R. Bevilacqua O. Menchi

APPUNTI DI CALCOLO NUMERICO

Introduzione

Nell'analisi dei problemi del mondo reale la matematica svolge un ruolo determinante. In ogni disciplina scientifica e in ogni settore della tecnologia i modelli matematici che approssimano l'evolversi dell'evento oggetto di studio consentono di simulare, e quindi prevedere, lo sviluppo del fenomeno senza dover effettuare fisicamente esperimenti complessi, costosi e in alcuni casi anche pericolosi: nella progettazione di un velivolo la forma delle ali può essere successivamente adattata, senza dover costruire alcun prototipo, in base ai risultati delle simulazioni numeriche fatte con il calcolatore; lo studio dell'inquinamento ambientale, quale il propagarsi di una sostanza tossica nelle acque di un fiume, può essere condotto mediante calcolatore con un adeguato modello matematico, senza dover realizzare anche in forma ridotta, un reale pericoloso esperimento.

Il processo di risoluzione di un problema del mondo reale può essere così schematizzato: una prima fase (di modellizzazione) in cui al problema reale si associa un modello matematico che ne approssima l'evoluzione; una seconda fase in cui il modello matematico viene analizzato e da questo si ricavano proprietà qualitative della soluzione, come esistenza, unicità, regolarità; una terza fase in cui si individuano dei metodi di risoluzione e se ne analizza l'efficienza; infine una quarta fase in cui il metodo di risoluzione viene implementato su calcolatore mediante un adeguato linguaggio di programmazione.

Un aspetto importante in questo processo è che nella quasi totalità dei casi la soluzione del problema non è esprimibile in forma esplicita, ad esempio mediante funzioni elementari, e comunque le sole proprietà qualitative non sono sufficienti per gli scopi richiesti. Si pensi ad esempio alla traiettoria che deve descrivere una sonda spaziale per raggiungere un certo obiettivo: la funzione che la descrive è la soluzione di un opportuno sistema di equazioni differenziali; in questo caso è necessario conoscere le coordinate della navicella in un insieme stabilito di istanti temporali.

Si presenta quindi la necessità di risolvere algoritmicamente il problema matematico, cioè di ottenere, a partire da un insieme finito di numeri (rappresentati con un numero finito di cifre) con un numero finito di operazioni aritmetiche una informazione approssimata, ma adeguata alle richieste, della soluzione.

Nella risoluzione di problemi di analisi matematica che generalmente hanno natura continua questo processo algoritmico, necessariamente discreto poiché finito, si inquadra nel settore noto come *analisi numerica*.

L'analisi numerica ha assunto le caratteristiche di una disciplina autonoma solo con l'introduzione e l'uso dei calcolatori, quando l'elaborazione di grandi quantità di dati ha messo in luce nuovi problemi non emersi nel calcolo manuale e quando il dover risolvere in modo finito problemi di natura continua ha creato delle nuove problematiche. Fra queste, particolare rilevanza ha lo studio del condizionamento numerico di un problema, della stabilità numerica di un algoritmo, della complessità computazionale di un problema e le questioni legate alla discretizzazione di un problema continuo. L'introduzione e l'uso di calcolatori ha dato e sta dando un impulso notevole al settore dell'analisi numerica. Sono stati e vengono tuttora realizzati nuovi ed efficienti metodi numerici, che permettono di trattare problemi sempre più complessi che era impensabile trattare pochi anni fa.

In questi appunti sono esposti solo i più semplici metodi numerici per risolvere equazioni non lineari e sistemi lineari e per approssimare funzioni e integrali. Infatti lo scopo non è quello di fornire un manuale di metodi pronti per l'uso, ma di mettere in luce le problematiche tipiche del settore numerico, in particolare i problemi legati al condizionamento, alla stabilità numerica e al costo computazionale dei metodi. Gli algoritmi più importanti sono corredati di schemi semplificati di programmi in Matlab, che illustrano come si applicano le formule.

Per risolvere davvero un problema numerico conviene fare riferimento alle librerie di software che implementano i metodi numerici. I programmi di queste librerie soddisfano specifici criteri di qualità, in particolare per quanto riguarda l'efficienza e la robustezza: consentono quindi di scegliere il metodo più adatto e nello stesso tempo di prevedere ogni possibile fonte di errore. Attraverso INTERNET è possibile accedere a NETLIB che contiene una completa collezione di programmi numerici di dominio publico. La lista delle librerie disponibili si ottiene all'indirizzo

http://www.netlib.org//liblist.html

Di particolare interesse per gli argomenti trattati in questi appunti sono

LINPACK, ITPAK, EISPACK,

che contengono programmi per risolvere problemi lineari (rispettivamente risoluzione di sistemi lineari con metodi diretti, con metodi iterativi e calcolo di autovalori)

FUNPACK, MINPACK, QUADPACK,

che contengono programmi per calcolare valori di funzioni speciali, risolvere equazioni e sistemi non lineari, approssimare integrali di funzioni di una variabile.

Capitolo 1

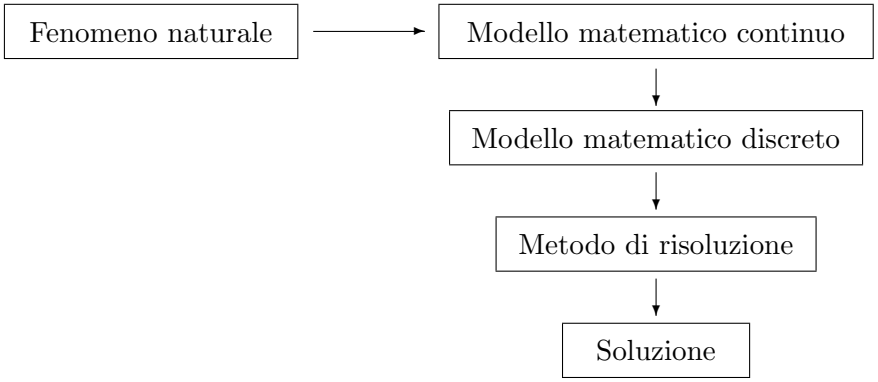
Analisi dell'errore

1.1 La generazione degli errori.

Per studiare un "fenomeno", sia esso naturale quale il moto di un pianeta o sia esso un prodotto umano quale il propagarsi dell'inquinamento atmosferico o la costruzione di un ponte, la matematica fornisce strumenti per realizzare ed analizzare dei modelli che descrivono, sia pure in modo approssimato, il fenomeno stesso. Ad esempio, nel modello matematico per la costruzione di un ponte, è richiesto il bilancio delle forze esercitate sui vari elementi del ponte stesso (peso degli elementi, peso degli automezzi in transito, azione del vento, ecc.). Non si tiene però conto delle forze gravitazionali esercitate sugli elementi del ponte dalla luna o dal sole; tali forze sono realmente esistenti e non nulle, e quindi dovrebbero essere considerate in un modello matematico esatto, ma possono di fatto essere trascurate in un modello che *approssima* il problema reale con errori contenuti.

Generalmente un modello matematico è un modello continuo (di solito un sistema di equazioni differenziali) che consente di individuare le proprietà qualitative della soluzione, come esistenza, unicità, regolarità, ecc., ma spesso non permette di determinare analiticamente la soluzione. È perciò necessario effettuare un'ulteriore approssimazione sostituendo il modello continuo con un modello discreto (*processo di discretizzazione*). Questa sequenza di approssimazioni è illustrata schematicamente nella figura.

In ogni passaggio da un blocco all'altro viene commesso un errore di approssimazione. L'errore introdotto nel processo di discretizzazione è detto *errore analitico*. È importante che gli errori commessi siano tutti dello stesso ordine: infatti ha scarsa utilità calcolare molto accuratamente la soluzione del modello discreto, quando questo approssima con scarsa precisione il modello continuo o quando il modello continuo è una grossolana descrizione del fenomeno naturale.



Gli errori che si producono durante la risoluzione sono principalmente dovuti al fatto che il calcolatore opera su dati numerici rappresentati per mezzo di una sequenza *finita* di cifre. Così numeri non razionali, come ad esempio π o $\sqrt{2}$, o numeri razionali con rappresentazione periodica, come ad esempio $1/3$, possono essere rappresentati in un calcolatore solo troncandone lo sviluppo e quindi commettendo un errore. Questo tipo di errore, che si produce già nell'immissione dei dati, prima ancora di eseguire effettivamente i calcoli, viene detto *errore di rappresentazione*.

La rappresentazione con un numero finito di cifre impone inoltre l'uso di un'aritmetica approssimata (*aritmetica finita*) che introduce errori anche nell'esecuzione delle operazioni aritmetiche.

Gli errori di rappresentazione dei dati e gli errori prodotti dalle singole operazioni si propagano nel corso dei calcoli e generano nella soluzione effettivamente calcolata degli errori che vengono chiamati rispettivamente *errore inerente* e *errore algoritmico*.

