel seguente modo:

$$y(n+1) = f[\underline{y}_R(n)] \qquad (5.6)$$

L'approssimazione di questa funzione è il fulcro del modellamento poiché, una volta determinata f(·), la funzione di transizione dello stato può essere ottenuta a partire da essa mediante semplici operazioni matriciali.

L'equazione (5.6) definisce un modello autoregressivo non lineare e deterministico del segnale. Ciò pone una base teorica per il modellamento della dinamica del sistema, nel senso che apre alla possibilità di costruire un modello per la sua approssimazione a partire da una serie di campioni osservati. La procedura di modellamento dinamico consiste di due passi successivi: il primo è trasformare, utilizzando una tecnica di embedding, la serie temporale osservata in una traiettoria nello spazio di ricostruzione; il secondo passo consiste nella costruzione del modello predittivo (5.6) a partire dalla traiettoria nello spazio di stato ricostruito. Se abbiamo fin qui discusso del primo passo, passiamo ora alla fase cruciale rappresentata dal secondo passo. Si tratta, cioè, di risolvere un problema di approssimazione funzionale di tipo data driven, in quanto le informazioni disponibili per l'apprendimento sono contenute nei campioni osservati dell'uscita del sistema.

Il problema di ricostruire il modello di un sistema dinamico a partire dalla sua manifestazione (la grandezza y(n) osservata) è, in realtà, un problema *mal posto* [Brown 1991]. La definizione di problema mal posto può essere ricavata da quella di problema *ben posto* data da Hadamard, facendo vedere che il problema in questione non soddisfa a quest'ultima ipotesi. Un problema è ben posto se soddisfa alle seguenti tre condizioni:

1. *Esistenza*. Per ogni vettore di ingresso $\underline{x}$ esiste un vettore di uscita $\underline{y} = g(\underline{x})$.

2. *Unicità*. Per ogni coppia di vettori di ingresso $\underline{x}$, $\underline{z}$ si ha $g(\underline{x}) = g(\underline{z})$ se e solo se $\underline{x} = \underline{z}$.

3. *Continuità*. La funzione di mapping è continua, cioè per ogni $\varepsilon > 0$ esiste un $\delta = \delta(\varepsilon)$ tale che la condizione $d_X(\underline{x}, \underline{z}) < \delta$ implica $d_Y(g(\underline{x}), g(\underline{z})) < \varepsilon$, dove $d(\cdot)$ è la distanza tra i due argomenti nei loro rispettivi spazi.

Se una qualsiasi di queste condizioni non è soddisfatta il problema si dice mal posto. Nel nostro caso si possono riscontrare violazioni della seconda e della terza condizione. Infatti, potrebbe non esserci sufficiente informazione nella serie temporale osservata per potere ricostruire la dinamica del sistema in modo unico. Inoltre, l'inevitabile presenza di rumore, o di una qualche altra forma di imprecisione nell'acquisizione della serie, attribuirebbe incertezza alla ricostruzione dinamica. In

particolare, se il livello del rumore è troppo alto è possibile che venga violata la condizione di continuità.

La soluzione di un problema mal posto può essere ottenuta solo aggiungendo ulteriore informazione al problema stesso. E' in questo ambito che tornano utili sistemi di modellamento flessibili e robusti, come quelli presenti nell'ambito delle reti neurali. Un modo classico per aggiungere informazione è quello di ricorrere alla *teoria della regolarizzazione di Tikhonov* [Tikhonov 1977], basata sull'ipotesi di legame regolare nella funzione ingresso-uscita da approssimare. Non entreremo ulteriormente nel dettaglio di questo approccio, peraltro già ampiamente discusso in letteratura. Ciò che è importante notare è che tale metodo mira essenzialmente a massimizzare la capacità di generalizzazione del modello, vincolando la minimizzazione dell'errore di predizione così come accade nei metodi di ottimizzazione strutturale fin qui discussi.