Il numero di cifre utilizzate nei calcolatori per rappresentare i dati numerici e per eseguire le operazioni è in generale abbastanza grande perché gli errori di rappresentazione e gli errori prodotti dalle singole operazioni siano piccoli. Si potrebbe quindi pensare che anche l'errore inerente e l'errore algoritmico siano piccoli, per cui non sia necessario valutare l'attendibilità dei risultati ottenuti. Purtroppo questo non è sempre vero, perché gli errori durante la propagazione possono essere fortemente amplificati. I seguenti esempi mostrano come la propagazione degli errori dovuti alla rappresentazione finita e all'uso di un'aritmetica finita possa alterare i risultati al di là di ogni ragionevole previsione. Gli esempi sono stati implementati in Matlab. I risultati calcolati vengono qui rappresentati in notazione esponenziale con 5 cifre significative.

Esempio. Sia p un numero reale non nullo. L'equazione di secondo grado

$$x^2 - qx + 1 = 0, \quad \text{dove} \quad q = p + 1/p,$$

ha le soluzioni $x_1 = p$ e $x_2 = 1/p$, che possono essere calcolate con la formula risolutiva

$$x_1 = \frac{q + \sqrt{q^2 - 4}}{2}, \quad x_2 = \frac{q - \sqrt{q^2 - 4}}{2}. \tag{1.1}$$

Si calcolano le soluzioni per valori di $p = 10^k$, con k che va da 2 a 8. Si ottiene così la seguente tabella, in cui accanto alle soluzioni $\widetilde{x}_1$ e $\widetilde{x}_2$ effetivamente calcolate viene dato anche l'errore relativo.

k	x_1	$\widetilde{x}_1$	e_1	x_2	$\widetilde{x}_2$	e_2
2	10^2	$1.0000\ 10^2$	0	10^{-2}	$1.0000\ 10^{-2}$	$1.9897\ 10^{-13}$
3	10^3	$1.0000\ 10^3$	0	10^{-3}	$1.0000\ 10^{-3}$	$2.3647\ 10^{-11}$
4	10^4	$1.0000\ 10^4$	0	10^{-4}	$1.0000\ 10^{-4}$	$2.0227\ 10^{-9}$
5	10^5	$1.0000\ 10^5$	0	10^{-5}	$1.0000\ 10^{-5}$	$3.3854\ 10^{-7}$
6	10^6	$1.0000\ 10^6$	0	10^{-6}	$1.0000\ 10^{-6}$	$7.6145\ 10^{-6}$
7	10^7	$1.0000\ 10^7$	0	10^{-7}	$9.9652\ 10^{-8}$	$3.4848\ 10^{-3}$
8	10^8	$1.0000\ 10^8$	0	10^{-8}	$1.4901\ 10^{-8}$	$4.9012\ 10^{-1}$

Mentre la soluzione calcolata $\widetilde{x}_1$ è corretta, la soluzione $\widetilde{x}_2$ ha un errore relativo crescente con p e per $p = 10^8$ solo la prima cifra del risultato fornito dal calcolatore è corretta. Per questa equazione, poiché il prodotto delle radici è uguale a 1, la soluzione x_2 può essere calcolata anche per mezzo della formula

$$x_2 = 1/x_1. \tag{1.2}$$

Applicando la (1.1) per calcolare x_1 e la (1.2) per calcolare x_2, si ottengono risultati affetti da errori trascurabili. □

Esempio. Proviamo ad utilizzare il calcolatore per avere delle indicazioni sul valore del limite

$$\lim_{x\to\infty} f(x), \quad f(x) = x(\sqrt{x^2+1} - x), \tag{1.3}$$

calcolando il valore di $f(x)$ per valori crescenti di x. Si osservi che la funzione $f(x)$ data in (1.3) può essere scritta anche nelle forme

$$f(x) = x\sqrt{x^2+1} - x^2, \tag{1.4}$$

$$f(x) = \frac{x}{\sqrt{x^2+1} + x}\,. \tag{1.5}$$

È quindi possibile calcolare $f(x)$ in questi tre modi diversi. Nella tabella che segue sono riportati i valori $\widetilde{f}_1$, $\widetilde{f}_2$ e $\widetilde{f}_3$ ottenuti usando rispettivamente le (1.3), (1.4) e (1.5) per x che va da 10^7 a $9\ 10^7$.

Le informazioni ottenute sono evidentemente discordanti: secondo i valori di $\widetilde{f}_1$ e di $\widetilde{f}_2$ si potrebbe dedurre che

$$\lim_{x\to\infty} f(x) = 0,$$

x	$\widetilde{f}_1$	$\widetilde{f}_2$	$\widetilde{f}_3$
$1.0000\ 10^7$	$5.0291\ 10^{-1}$	$5.0000\ 10^{-1}$	$5.0000\ 10^{-1}$
$2.0000\ 10^7$	$5.2154\ 10^{-1}$	$5.0000\ 10^{-1}$	$5.0000\ 10^{-1}$
$3.0000\ 10^7$	$4.4703\ 10^{-1}$	$5.0000\ 10^{-1}$	$5.0000\ 10^{-1}$
$4.0000\ 10^7$	$5.9605\ 10^{-1}$	$5.0000\ 10^{-1}$	$5.0000\ 10^{-1}$
$5.0000\ 10^7$	$3.7253\ 10^{-1}$	$5.0000\ 10^{-1}$	$5.0000\ 10^{-1}$
$6.0000\ 10^7$	$3.7253\ 10^{-1}$	$5.0000\ 10^{-1}$	$5.0000\ 10^{-1}$
$7.0000\ 10^7$	0	0	$5.0000\ 10^{-1}$
$8.0000\ 10^7$	0	0	$5.0000\ 10^{-1}$
$9.0000\ 10^7$	0	0	$5.0000\ 10^{-1}$

mentre secondo i valori di $\widetilde{f}_3$ si deduce il valore corretto del limite

$$\lim_{x\to\infty} f(x) = 0.5$$ □

Esempio. Assegnato un valore di x, per approssimare il valore di e^x si utilizza lo sviluppo in serie

$$\mathrm{e}^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots + \frac{x^n}{n!} + \ \ldots$$

opportunamente troncato. Si osserva che esiste un $\overline{n}$ per cui, sommando n termini della serie con $n \geq \overline{n}$, le prime cifre non si modificano più. Ad esempio, per $x = -20$ è

$$\mathrm{e}^{-20} = 1 - 20 + \frac{20^2}{2!} - \frac{20^3}{3!} + \frac{20^4}{4!} + \ldots + \frac{(-20)^n}{n!} + \ \ldots$$

e le prime 5 cifre delle approssimazioni calcolate sommando n termini si stabilizzano sul valore $6.1383\ 10^{-9}$ per $n \geq 75$. Calcolando per confronto il valore di e^{-20} con la funzione di libreria `exp` si ottiene il valore $2.0612\ 10^{-9}$. Una differenza così elevata indica che il valore da noi calcolato è inattendibile. È però possibile calcolare lo stesso valore sfruttando la proprietà che $\mathrm{e}^{-x} = 1/\mathrm{e}^x$ per cui è

$$\mathrm{e}^{-20} = \frac{1}{\mathrm{e}^{20}} = \frac{1}{1 + 20 + \frac{20^2}{2!} + \frac{20^3}{3!} + \frac{20^4}{4!} + \ldots + \frac{(20)^n}{n!} + \ \ldots}\,.$$

Operando in questo modo si ottiene in effetti $2.0612\ 10^{-9}$. □

Dagli esempi esaminati risulta che gli errori generati dalla rappresentazione con un numero finito di cifre e dall'uso di un'aritmetica finita, possono essere così elevati da togliere ogni validità ai risultati ottenuti ed è quindi molto importante riuscire a

valutarne e controllarne la propagazione. Si è visto anche che, utilizzando un procedimento di calcolo opportuno, è possibile contenere tali errori. Esistono quindi, per uno stesso problema, procedimenti di calcolo (*algoritmi*) che possono generare errori in misura diversa. In questi problemi elementari, dei quali si conosce la soluzione, è stato facile distinguere gli algoritmi *numericamente instabili*, ossia quelli per cui si presenta una elevata propagazione dell'errore, da quelli *numericamente stabili*. In generale nei problemi concreti tale distinzione non è facile ed è quindi molto importante disporre di tecniche per stabilire a priori se un dato algoritmo è numericamente stabile o instabile.

1.2 Problemi mal condizionati.

Esistono anche problemi tali che, qualunque algoritmo venga utilizzato per risolverli, l'errore generato nel risultato risulta elevato e talvolta tale da rendere privo di significato il risultato stesso. Questo fenomeno è una particolarità intrinseca del problema e non dipende dagli algoritmi utilizzati. Per questi problemi, detti *mal condizionati*, piccole variazioni nei dati inducono grosse variazioni nei risultati. Quindi in questo caso già la inevitabile perturbazione dei dati, dovuta alla rappresentazione finita in base, genera errori elevati nei risultati, qualunque sia l'algoritmo utilizzato. Vi sono problemi per i quali esiste la soluzione ed esiste un algoritmo per calcolarla, ma che sono praticamente irrisolubili perché fortemente mal condizionati, per cui i soli errori di rappresentazione dei dati generano sul risultato errori superiori al 100%.

Si consideri ad esempio il seguente sistema di equazioni lineari

$$\begin{cases} 1.01\,x + 1.02\,y = 2.03 \\ 0.99\,x + y = 1.99 \end{cases}$$

che ha la soluzione $x = 1,\ y = 1$. Se ne altera il coefficiente della x nella seconda equazione dell'1%. Il nuovo sistema perturbato

$$\begin{cases} 1.01\,x + 1.02\,y = 2.03 \\ x + y = 1.99 \end{cases}$$

ha soluzione $\widetilde{x} = -0.02$, $\widetilde{y} = 2.01$. La soluzione del sistema perturbato presenta, rispetto alla soluzione del sistema non perturbato, una variazione maggiore del 100% sia nella x che nella y.

La natura del mal condizionamento del problema precedente può essere descritta mediante un'interpretazione geometrica. Nella figura 1.1 le due equazioni del sistema corrispondono alle due rette tracciate con linea continua e la loro intersezione ha per componenti la soluzione del sistema. Poiché le due rette formano tra loro un angolo molto piccolo, perturbando la seconda equazione, si altera di poco la posizione della seconda retta (ottenendo la retta tratteggiata), ma cambia molto la posizione del punto intersezione e quindi le coordinate di quest'ultimo, che sono le soluzioni del sistema perturbato.

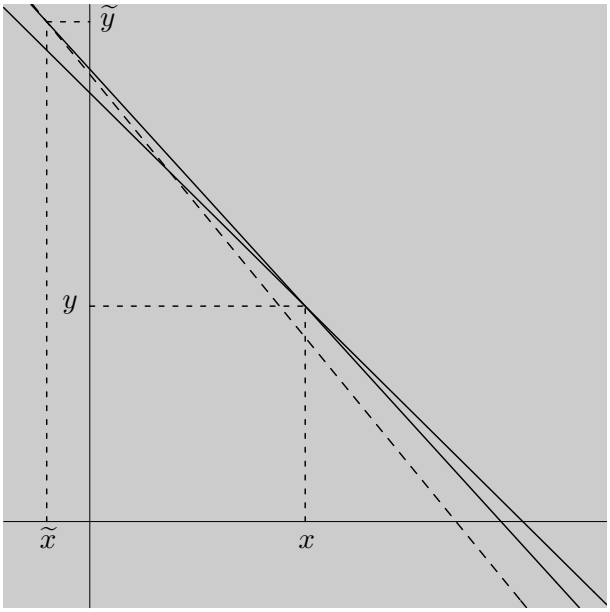


Figura 1.1: Interpretazione geometrica del mal condizionamento.

Nel caso di mal condizionamento, per trattare il problema si potrebbe pensare di aumentare la precisione della rappresentazione dei dati, aumentando il numero di cifre, in modo da ridurre l'errore di rappresentazione. Però l'uso di un numero maggiore di cifre per ridurre la propagazione degli errori comporterebbe un impiego maggiore delle risorse *quantità di memoria* e *tempo di elaborazione*. Una scelta accorta di un metodo per la risoluzione di un problema consente una migliore utilizzazione delle risorse tempo di elaborazione e quantità di memoria, permettendo così di trattare anche problemi di elevate dimensioni.

1.3 Complessità computazionale.

In molti casi per risolvere uno stesso problema sono disponibili diversi algoritmi.

Esempio. Il calcolo del valore del polinomio di grado n

$$p(x) = \sum_{i=0}^{n} a_i x^i$$

in un punto assegnato x può essere effettuato calcolando separatamente le potenze x^i, mediante il seguente algoritmo

$$\begin{aligned} &p_0 = a_0, \quad y_0 = 1 \\ &y_i = y_{i-1}x, \quad p_i = a_i y_i + p_{i-1}, \quad i = 1, 2, \ldots, n, \\ &p(x) = p_n, \end{aligned}$$

che richiede $2n-1$ moltiplicazioni ed n addizioni. Poiché il polinomio $p(x)$ può anche essere rappresentato nel modo seguente

$$p(x) = (\dots((a_n x + a_{n-1})x + a_{n-2})x + \dots + a_1)x + a_0,$$

si può ottenere un altro algoritmo per il calcolo di $p(x)$, detto *metodo di Ruffini-Horner* o della *divisione sintetica*:

$$\begin{aligned} &p_0 = a_n, \\ &p_i = p_{i-1}x + a_{n-i}, \quad i = 1, 2, \dots, n, \\ &p(x) = p_n, \end{aligned}$$

che impiega n moltiplicazioni ed n addizioni, cioè il numero di moltiplicazioni è dimezzato rispetto al metodo precedente.

Esempio. Per calcolare la soluzione del sistema lineare

$$A\boldsymbol{x} = \boldsymbol{b},$$

dove A è una matrice non singolare di ordine n, si può applicare la regola di Cramer il cui costo, in termini di operazioni moltiplicative, è circa $(n+1)!$. Se invece il sistema viene risolto col metodo di sostituzione (metodo di eliminazione di Gauss) che studieremo più avanti, il numero di moltiplicazioni richieste è circa $n^3/3$.

Se per semplicità il tempo di esecuzione viene stimato moltiplicando il tempo richiesto da una singola moltiplicazione per il numero di moltiplicazioni, nell'ipotesi che una moltiplicazione venga eseguita in 10^{-7} secondi, per la risoluzione di un sistema di n equazioni ed n incognite con i due metodi in esame si avrebbero i seguenti tempi

n	metodo di Cramer	metodo di sostituzione
10	4 secondi	$3.3\ 10^{-5}$ secondi
12	10 minuti	$5.8\ 10^{-5}$ secondi
14	36 ore	$9.1\ 10^{-5}$ secondi
16	412 giorni	$1.3\ 10^{-4}$ secondi
18	386 anni	$1.9\ 10^{-4}$ secondi

Se si utilizza il metodo di Cramer, il problema della risoluzione di un sistema di equazioni lineari non risulta trattabile già per valori piccoli di n.

La possibilità di risolvere sistemi di grosse dimensioni, come quelli che derivano da problemi concreti, non dipende solo dalla disponibilità di calcolatori potenti (anche con tali calcolatori il metodo di Cramer richiederebbe tempi inaccettabili di elaborazione) ma dall'uso di metodi computazionalmente efficienti. È quindi con lo sviluppo di algoritmi efficienti e la realizzazione del relativo *software*, oltre che con lo sviluppo del *hardware*, che è possibile trattare problemi di dimensioni sempre più elevate. Il metodo di Gauss permette di ridurre il costo computazionale della risoluzione del problema da $(n+1)!$ a circa $n^3/3$ moltiplicazioni. È naturale chiedersi

se tale metodo sia ottimo, ossia se non esistano algoritmi che richiedano un numero inferiore di moltiplicazioni (l'ottimalità è stata dimostrata relativamente ai metodi che usano solo combinazioni di righe e di colonne). Tale tipo di problema fa parte di una classe più ampia, oggetto di studio del settore della *complessità computazionale* che ha avuto recentemente consistenti sviluppi. Nel caso della risoluzione di un sistema di n equazioni lineari in n incognite sono stati individuati metodi che richiedono non più di kn^θ operazioni aritmetiche, dove k è una costante positiva e $\theta < 3$ (il più piccolo valore noto di θ è $2.38\ldots$).

1.4 Rappresentazione in base di un numero.

L'insieme dei numeri rappresentabili con un calcolatore è finito: con n cifre che possono assumere i valori binari 0 e 1, il massimo numero di configurazioni diverse ottenibili è 2^n. Ad esempio, con $n = 3$ si hanno le otto configurazioni seguenti:

$$000,\ 001,\ 010,\ 011,\ 100,\ 101,\ 110,\ 111.$$

Un problema fondamentale è quello di associare ad ogni configurazione di cifre binarie un numero reale in modo che l'insieme dei numeri rappresentati sia opportunamente distribuito nell'intervallo della retta reale a cui appartengono i numeri da rappresentare e in modo che l'errore che si commette risulti sufficientemente contenuto.

Per affrontare tale problema è opportuno fare riferimento al sistema di rappresentazione in base dei numeri reali. Sia $\beta \geq 2$ il numero *intero* che si assume come *base della rappresentazione*. Vale il seguente teorema.

Teorema di rappresentazione in base. *Sia x un numero reale non nullo; allora esistono e sono unici il numero intero p e la successione $\{d_i\}_{i=1,2,\ldots}$ di numeri interi, $0 \leq d_i < \beta$, $d_1 \neq 0$, non definitivamente uguali a $\beta - 1$, tali che*

$$x = sgn(x)\beta^p \sum_{i=1}^{\infty} d_i \beta^{-i}, \tag{1.6}$$

dove $sgn(x) = 1$ se $x > 0$, $sgn(x) = -1$ se $x < 0$. □

La (1.6) viene detta *rappresentazione in base* del numero x. Le quantità che compaiono nella (1.6) vengono dette

$$\begin{array}{ll} p & esponente, \\ d_i & cifre\ della\ rappresentazione, \\ \sum\limits_{i=1}^{\infty} d_i\beta^{-i} & mantissa. \end{array}$$

Se esiste un indice k tale che $d_k \neq 0$ e $d_i = 0$ per $i > k$, la rappresentazione in base β si dice *finita di lunghezza k*.