Ai fini di valutare ulteriormente le prestazioni della sintesi OHCS per le reti ANFIS, nel successivo paragrafo saranno presentati i risultati relativi all'applicazione di tale metodo in merito a problemi di predizione di alcune serie reali di particolare interesse pratico. Abbiamo scelto tali sequenze perché sono caratterizzate da un comportamento prevalentemente caotico, legato alle evoluzioni di sistemi fisici reali lo studio dei quali è di notevole interesse pratico. Le sequenze studiate sono ottenute dall'osservazione, effettuata in precise zone del centro di Roma, di alcuni agenti inquinanti e dei consumi di energia elettrica.

5.4 Analisi delle prestazioni nei problemi di predizione

L'efficacia della procedura di sintesi OHCS sarà valutata comparandone la capacità di predizione rispetto a quella di due differenti tipi di predittore: il primo è il predittore lineare introdotto al par. 5.1; il secondo è una rete ANFIS sintetizzata mediante un algoritmo di *clustering sottrattivo* (Subtractive Clustering – SUBCL) per l'estrazione delle regole [Chiu 1994] e, successivamente, con un metodo di tuning dei parametri basato su minimi quadrati e discesa a gradiente [Jang 1997].

Tutti questi modelli, che implementano l'approssimazione funzionale della (5.6), sono allenati sui primi 2000 campioni della serie osservata y(n). Questi campioni sono

anche utilizzati per valutare i parametri di embedding T e D utilizzando, rispettivamente, i metodi della mutua informazione e dei falsi vicini, entrambi menzionati al par. 5.2.3. La performance in termini di errore di predizione è valutata sui successivi 600 campioni della sequenza. L'espressione utilizzata per valutare l'errore di predizione è quella dell'errore quadratico medio normalizzato (Normalized Mean Squared Error – NMSE), definito come il rapporto tra l'errore quadratico medio e la varianza della sequenza da predire:

$$\mathrm{NMSE} = \frac{\sum_n \left| y(n) - \hat{y}(n) \right|^2}{\sum_n \left| y(n) - y_{ts} \right|^2}$$

dove y_{ts} è la media dei campioni del test set e $\hat{y}(n)$ è il valore predetto.

Le prestazioni del predittore proposto (nel seguito, per ragioni di sintesi, indicato con OHCS) sono confrontate anche con quelle prodotte da alcuni modelli di riferimento molto semplici. E' infatti evidente che ha tanto più senso progettare un sistema di modellamento complesso, come quello realizzato da una rete ANFIS, quanto più esso si rivela vantaggioso rispetto a modelli predittivi estremamente semplici [Panella 2001(bis)]. Considereremo nel seguito due di questi metodi di riferimento; per un generico passo di predizione m, essi sono definiti dalle seguenti relazioni:

1. Predittore "No Change" (NC):

   $y^{NC}(n+m)=y(n)$

2. Predittore "Unconditional Mean" (UM):

   $y^{UM}(n+m) = \bar{y}_{tr}$

   dove $\bar{y}_{tr}$ rappresenta il valore medio di y(n) nel training set.

Il guadagno di predizione ottenuto con il predittore OHCS rispetto a ciascuno di questi predittori sarà indicato con i termini guadagno-NC e guadagno-UM. Entrambi sono definiti come il rapporto tra il valore di NMSE ottenuto dal predittore OHCS e quello ottenuto, rispettivamente, dal predittore NC o UM. In particolare, il guadagno-UM fornisce informazioni riguardo al cambio di comportamento (ovvero del bacino di attrazione nello SR) del test set rispetto al training set, in quanto esso coincide con

l'NMSE dell'OHCS se $\bar{S}_{tr} = \bar{S}_{ts}$. Il livello di caos presente nelle sequenze è misurato dall'*Entropia Spazio-Temporale* (Spatio-Temporal Entropy - STE), definita e misurata attraverso gli strumenti forniti in [Kononov 1999]. Il valore di STE si aggira attorno al 50% in caso di sequenze puramente caotiche, mentre tende al 100% in caso di rumore e allo 0% in caso di sequenze periodiche.