Per la rappresentazione (1.6) in base β si usa la notazione posizionale

$$x = \pm\, (0.d_1 d_2 \ldots)_\beta\ \beta^p \quad \text{o} \quad x = \pm\, 0.d_1 d_2 \ldots\ \beta^p.$$

La base β, quando è chiara dal contesto, non viene indicata. Anche lo zero prima del punto può essere omesso. Nella scrittura corrente, in cui $\beta = 10$, si scrive

$$x = \pm\, 0.d_1 d_2 \ldots\ 10^p \quad \text{e} \quad x = \pm\, 0.d_1 d_2 \ldots \text{ se } p = 0.$$

Poiché $d_1 \neq 0$, la rappresentazione (1.6) è *normalizzata*. La normalizzazione, oltre ad essere necessaria per l'unicità, si rivela vantaggiosa quando si rappresenta solo un numero limitato di cifre. Ad esempio il numero $x = 1/7000$ può essere così rappresentato in base 10:

$$\begin{array}{ll} x = 0.142857142857\ldots\ 10^{-3} & \text{(rappresentazione normalizzata),} \\ x = 0.000142857142\ldots & \text{(rappresentazione non normalizzata).} \end{array}$$

Se si tronca la rappresentazione a 6 cifre, si ottiene nel primo caso il numero $x_1 = 0.142857\ 10^{-3}$ e, nel secondo caso, il numero $x_2 = 0.000142$. È evidente come x_1 fornisca una migliore approssimazione, rispetto a x_2, del valore di x. L'informazione fornita dagli zeri dopo il punto può essere trasferita, con minor spreco di memoria, nell'esponente.

1.5 Conversione di base.

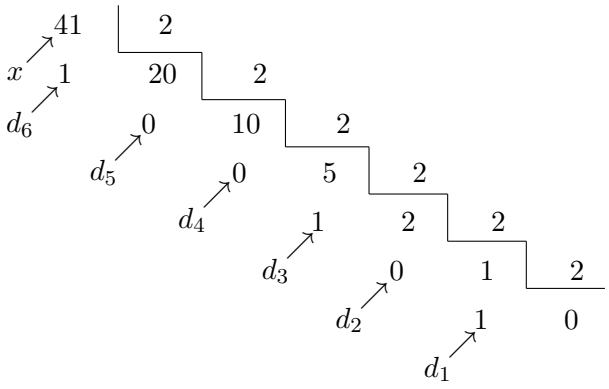
Dato un numero $x > 0$ scritto in base 10, l'operazione con cui si ricavano le cifre della sua rappresentazione in base β si chiama *conversione di base*. La conversione viene fatta seguendo algoritmi diversi, a seconda che x sia un numero intero o no.

(a) Se x è intero, allora

$$x = \beta^p \sum_{i=1}^{p} d_i \beta^{-i} = \sum_{i=1}^{p} d_i \beta^{p-i} = \beta\ q_1 + d_p, \quad \text{dove} \quad q_1 = \sum_{i=1}^{p-1} d_i \beta^{(p-1)-i}.$$

Poiché q_1 e d_p sono due interi e $0 \leq d_p < \beta$, q_1 è il quoziente della divisione di x per β e d_p è il resto. Si è così ricavata l'ultima cifra della rappresentazione in base β di x. Riapplicando il ragionamento a q_1 si ricava d_{p-1} e così via.

Per esempio, si converte in base 2 il numero intero $x = 41$ (scritto nella base 10).



La successione dei resti fornisce le cifre della rappresentazione in base 2 di x a partire da quella più a destra. Quindi

$$x = 101001_2 = (0.101001)_2\ 2^6.$$

Il procedimento di conversione viene eseguito dal seguente algoritmo scritto in Matlab.

```
i=0;
while x>0
  i=i+1;
  c(i)=rem(x,beta);
  x=floor(x/beta);
end;
p=i;
d=c(p:-1:1);
```

La funzione predefinita `rem` dà il resto della divisione fra interi; la funzione `floor` dà la parte intera inferiore di un numero reale e quindi, nel nostro caso, il quoziente intero della divisione. Applicando l'algoritmo alla conversione del numero `x` $= 79$ in base 2 si ottiene `p` $= 7$ e `d` $= [\,1\ 0\ 0\ 1\ 1\ 1\ 1\,]$.

Si noti che la rappresentazione di un numero intero è finita qualunque sia la base.

(b) Esaminiamo poi il caso che $0 < x < 1$. Come prima cosa dobbiamo determinare l'esponente p, che è minore o uguale a 0. Si considera la successione

$$y_1 = \beta\ x, \quad y_2 = \beta^2\ x, \quad \ldots, \quad y_j = \beta^j\ x,$$

arrestandosi al primo indice j per cui $y_j \geq 1$. È

$$p = 1 - j \quad \text{e} \quad y_j = \beta \sum_{i=1}^{\infty} d_i \beta^{-i} = d_1 + y_{j+1}, \quad \text{dove} \quad y_{j+1} = \sum_{i=2}^{\infty} d_i \beta^{-(i-1)}.$$

Poiché $y_{j+1} < 1$, d_1 risulta essere la parte intera di y_j. Si è così trovata la prima cifra della rappresentazione in base β di x. Si riapplica il ragionamento a y_{j+1}, ottenendo

$$\beta y_{j+1} = d_2 + y_{j+2}, \quad \text{dove} \quad y_{j+2} = \sum_{i=3}^{\infty} d_i \beta^{-(i-2)},$$

e così via.

Per esempio, si converte in base 2 il numero $x = 0.4$ (scritto nella base 10).

$$\begin{array}{lcll} 2 \times 0.4 = 0.8 & & & \\ & \searrow & & \\ 2 \times 0.8 = 1.6 & & & d_1 = 1 \\ & \searrow & & \\ 2 \times 0.6 = 1.2 & & & d_2 = 1 \\ & \searrow & & \\ 2 \times 0.2 = 0.4 & & & d_3 = 0 \\ & \searrow & & \\ 2 \times 0.4 = 0.8 & & & d_4 = 0 \end{array}$$

e così via. È $y_2 = 1.6$, quindi $p = -1$. Le successive moltiplicazioni per 2 forniscono le cifre $d_1 d_2 d_3 d_4$. Poiché risulta $y_1 = y_5 = 0.8$, le cifre successive si ripetono periodicamente:

$$x = (0.110011001100\ldots)_2\ 2^{-1}.$$

Quindi il numero 0.4 che in base 10 ha una rappresentazione finita, in base 2 ha una rappresentazione periodica.

Il seguente algoritmo determina le prime k cifre in base β di x, con $0 < x < 1$.

```
% determinazione dell'esponente
p=1;
while floor(x)==0
  x=beta*x;
  p=p-1;
end;
% determinazione delle cifre
i=0;
while i<k
  i=i+1;
  d(i)=floor(x);
  x=beta*(x-d(i));
end
```

Applicando l'algoritmo alla conversione del numero `x` $= 0.1$ in base 2 e calcolando le prime `k` $= 10$ cifre, si ottiene `p` $= -3$ e `d` $= [\,1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\,]$.

Se la rappresentazione di x è finita di lunghezza minore di k, le ultime cifre di x risultano nulle.

(c) Infine se $x > 1$ non è intero, si scrive $x = x_1 + x_2$, dove $x_1 = \lfloor x \rfloor$ è intero e $0 < x_2 < 1$. I due numeri x_1 e x_2 vengono convertiti separatamente, ottenendo le rappresentazioni:

$$x_1 = \beta^p \sum_{i=1}^{p} d_i \beta^{-i} \quad \text{e} \quad x_2 = \beta^q \sum_{i=1}^{\infty} c_i \beta^{-i}.$$

Si ha allora:

$$x = \beta^p \sum_{i=1}^{\infty} e_i \beta^{-i},$$

dove

$$e_i = \begin{cases} d_i & \text{per } i = 1, \ldots, p, \\ 0 & \text{per } i = p+1, \ldots, p-q, \quad \text{se } p+1 \le p-q, \\ c_{i-p+q} & \text{per } i \ge p-q+1. \end{cases}$$

Per esempio, per convertire in base 2 il numero $x = \dfrac{3270}{7}$, si pone $x = x_1 + x_2$, dove $x_1 = 467$, $x_2 = \dfrac{1}{7}$, e si convertono separatamente x_1 e x_2. Si ottiene

$$x_1 = (0.111010011)_2\ 2^9, \qquad x_2 = (0.100100100\ldots)_2\ 2^{-2},$$

da cui

$$x = (0.111010011001001001\ldots)_2\ 2^9.$$

1.6 Numeri di macchina.

La rappresentazione in base dei numeri è il punto di partenza per definire nel modo migliore il sottoinsieme (finito) dell'insieme dei numeri reali che possiamo rappresentare in un calcolatore.