Le prime due sequenze considerate sono relative ai livelli di Ozono (in μg) e di rumore acustico (in dB) misurati in una precisa zona del centro di Roma. Tali sequenze sono state ottenute con un tasso di campionamento pari a 5 minuti. La terza sequenza è invece relativa al consumo di energia elettrica (in MW), sempre misurato nella città di Roma. I valori sono ottenuti in questo caso con un tasso di campionamento pari ad 1 ora. I risultati dei test effettuati secondo quanto precedentemente indicato sono riportati in tab. 5.1, con evidente significato dei valori riportati nelle varie colonne.

Tabella 5.1- Risultati dei test di predizione su sequenze reali.

Test	T	D	STE	Lineare	SUBCL	OHCS			
						NMSE	M	guad-NC	guad-UM
Ozono	3	5	58%	$2.43 \cdot 10^{-1}$	$2.19 \cdot 10^{-1}$	$1.72 \cdot 10^{-1}$	16	$2.79 \cdot 10^{-1}$	$1.30 \cdot 10^{-1}$
Rumore acustico	4	14	65%	$4.39 \cdot 10^{-1}$	$3.04 \cdot 10^{-1}$	$3.47 \cdot 10^{-1}$	9	$3.59 \cdot 10^{-1}$	$2.37 \cdot 10^{-1}$
Carico elettrico	7	5	35%	$4.95 \cdot 10^{-2}$	$3.45 \cdot 10^{-2}$	$9.93 \cdot 10^{-3}$	13	$2.17 \cdot 10^{-1}$	$1.04 \cdot 10^{-1}$

5.5 Considerazioni sui risultati ottenuti

Come già dimostrato nel cap. 4, le reti ANFIS sono sistemi di modellamento neurofuzzy particolarmente adatti alla soluzione di problemi di approssimazione funzionale. In questo capitolo abbiamo analizzato le prestazioni della tecnica di sintesi OHCS per la soluzione di problemi di predizione basati sulla ben nota tecnica della ricostruzione dinamica per il modellamento di sistemi caotici. Nella fattispecie, le reti ANFIS ottenute utilizzando la procedura OHCS sono state usate per la predizione di

sequenze relative all'osservazione di fenomeni reali. I risultati ottenuti dimostrano che tale procedura di sintesi è particolarmente vantaggiosa, sia perché funziona meglio di altri sistemi di modellamento già affermati in letteratura (predittore lineare, ANFIS sintetizzata con il clustering sottrattivo), sia perché le sue prestazioni sono di gran lunga superiori a quelle di sistemi di predizione rudimentali e poco costosi da implementare (NC, UM). Questi risultati sono incoraggianti affinché si possa estendere il campo di applicazione della procedura OHCS anche a problemi di predizione più critici, sia di tipo industriale che di tipo ambientale. In tali applicazioni, infatti, anche piccoli miglioramenti nella predizione possono corrispondere ad una gestione molto più efficiente delle risorse umane, economiche ed ambientali a disposizione.

BIBLIOGRAFIA

Abarbanel H.D.I., Kennel M.B., “Local False Neighbours and Dynamical Dimensions from Observed Chaotic Data”, *Phys. Rev. E*, Vol. 47, pp. 3057-3068, 1993.

Abarbanel H.D.I., *Analysis of Observed Chaotic Data*, Springer-Verlag, Berlin Heidelberg New York, 1996.

Abu-Mustafa Y.S., “Hints”, *Neural Computation*, pp. 639-671, 1995.

Anderberg M.R., *Cluster Analysis for Applications*, Academic Press, Inc., New York, 1973.

Bezdek J.C., “A Review of Probabilistic, Fuzzy, and Neural Models for Pattern Recognition”, *Journal of Intelligent and Fuzzy Systems*, vol. 1 (1), pp. 1–25, John Wiley & Sons, Inc., 1993.

Bishop C.M., *Neural Networks for Pattern Recognition*, Oxford Univ. Press Inc., N.Y., 1995.

Bonissone P.P., “Soft computing: the convergence of emerging reasoning technologies”, *Soft Computing*, Vol. 1, No. 1, pp. 6-18, aprile, 1997.

Brown R., Bryant P., Abarbanel H.D.I., “Computing the Lyapunov Spectrum of a Dynamical System from an Observed Time Series”, *Phys. Rev. A*, Vol. 43, No. 6, pp. 2787-2806, marzo, 1991.

Chen M.S., Liou R.J., “An efficient Learning Method of Fuzzy Inference Systems”, *Proc. of FUZZ-IEEE ‘99*, Seoul, Korea, Vol. II: 634-638, 1999.

Chiu S., "Fuzzy Model Identification Based on Cluster Estimation", *Journal of Intelligent & Fuzzy Systems*, Vol. 2, No. 3, 1994.

Ciccarella G., Marietti P., Trifiletti A., *Strumentazione e Misure Elettroniche*, Ed. Masson S.p.A., Milano, 1993.

Costantini G., Antici P., Panella M., Frattale Mascioli F.M., "Nonexclusive classification of musical sources using pattern recognition", *Proc. of EIS'2000*, Paisley, Scozia, U.K., 27-30 giugno, 2000.

Costantini G., Antici P., Panella M., Frattale Mascioli F.M., "Nonexclusive classification and recognition of traditional musical instruments", *Proc. of CIM 2000*, L'Aquila, 4-6 settembre, 2000(bis).

Davies D.L., Bouldin D.W., "A Cluster Separation Measure", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI 1, pp. 224-227, 1979.

Dixon J.K., "Pattern recognition with partly missing data", *IEEE Transactions on Systems, Man and Cybernetics*, SMC 9, pp. 617-621, 1979.

Everitt B.S., *Cluster Analysis*, John Wiley & Sons, Inc., 1974.

Forgy E., "Cluster analysis of multivariate data: efficiency versus interpretability of classifications", *Biometrics*, No. 21, p. 768 (abstract), 1965.

Fraser A.M., Swinney H.L., "Independent Coordinates for Strange Attractors from Mutual Information", *Phys. Rev. A*, Vol. 33, No. 2, pp. 1134-1140, febbraio, 1986.

Frattale Mascioli F.M., Rizzi A., Martinelli G., "Compactness-separability optimization of fuzzy clusters", *Proceedings of ISIS'97*, Reggio Calabria, pp. 452-457, settembre, 1997.

Frattale Mascioli F.M., Martinelli G., "A constructive approach to neuro-fuzzy networks", *Signal Processing*, Vol. 64, No. 3, pp. 347-358, 1998.

Frattale Mascioli F.M., Rizzi A., Panella M., Martinelli G., "Clustering with uncostrained hyperboxes", *IEEE Int. Fuzzy Systems Conference*, Seul, Corea, Vol. II: 1075-1080, agosto, 1999.

Frattale Mascioli F.M., Rizzi A, Panella M, Martinelli G., "Scale-Based Approach to Hierarchical Fuzzy Clustering", *Signal Processing*, Vol. 80, No. 6, pp. 1001-1016, maggio, 2000.

Frattale Mascioli F.M., Panella M., Rizzi A., Martinelli G., "Scale-Based Clustering with Latent Variables", *Proc. of EUSIPCO 2000*, Tampere, Finlandia, 5-8 settembre, 2000(bis).

Frattale Mascioli F.M., Mancini A., Rizzi A., Panella M., Martinelli G., "Neurofuzzy Approximator based on Mamdani's Model", *Proc. of WIRN VIETRI-2001*, Vietri Sul Mare, Salerno, 17-19 maggio, 2001.

Gower J.C., Legendre P., "Metric and Euclidean properties of dissimilarity coefficients", *Journal of Classification*, No. 3, pp. 5-48, 1986.

Guély F., Siarry P., "Gradient Descent Method for Optimizing Various Fuzzy Rule Bases", *Proc. of Second IEEE International Conference on Fuzzy Systems*, Vol.2, pp. 1241-1246, 1993.

Guillame S., "Designing Fuzzy Inference Systems from Data: an Interpretability Oriented Review", *IEEE Transactions on Fuzzy Systems*, Vol. 9, No. 3, pp. 426-443, 2001.

Hartman E.J., Keeler J.D., Kowalski J.M., "Layered Neural Networks with Gaussian Hidden Units as Universal Approximations", *Neural Computation*, No. 2, pp. 210-215, 1990.