Siano $t,\ m,\ M$, numeri interi tali che $t \ge 1$, $m,\ M > 0$. Si definisce insieme dei *numeri di macchina* in base β con t *cifre significative*, l'insieme

$$\mathcal{F}_{(\beta,t,m,M)} = \{0\} \cup \{x \in \mathbb{R}\ :\ x = \mathrm{sgn}(x)\beta^p \sum_{i=1}^{t} d_i \beta^{-i},$$
$$\text{con } 0 \le d_i < \beta \text{ per } i = 1, 2, \ldots, t,\ d_1 \ne 0,\ -m \le p \le M\}.$$

Tutti i numeri di macchina, eccetto lo zero, hanno una rappresentazione normalizzata. Si usa dire che i numeri di $\mathcal{F}_{(\beta,t,m,M)}$ sono rappresentati *in virgola mobile* (dall'inglese *floating point*).

Il minimo numero positivo di $\mathcal{F}_{(\beta,t,m,M)}$ è

$$\omega = 0.1\ \beta^{-m} = \beta^{-m-1},$$

il massimo è

$$\Omega = \beta^M(\beta - 1)\sum_{i=1}^{t}\beta^{-i} = \beta^M(1 - \beta^{-t}).$$

Poiché con t cifre in base β si possono formare β^t numeri diversi, e fra questi vanno esclusi quelli la cui prima cifra è nulla, l'insieme $\mathcal{F}_{(\beta,t,m,M)}$, oltre allo zero, contiene $(m + M + 1)(\beta^t - \beta^{t-1})$ numeri positivi compresi fra ω e Ω e altrettanti numeri negativi compresi fra $-\Omega$ e $-\omega$.

Ad esempio, nella figura 1.2 sono riportati i numeri di $\mathcal{F}_{(2,3,1,1)}$. I numeri positivi sono compresi fra $\frac{1}{4}$ e $\frac{7}{4}$, quelli negativi fra $-\frac{7}{4}$ e $-\frac{1}{4}$.

Figura 1.2: I numeri di $\mathcal{F}_{(2,3,1,1)}$.

Per quanto riguarda l'effettiva rappresentazione dei numeri di macchina su calcolatore, nel corso degli anni si sono succeduti diversi sistemi, sviluppati dalle diverse ditte costruttrici. Sono state usate varie basi, sia la base 10 che la base 2 e varie potenze di 2, e molte diverse precisioni. Questa completa anarchia, la cui conseguenza più immediata era la totale incertezza su come uno stesso programma si sarebbe comportato su macchine diverse, ha avuto termine con la creazione dello standard IEEE per la rappresentazione dei numeri floating point (vedere par. 1.14).

Se il numero reale non nullo $x = \pm 0.d_1d_2\ldots\ \beta^p$ è tale che $-m \le p \le M$, $d_1 \neq 0$ e $d_i = 0$, per $i > t$, allora $x \in \mathcal{F}_{(\beta,t,m,M)}$. Se x non appartiene a $\mathcal{F}_{(\beta,t,m,M)}$, si pone il problema di associare, in modo adeguato, ad x un numero di macchina $\widetilde{x}$, cioè di *approssimare* x o, come si usa dire, di *rappresentare* x con $\widetilde{x}$. Nel seguito si suppone per semplicità che $x > 0$ (risultati analoghi si ottengono per $x < 0$), e che la base β sia un numero pari.

Si possono presentare i seguenti casi:

a) L'esponente p non appartiene all'intervallo $[-m, M]$. Se $p < -m$, si verifica la situazione di *underflow*, se $p > M$, si verifica la situazione di *overflow*. Il comportamento del sistema di calcolo può variare: vi può essere o meno una segnalazione, il calcolo può essere interrotto o può continuare con l'assegnazione di valori predefiniti (per esempio ω e Ω).

b) L'esponente p appartiene all'intervallo $[-m, M]$ ma x non appartiene a $\mathcal{F}_{(\beta,t,m,M)}$ perché le sue cifre d_i, $i > t$, non sono tutte nulle. In questo caso x viene rappresentato

con uno dei due seguenti numeri di macchina:

$$\mathrm{trn}(x) = \beta^p \sum_{i=1}^{t} d_i \beta^{-i},$$

ottenuto per *troncamento* di x alla t-esima cifra, e

$$\mathrm{arr}(x) = \beta^p \ \mathrm{trn}\ \Big(\sum_{i=1}^{t+1} d_i \beta^{-i} + \frac{1}{2}\beta^{-t} \Big),$$

ottenuto per *arrotondamento* di x alla t-esima cifra. Quindi è

$$\mathrm{arr}(x) = \begin{cases} \mathrm{trn}(x) & \text{se } d_{t+1} < \beta/2, \\ \mathrm{trn}(x) + \beta^{p-t} & \text{se } d_{t+1} \geq \beta/2. \end{cases}$$

Da notare che nell'effettuare l'arrotondamento di un numero si può verificare overflow. Infatti, se

$$x = 0.d_1 d_2 \ldots d_{t+1} \ldots \ \beta^M, \quad \text{con} \quad d_i = \beta - 1,\ i = 1, 2, \ldots, t \text{ e } d_{t+1} \geq \frac{\beta}{2},$$

risulta $\mathrm{arr}(x) = 0.10\ldots0\ \beta^{M+1}$ e questo numero non appartiene a $\mathcal{F}_{(\beta,t,m,M)}$.

1.7 Errori di rappresentazione.

Per valutare l'errore commesso nel rappresentare un numero reale $x > 0$ con un numero di macchina $\widetilde{x}$, si considerano le seguenti quantità

$$\widetilde{x} - x \quad \textit{errore assoluto},$$

$$\epsilon_x = \frac{\widetilde{x} - x}{x} \quad \textit{errore relativo}.$$

ϵ_x viene detto *errore di rappresentazione* di x.

L'insieme dei numeri di macchina gode dell'importante proprietà di fornire approssimazioni con errore relativo uniformemente limitato da una costante. Valgono infatti i seguenti teoremi.

Teorema. *Se non si verificano situazioni di underflow o di overflow risulta*

$$|trn(x) - x| < \beta^{p-t}, \tag{1.7}$$

$$|arr(x) - x| \leq \frac{1}{2}\,\beta^{p-t}, \tag{1.8}$$

dove il segno di uguaglianza vale se e solo se $d_{t+1} = \beta/2$ *e* $d_{t+i} = 0$ *per* $i \geq 2$.

Dim. Se x coincide con Ω, allora $\mathrm{trn}(x) - x = \mathrm{arr}(x) - x = 0$. Altrimenti siano a e b i due numeri di macchina consecutivi

$$a = \beta^p \sum_{i=1}^{t} d_i \beta^{-i}, \quad b = \beta^p \Big(\sum_{i=1}^{t} d_i \beta^{-i} + \beta^{-t} \Big).$$

È $a \le x < b$ e risulta

$$\mathrm{trn}(x) = a, \qquad x - \mathrm{trn}(x) < b - a = \beta^{p-t},$$

da cui segue la (1.7). Risulta poi

$$\mathrm{arr}(x) = \begin{cases} a & \text{se } x < \dfrac{a+b}{2}\,, \\ b & \text{se } x \ge \dfrac{a+b}{2}\,, \end{cases}$$

e quindi

$$|\mathrm{arr}(x) - x| \le \frac{b-a}{2} = \frac{1}{2}\,\beta^{p-t},$$

e l'uguaglianza vale se e solo se

$$x = \frac{a+b}{2}\,, \quad \text{cioè } d_{t+1} = \frac{\beta}{2} \text{ e } d_{t+i} = 0,\ i \ge 2. \qquad \square$$

Teorema. *Se non si verificano situazioni di underflow o di overflow risulta*

$$\left| \frac{\widetilde{x} - x}{x} \right| < u, \quad \textit{dove} \quad u = \begin{cases} \beta^{1-t} & \textit{se } \widetilde{x} = trn(x), \\ \dfrac{1}{2}\,\beta^{1-t} & \textit{se } \widetilde{x} = arr(x). \end{cases} \tag{1.9}$$

La quantità u definita nella (1.9) è detta *precisione di macchina.*

Dim. Poiché $d_1 \neq 0$, si ha

$$x \ge \beta^p d_1 \beta^{-1} \ge \beta^{p-1},$$

e quindi

$$\left| \frac{\widetilde{x} - x}{x} \right| \le \frac{|\widetilde{x} - x|}{\beta^{p-1}}\,.$$

Per il troncamento dalla (1.7) segue che

$$\left| \frac{\mathrm{trn}(x) - x}{x} \right| < \frac{\beta^{p-t}}{\beta^{p-1}} = \beta^{-t+1}.$$

Analogamente per l'arrotondamento, quando nella (1.8) vale il segno di $<$ stretto, si ha

$$\left| \frac{\mathrm{arr}(x) - x}{x} \right| < \frac{1}{2}\,\beta^{-t+1}.$$

Se nella (1.8) vale il segno di uguaglianza, cioè $d_{t+1} = \dfrac{\beta}{2}$ e $d_{t+i} = 0$ per $i \geq 2$, si ha

$$x \geq \beta^p (d_1 \beta^{-1} + d_{t+1} \beta^{-t-1}) > \beta^{p-1},$$

e quindi

$$\left| \frac{\mathrm{arr}(x) - x}{x} \right| = \frac{1}{2} \, \frac{\beta^{p-t}}{x} < \frac{1}{2} \, \beta^{-t+1}. \qquad \square$$

La rappresentazione con arrotondamento è in generale più accurata. Per questo, anche se richiede un costo maggiore, essa viene di solito utilizzata nella conversione dei dati in ingresso-uscita, normalmente realizzata al livello del software.

Dalla (1.9) segue che la rappresentazione $\widetilde{x}$ di x per troncamento o per arrotondamento verifica

$$\widetilde{x} = x(1 + \epsilon_x), \quad \text{con } |\epsilon_x| < u. \tag{1.10}$$

Si può vedere che la precisione di macchina è il minimo numero di macchina che aggiunto a 1 dà un risultato la cui rappresentazione è maggiore di 1.

1.8 Operazioni di macchina.

La somma di due numeri di macchina $x,\ y \in \mathcal{F}_{(\beta,t,m,M)}$ è un numero che può non appartenere all'insieme $\mathcal{F}_{(\beta,t,m,M)}$. Ad esempio se $\beta = 10$ e $t = 2$, la somma dei due numeri $x = 0.11\ 10^0$, $y = 0.11\ 10^{-2}$, è $x + y = 0.1111\ 10^0$ che non appartiene ad $\mathcal{F}_{(10,2,m,M)}$. Questo può accadere anche con le altre operazioni aritmetiche.

Si presenta dunque il problema di approssimare, nel modo più conveniente possibile, il risultato di un'operazione aritmetica fra due numeri di macchina con un numero di macchina. Occorre perciò individuare le "operazioni aritmetiche" su $\mathcal{F}_{(\beta,t,m,M)}$, ossia definire un'*aritmetica di macchina*, detta anche *aritmetica approssimata* o *aritmetica finita*, che meglio approssimi l'*aritmetica esatta* dei numeri reali.

Indicando con $\circledcirc$ l'operazione di macchina che approssima l'operazione esatta $\circ$, per tutti i numeri di macchina x e y per cui l'operazione non dia luogo a situazioni di underflow o di overflow, deve valere una relazione analoga alla (1.10):

$$x \circledcirc y = (x \circ y)(1 + \epsilon), \quad |\epsilon| < u, \tag{1.11}$$

dove u è la precisione di macchina.

Le operazioni di macchina che soddisfano la (1.11) sono dette *operazioni in virgola mobile* e l'aritmetica associata è detta *aritmetica in virgola mobile*, l'errore relativo ϵ commesso è detto *errore locale* dell'operazione.

Le operazioni:

$$x \circledcirc y = \mathrm{trn}(x \circ y) \quad \text{e} \quad x \circledcirc y = \mathrm{arr}(x \circ y)$$

sono operazioni di macchina che soddisfano la limitazione (1.11).

L'aritmetica di macchina con l'arrotondamento approssima l'aritmetica dei numeri reali meglio dell'aritmetica di macchina con il troncamento, ma la sua implementazione richiede l'uso di *registri* più lunghi e un maggior tempo di macchina per l'esecuzione delle operazioni. Ad esempio, gli algoritmi che realizzano l'addizione e la moltiplicazione di macchina con arrotondamento del risultato richiedono un registro di lunghezza $t+2$, mentre basta un registro di lunghezza $t+1$ per calcolare una somma o un prodotto che soddisfano la (1.11) con la precisione del troncamento.

Non tutte le proprietà algebriche delle operazioni nel campo reale sono soddisfatte dalle operazioni di macchina. Ad esempio non valgono le seguenti proprietà:

a) associatività dell'addizione: $(x+y)+z = x+(y+z)$;

b) associatività della moltiplicazione: $(xy)z = x(yz)$;

c) legge di cancellazione: se $xy = yz,\ y \neq 0$ allora $x = z$;

d) distributività: $x(y+z) = xy + xz$;

e) semplificazione: $x(y/x) = y$;

come dimostrano i seguenti esempi in $\mathcal{F}_{(10,2,m,M)}$ e aritmetica con arrotondamento:

a) posto $x = 0.11\ 10^0,\ y = 0.13\ 10^{-1},\ z = 0.14\ 10^{-1}$, risulta

$$(x \oplus y) \oplus z = 0.12\ 10^0 \oplus z = 0.13\ 10^0,$$
$$x \oplus (y \oplus z) = x \oplus 0.27\ 10^{-1} = 0.14\ 10^0;$$

b) posto $x = 0.11\ 10^1,\ y = 0.31\ 10^1,\ z = 0.25\ 10^1$, risulta

$$(x \otimes y) \otimes z = 0.34\ 10^1 \otimes z = 0.85\ 10^1,$$
$$x \otimes (y \otimes z) = x \otimes 0.78\ 10^1 = 0.86\ 10^1;$$

c) posto $x = 0.51\ 10^1,\ y = 0.22\ 10^1,\ z = 0.52\ 10^1$, risulta

$$x \otimes y = 0.11\ 10^2, \qquad z \otimes y = 0.11\ 10^2,$$

e quindi $x \otimes y = z \otimes y,\ y \neq 0,\ x \neq z$;

d) posto $x = 0.11\ 10^1,\ y = 0.23\ 10^1,\ z = 0.24\ 10^1$, risulta

$$(x \otimes y) \oplus (x \otimes z) = 0.25\ 10^1 \oplus 0.26\ 10^1 = 0.51\ 10^1,$$
$$x \otimes (y \oplus z) = x \otimes 0.47\ 10^1 = 0.52\ 10^1;$$

e) posto $x = 0.70\ 10^1,\ y = 0.80\ 10^1$, si ha

$$x \otimes (y \oslash x) = x \otimes 0.11\ 10^1 = 0.77\ 10^1 \neq y.$$

1.9 Errore nel calcolo di una funzione razionale.

Sia $f : \mathbb{R}^n \to \mathbb{R}$ una funzione razionale, cioè una funzione il cui valore $f(\boldsymbol{x})$ assunto in corrispondenza a $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ è ottenuto mediante un numero finito di operazioni aritmetiche su $x_1, x_2, \ldots, x_n$. Il valore effettivamente calcolato risulta affetto dagli errori indotti da due diversi fenomeni:

a) errore generato dalla rappresentazione dei dati $x_1, x_2, \ldots, x_n$ come numeri di macchina (*errore inerente*);

b) errore generato dal fatto che le operazioni sono effettuate in aritmetica finita (*errore algoritmico*).

Se $\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_n$ sono le rappresentazioni come numeri di macchina dei dati $x_1, x_2, \ldots, x_n$, e $\psi(\boldsymbol{x})$ è la funzione effettivamente calcolata, il valore che si ottiene al posto di $f(\boldsymbol{x})$ è allora $\psi(\widetilde{\boldsymbol{x}})$, dove $\widetilde{\boldsymbol{x}} = (\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_n)$.

Se $f(\boldsymbol{x}) \neq 0$ e $f(\widetilde{\boldsymbol{x}}) \neq 0$, si definiscono i seguenti errori:

errore totale di $\psi(\widetilde{\boldsymbol{x}})$ rispetto a $f(\boldsymbol{x})$ $\quad \epsilon_{tot} = \dfrac{\psi(\widetilde{\boldsymbol{x}}) - f(\boldsymbol{x})}{f(\boldsymbol{x})}$,

errore inerente $\quad \epsilon_{in} = \dfrac{f(\widetilde{\boldsymbol{x}}) - f(\boldsymbol{x})}{f(\boldsymbol{x})}$,

errore algoritmico $\quad \epsilon_{alg} = \dfrac{\psi(\widetilde{\boldsymbol{x}}) - f(\widetilde{\boldsymbol{x}})}{f(\widetilde{\boldsymbol{x}})}$,

e si ha

$$\epsilon_{tot} = \frac{\psi(\widetilde{\boldsymbol{x}}) - f(\boldsymbol{x})}{f(\boldsymbol{x})} = \frac{\psi(\widetilde{\boldsymbol{x}})}{f(\boldsymbol{x})} - 1 = \frac{\psi(\widetilde{\boldsymbol{x}})}{f(\widetilde{\boldsymbol{x}})} \frac{f(\widetilde{\boldsymbol{x}})}{f(\boldsymbol{x})} - 1$$
$$= (1 + \epsilon_{alg})(1 + \epsilon_{in}) - 1 = \epsilon_{alg} + \epsilon_{in} + \epsilon_{alg}\epsilon_{in}.$$