Hathaway R.J., Bezdek J.C., "Switching regression models and fuzzy clustering", *IEEE Transactions on Fuzzy Systems*, Vol. 1, No. 3, pp. 195-204, 1993.

Haykin S., Li X.B., "Detection of Signals in Chaos", *Proceedings of the IEEE*, Vol. 83, No. 1, pp. 94-122, gennaio, 1995.

Haykin S., *Neural Networks, a Comprehensive Foundation*, 2nd ed., Prentice Hall, 1999.

Jain A.K., Dubes R.C., *Algorithms for Clustering Data*, Prentice-Hall, Englewood Cliffs, 1988.

Jang J.-S.R., Sun C.-T., "Functional equivalence between radial basis function networks and fuzzy inference systems", *IEEE Transactions on Neural Networks*, Vol. 4, No.1, pp. 156-159, gennaio, 1993.

Jang J.-S.R., Sun C.-T., Mizutani E., *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, Prentice Hall, Upper Saddle River, NJ, 1997.

Kandel E.R., Schwartz J.H., Jessell T.M., *Principi di Neuroscienze.* Casa Editrice Ambrosiana, Seconda edizione, 1994.

Kennel M.B., Brown R., Abarbanel H.D.I., "Determining Minimum Embedding Dimension using a Geometrical construction", *Phys. Rev.*, Vol. 45, pp. 3403-3411, 1992.

Kim E., Park M., Ji S., Park M., "A new approach to fuzzy modeling", *IEEE Transactions on Fuzzy Systems*, Vol. 5, No. 3, pp. 328-337, 1997.

Kim E., Park M., Kim S., Park M., "A transformed input-domain approach to fuzzy modeling", *IEEE Transactions on Fuzzy Systems*, Vol. 6, No. 4, pp. 596-604, 1998.

Kohonen T., *Self-organizing maps*, Springer, 1995.

Kohonen T., Oja E., Simula O., Visa A., Kangas J., "Engineering applications of the self-organizing maps", *Proceedings of IEEE*, pp. 1358-1384, ottobre, 1996.

Kononov E., *Visual Recurrence Analysis (VRA)*, disponibile su 'http://pw1.netcom.com/~eugenek', Versione 4.2, 15 novembre, 1999.

Kosko B., Dickerson J.A., "Function Approximation with Additive Fuzzy Systems", in *Theoretical Aspects of Fuzzy Control*, T. Nguyen, M. Sugeno, R. Tong e R. Yager editori, cap. 12, New York: Wiley, 1994.

Kosko B., "Fuzzy Systems as Universal Approximators", *IEEE Transactions on Computers*, Vol. 43, No. 11, pp. 1329-1333, 1994(bis).

Kosko B., Haykin S., "Special Issue on Intelligent Signal Processing", *Proceedings of IEEE*, novembre, 1998.

Kruskal J.B., "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis", *Psychometrika*, No. 29, pp. 1-27, 1964.

Kruskal J.B., "Nonmetric multidimensional scaling: a numerical method", *Psychometrika*, No. 29, pp. 115-129, 1964(bis).

Light W.A., "Some Aspects of Radial Basis Function Approximation", *Approximation Theory, Spline Functions and Applications* (S.P. Singh, ed.), NATO ASI Series, Vol. 256, pp. 163-190, Boston, MA: Kluwer Academic Publishers, 1992.

Martinelli G., *Fondamenti di Reti Neurali*, 2nd ed., EURoma, 2001.

Masulli F., Studer L., "Time Series Forecasting and Neural Networks", Tutorial in *Proc. of IJCNN'99*, Washington D.C., USA, 1999.

Mendel J.M., "Fuzzy Logic Systems for Engineering: A Tutorial", *Proceedings of IEEE*, Vol. 83, No. 3, pp. 345-387, marzo, 1995.

Panella M., Frattale Mascioli F.M., Rizzi A., Martinelli G., "Optimisation of Bayesian Classifiers by Using A Splitting Hierarchical EM Algorithm", *Proc. of Neural Computation (NC'2000)*, Berlino, Germania, 23-26 maggio, 2000.