Questa formula contiene due termini lineari e un termine non lineare negli errori ϵ_{alg} e ϵ_{in}. Si fa l'ipotesi che la precisione di macchina u con cui si opera sia un numero molto più piccolo di 1 e che gli errori ϵ_{alg} e ϵ_{in} siano dell'ordine di u. Allora, in presenza dei termini lineari, il contributo del termine non lineare può essere trascurato. L'analisi dell'errore che si fa operando in questo modo viene detta *analisi dell'errore al primo ordine*. Indicando con il simbolo $\doteq$ l'uguaglianza in un'analisi dell'errore al primo ordine, vale

$$\epsilon_{tot} \doteq \epsilon_{alg} + \epsilon_{in}. \tag{1.12}$$

Se la funzione f è differenziabile due volte in un intorno di $\boldsymbol{x}$, possiamo esprimere l'errore inerente mediante gli errori di rappresentazione dei dati x_i. Consideriamo prima il caso $n = 1$, cioè $\boldsymbol{x}$ ha una sola componente che indichiamo con x. Dalla formula di Taylor si ha

$$f(\widetilde{x}) - f(x) = f'(x)(\widetilde{x} - x) + O((\widetilde{x} - x)^2).$$

Se $x \neq 0$ si ha $(\widetilde{x}-x)^2 = x^2\epsilon_x^2$, per cui l'ultimo termine risulta non lineare nell'errore. In un'analisi dell'errore al primo ordine esso viene trascurato e si ottiene

$$\frac{f(\widetilde{x}) - f(x)}{f(x)} \doteq \frac{xf'(x)}{f(x)} \, \frac{\widetilde{x} - x}{x},$$

cioè $$\epsilon_{in} \doteq c_x \epsilon_x, \quad \text{dove} \quad c_x = \frac{xf'(x)}{f(x)} \, .$$

Il coefficiente c_x è detto *coefficiente di amplificazione* e dà una misura di quanto influisce l'errore relativo ϵ_x, da cui è affetto il dato x, sul risultato: se c_x ha modulo elevato, anche un piccolo errore ϵ_x può indurre un grosso errore su $f(x)$. In questo caso il problema del calcolo di $f(x)$ è *mal condizionato*.

Esempio. La funzione $f(x) = x^k$ ha coefficiente di amplificazione $c_x = k$ e quindi il problema del calcolo di $f(x)$ è tanto peggio condizionato quanto più grande è k. Nel caso $x = 1.00001$, l'errore di rappresentazione di x in base 2 con 24 cifre è dell'ordine di 10^{-7}, l'errore da cui è affetto x^k è dell'ordine di 10^{-5} per $k = 10$ e di 10^{-4} per $k = 100$. □

Passando al caso $n = 2$, scriviamo la formula di Taylor di $f(\boldsymbol{x})$ con i soli termini lineari

$$f(\widetilde{\boldsymbol{x}}) - f(\boldsymbol{x}) = f(\widetilde{x}_1, \widetilde{x}_2) - f(x_1, x_2) \doteq \frac{\partial f(\boldsymbol{x})}{\partial x_1} \, (\widetilde{x}_1 - x_1) \; + \frac{\partial f(\boldsymbol{x})}{\partial x_2} \, (\widetilde{x}_2 - x_2).$$

Quindi se $x_1 \neq 0$ e $x_2 \neq 0$, in un'analisi dell'errore al primo ordine si ha

$$\frac{f(\widetilde{\boldsymbol{x}}) - f(\boldsymbol{x})}{f(\boldsymbol{x})} \doteq \frac{x_1}{f(\boldsymbol{x})} \, \frac{\partial f(\boldsymbol{x})}{\partial x_1} \, \frac{\widetilde{x}_1 - x_1}{x_1} + \frac{x_2}{f(\boldsymbol{x})} \, \frac{\partial f(\boldsymbol{x})}{\partial x_2} \, \frac{\widetilde{x}_2 - x_2}{x_2},$$

cioè

$$\epsilon_{in} \doteq c_1\epsilon_1 + c_2\epsilon_2, \quad \text{dove} \quad c_i = \frac{x_i}{f(\boldsymbol{x})} \, \frac{\partial f(\boldsymbol{x})}{\partial x_i} \quad \text{e} \quad \epsilon_i = \frac{\widetilde{x}_i - x_i}{x_i}. \tag{1.13}$$

Anche in questo caso i coefficienti c_i sono detti *coefficienti di amplificazione*. Se almeno uno dei coefficienti di amplificazione ha modulo elevato, il problema del calcolo di $f(\boldsymbol{x})$ è *mal condizionato*.

Nel caso generale, in cui n è un intero qualsiasi, si dimostra che vale una formula analoga alla (15), per cui ϵ_{in} è dato dalla somma di n termini, ognuno riferito ad una diversa variabile.

1.10 Errore nelle operazioni di macchina.

Si esamina ora il caso in cui la funzione f sia una delle operazioni aritmetiche. In questo caso l'errore algoritmico è dato dal solo errore locale ϵ dell'operazione. Da (1.12) e (1.13) si ha

$$\epsilon_{tot} \doteq \epsilon + c_1\epsilon_1 + c_2\epsilon_2, \quad \text{dove} \quad |\epsilon| < u,$$

e

a) se $f(x_1, x_2) = x_1 \pm x_2$, è $\quad c_1 = \dfrac{x_1}{x_1 \pm x_2}, \quad c_2 = \dfrac{\pm x_2}{x_1 \pm x_2},$

b) se $f(x_1, x_2) = x_1 x_2$, è $\quad c_1 = 1, \quad c_2 = 1,$

c) se $f(x_1, x_2) = x_1/x_2$, è $\quad c_1 = 1, \quad c_2 = -1.$

Nel caso dell'addizione di due numeri x_1 e x_2 dello stesso segno (o nel caso della sottrazione di due numeri x_1 e x_2 di segno opposto), i coefficienti di amplificazione sono limitati in modulo da 1 e si può dare una limitazione superiore del modulo dell'errore totale indipendente da x_1 e x_2; nel caso dell'addizione di due numeri x_1 e x_2 di segno opposto (o nel caso della sottrazione di due numeri x_1 e x_2 dello stesso segno), questo non è in genere possibile. Tanto più piccolo è il modulo di $x_1 + x_2$, tanto più grandi sono i moduli dei coefficienti di amplificazione c_1 e c_2.

In questi casi l'elevato errore che si può ritrovare nel risultato non è generato dalla operazione aritmetica in virgola mobile, infatti l'errore locale dell'operazione ha modulo minore di u, ma è dovuto alla presenza degli errori relativi non nulli ϵ_1 ed ϵ_2 in $\widetilde{x}_1$ e $\widetilde{x}_2$ che sono molto amplificati dalle operazioni aritmetiche di addizione e di sottrazione.

Ad esempio, rappresentando i numeri $x_1 = 0.123456$ e $x_2 = -0.123454$ in $\mathcal{F}_{(10,5,m,M)}$ con arrotondamento, si ottiene rispettivamente

$$\widetilde{x}_1 = \mathrm{arr}(x_1) = 0.12346\ 10^0, \quad \widetilde{x}_2 = \mathrm{arr}(x_2) = -0.12345\ 10^0.$$

Vale inoltre $\widetilde{x}_1 \oplus \widetilde{x}_2 = 0.1\ 10^{-4}$, e poiché $x_1 + x_2 = 0.2\ 10^{-5}$, nessuna cifra di $\widetilde{x}_1 \oplus \widetilde{x}_2$ è esatta e si ha $|\epsilon_{tot}| = 4$.

L'amplificazione dell'errore generata dalla addizione di numeri di segno opposto o dalla sottrazione di numeri dello stesso segno è detta *fenomeno di cancellazione numerica*; effettuando una sottrazione di numeri dello stesso segno con le prime cifre uguali, queste si cancellano a due a due. Nelle operazioni aritmetiche di moltiplicazione e divisione questo fenomeno non si verifica e l'errore presente negli operandi non viene amplificato.

Il fenomeno della cancellazione numerica, quando si verifica nel corso di un algoritmo, è alla base di gran parte dei casi di instabilità numerica, come negli esempi visti nel paragrafo 1.1.

Nel calcolo delle radici x_1 e x_2 dell'equazione $ax^2 + bx + c = 0$, mediante le formule

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}, \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a},$$

(nell'esempio è $a = c = 1$, $b = -q$), si può presentare cancellazione numerica nel calcolo della radice di modulo minore, cioè di x_1 se $b > 0$ e di x_2 se $b < 0$. Per eliminare il fenomeno di cancellazione numerica, calcolata la radice di modulo maggiore, si ricava la radice di modulo minore sfruttando la relazione $x_1 x_2 = c/a$.

Nel calcolo della funzione

$$f(x) = x(\sqrt{x^2 + 1} - x)$$

si presenta il fenomeno della cancellazione numerica quando x assume valori grandi, per i quali $\sqrt{x^2+1}-x$ è molto minore di x. Invece con la funzione

$$f(x) = \frac{x}{\sqrt{x^2+1}+x}$$

non si presenta il fenomeno di cancellazione numerica.