Panella M., Rizzi A., Frattale Mascioli F.M., Martinelli G., "A Constructive EM Approach to Density Estimation for Learning", *Proc. of IJCNN2001*, Washington D.C., U.S.A., 14-19 luglio, 2001.

Panella M., Rizzi A., Frattale Mascioli F.M., Martinelli G., "Improved Time Series Forecasting by a Twofold Neural Predictor", *Proc. of EANN 2001*, Cagliari, 16-18 luglio, 2001(bis).

Panella M., Rizzi A., Frattale Mascioli F.M., Martinelli G., "ANFIS Synthesis by Hyperplane Clustering", *Proc. of IFSA/NAFIPS 2001*, Vancouver, Canada, 25-28 luglio, 2001(ter).

Papoulis A., *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, Inc., International Edition, 1991.

Park J., Sandberg I.W., "Universal Approximation using Radial-Basis-Function Networks", *Neural Computation*, 3, pp. 246-257, 1991.

Penrose R., *Ombre della Mente*, Rizzoli, 1996.

Poggio T., Girosi F., "Networks for Approximation and Learning", *Proceedings of the IEEE* 78, pp. 1481-1497, 1990.

Rizzi A., Frattale Mascioli F.M., Martinelli G., "Adaptive Resolution Min-Max Classifier", *Proc. of WCCI/FUZZ-IEEE '98*, Anchorage, Alaska, USA, pp. 1435-1440, 1998.

Rizzi A., Panella M., Frattale Mascioli F.M., Martinelli G., "A Recursive Algorithm for Fuzzy Min-Max Networks", *Proc. of IJCNN2000*, Como, 24-27 luglio, 2000.

Rizzi A., Panella M., Frattale Mascioli F.M., Martinelli G., "Automatic Training of Generalized Min-Max Classifiers", *Proc. of IFSA/NAFIPS 2001*, Vancouver, Canada, 25-28 luglio, 2001.

Rizzi A., Panella M., Frattale Mascioli F.M., "Adaptive Resolution Min-Max Classifiers", *IEEE Transactions on Neural Networks*, Vol. 13, No. 2, pp. 402-414, 2002.

Rojas I., Pomares H., Ortega J., Prieto A., "Self-Organized Fuzzy System Generation from Training Examples", *IEEE Transactions on Fuzzy Systems*, Vol. 8, No. 1, pp. 23-36, 2000.

Selim S.Z., Ismail M.A., "K-Means-type algorithms: a generalized convergence theorem and characterization of local optimality", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI 6, pp. 81-87, 1984.

Simpson P.K., "Fuzzy Min-Max Neural Networks–Part 1: Classification", *IEEE Transactions on Neural Networks*, Vol. 3, No. 5, pp. 776-786, 1992.

Simpson P.K., "Fuzzy Min-Max Neural Networks-Part 2: Clustering", *IEEE Transactions on Fuzzy Systems*, Vol. 1, pp. 32-45, 1993.

Takens F., "Detecting Strange Attractors in Turbolence", in *Dynamical Systems and Turbolence*, Warwick, 1980, Vol. 898 di *Lecture Notes in Mathematics*, D.A. Rand e L.S. Young, Eds. Spriger, pp. 366-381, 1981.

Tikhonov A.N., Arsenin V.Y., *Solutions of Ill-posed Problems*, Ed. W.H. Winston, 1977.

Wang L.-X., Mendel J.M., "Fuzzy basis functions, universal approximation and orthogonal least squares learning", *IEEE Transactions on Neural Networks*, Vol. 3, No. 5, pp. 807-814, settembre, 1992.

Wang L.-X., *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice Hall, Canada, 1994.

Zadeh L.A., "Fuzzy Sets", *Information and Control*, Vol. 8, pp. 338-353, 1965.

Zadeh L.A., "Probability Measures of Fuzzy Events", *Journal of Math. Anal. Appl.*, No. 10, pp. 421-427, 1968.

Zadeh L.A., "Fuzzy Logic and Soft Computing: Issues, Contentions and Perspectives", *3rd International Conference on Fuzzy Logic, Neural Nets and Soft Computing*, pp. 1-2, Iisuka, Giappone, 1994.