Anche nel calcolo di e^{-20}, quando si utilizza la serie a segni alterni, si presenta il fenomeno di cancellazione numerica.

1.11 Uso dei grafi per l'analisi dell'errore.

Si esamina ora il problema del calcolo di una funzione razionale f che non sia una singola operazione aritmetica. Il valore $\psi(\widetilde{\boldsymbol{x}})$ effettivamente calcolato al posto di $f(\boldsymbol{x})$ è ottenuto sostituendo ai dati x_i i corrispondenti numeri di macchina $\widetilde{x}_i$, $i = 1, 2, \ldots, n$, e alle operazioni aritmetiche le corrispondenti operazioni di macchina, specificando anche l'ordine in cui esse vengono eseguite. La funzione ψ, che dipende dall'aritmetica di macchina, è ottenuta implementando un algoritmo costituito da p passi del tipo:

$$z^{(i)} = y_1^{(i)} \text{ op } y_2^{(i)}, \quad \text{per } i = 1, 2, \ldots, p, \tag{1.14}$$

dove gli operandi $y_1^{(i)}$ e $y_2^{(i)}$ possono essere dati iniziali o risultati delle operazioni ai passi precedenti, cioè

$$y_1^{(i)},\ y_2^{(i)} \in \{x_1, x_2, \ldots, x_n, z^{(1)}, z^{(2)}, \ldots, z^{(i-1)}\} \quad \text{e} \quad f(\boldsymbol{x}) = z^{(p)}.$$

Un esempio chiarirà meglio le posizioni fatte. Sia $f(x_1, x_2) = x_1^2 - x_2^2$. Questa funzione può essere calcolata con il seguente algoritmo:

$$\begin{aligned} z^{(1)} &= x_1 \cdot x_1 \\ z^{(2)} &= x_2 \cdot x_2 \\ z^{(3)} &= z^{(1)} - z^{(2)}. \end{aligned} \tag{1.15}$$

Per l'errore $\epsilon_{tot}^{(i)}$ del risultato effettivamente calcolato all'i-esimo passo dalla (1.14) si ha:

$$\epsilon_{tot}^{(i)} \doteq \epsilon^{(i)} + c_1^{(i)}\epsilon_1^{(i)} + c_2^{(i)}\epsilon_2^{(i)}, \quad i = 1,\ 2, \ldots, p, \tag{1.16}$$

dove $\epsilon^{(i)}$ è l'errore locale generato dalla i-esima operazione e $\epsilon_1^{(i)}$ e $\epsilon_2^{(i)}$ sono gli errori presenti negli operandi della i-esima operazione e possono essere errori di dati iniziali oppure errori accumulati nei risultati intermedi del calcolo: per $\epsilon_1^{(i)}$ ad esempio,

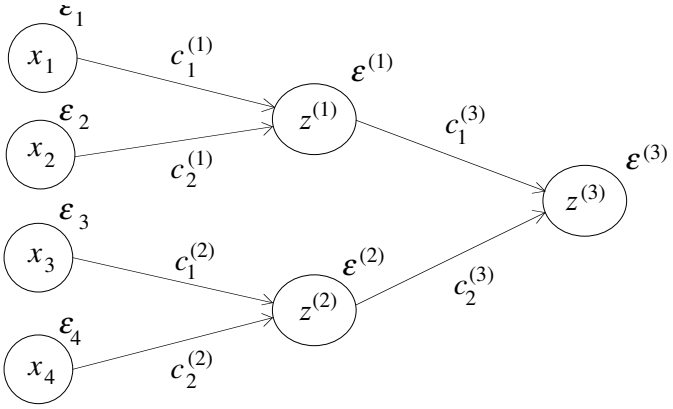
se $\quad y_1^{(i)} = x_j$ per qualche $j,\ 1 \le j \le n$, è $\epsilon_1^{(i)} = \epsilon_j$,

altrimenti se $\quad y_1^{(i)} = z^{(j)}$ per qualche $j,\ 1 \le j \le i-1$, è $\epsilon_1^{(i)} = \epsilon_{tot}^{(j)}$.

Inoltre, essendo $z^{(p)} = f(\boldsymbol{x})$, si ha $\epsilon_{tot} = \epsilon_{tot}^{(p)}$.

Con applicazioni ripetute della formula (1.16), si può esprimere l'errore ϵ_{tot} nei termini degli errori dei dati ϵ_i e degli errori locali $\epsilon^{(i)}$ delle operazioni (1.14).

Per valutare l'errore inerente, che è indipendente dall'algoritmo usato, conviene utilizzare la (1.13), mentre per valutare l'errore algoritmico, che ovviamente dipende dall'algoritmo, conviene utilizzare un *grafo*, che descrive la sequenza delle operazioni dell'algoritmo ed è costruito nel modo seguente: i nodi corrispondono ai dati x_i, $i = 1, 2, \ldots, n$, e ai risultati intermedi $z^{(i)}$, $i = 1, 2, \ldots, p$; in corrispondenza ai nodi x_i sono riportati gli errori ϵ_i di rappresentazione dei dati, e ai nodi $z^{(i)}$ sono riportati gli errori locali $\epsilon^{(i)}$ delle operazioni aritmetiche. Ai nodi $z^{(i)}$ arrivano gli archi orientati provenienti dai nodi corrispondenti agli operandi della i-esima operazione; in corrispondenza a ciascun arco è riportato il relativo coefficiente di amplificazione. Nella figura è riportato il grafo corrispondente ad un algoritmo composto da tre operazioni per il calcolo di $f(x_1, x_2, x_3, x_4)$.



L'errore relativo totale si ottiene percorrendo il grafo dall'ultimo nodo verso i nodi iniziali. Ad ogni nodo che si incontra, viene calcolato $\epsilon_{tot}^{(i)}$ dato dalla (1.16), cioè si sommano l'errore $\epsilon^{(i)}$ che corrisponde al nodo e gli errori precedentemente accumulati nei nodi ad esso collegati, moltiplicati per i coefficienti di amplificazione corrispondenti agli archi percorsi.

Per il grafo rappresentato nella figura precedente

al nodo $z^{(3)}$, è $\epsilon_{tot}^{(3)} \doteq \epsilon^{(3)} + c_1^{(3)}\epsilon_{tot}^{(1)} + c_2^{(3)}\epsilon_{tot}^{(2)}$, dove $\epsilon_{tot}^{(1)}$ è l'errore accumulato al nodo $z^{(1)}$ e $\epsilon_{tot}^{(2)}$ è l'errore accumulato al nodo $z^{(2)}$;

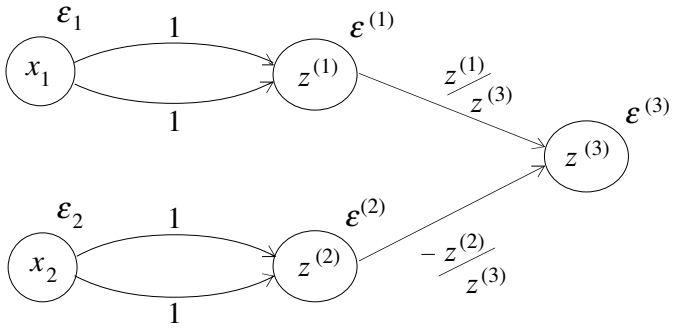
al nodo $z^{(1)}$, è $\epsilon_{tot}^{(1)} \doteq \epsilon^{(1)} + c_1^{(1)}\epsilon_1 + c_2^{(1)}\epsilon_2$, dove ϵ_1 è l'errore di rappresentazione di x_1 e ϵ_2 è l'errore di rappresentazione di x_2;

al nodo $z^{(2)}$, è $\epsilon_{tot}^{(2)} \doteq \epsilon^{(2)} + c_1^{(2)}\epsilon_3 + c_2^{(2)}\epsilon_4$, dove ϵ_3 è l'errore di rappresentazione di x_3 e ϵ_4 è l'errore di rappresentazione di x_4.

Esempio. L'errore inerente della funzione $f(x_1, x_2) = x_1^2 - x_2^2$ è dato da

$$\epsilon_{in} \doteq \frac{2x_1^2}{x_1^2 - x_2^2}\,\epsilon_1 - \frac{2x_2^2}{x_1^2 - x_2^2}\,\epsilon_2.$$

Quindi il problema del calcolo di $f(x_1, x_2)$ è mal condizionato quando il modulo di $x_1^2 - x_2^2$ è molto più piccolo di x_1^2 e di x_2^2. Si vuole ora determinare l'errore totale che si produce nel calcolo di $f(x_1, x_2)$, con l'algoritmo (1.15). Dal grafo riportato nella prossima figura si ricava l'errore totale:



$$\begin{aligned}\epsilon_{tot1} &\doteq \epsilon^{(3)} + \frac{z^{(1)}}{z^{(3)}}\,(\epsilon^{(1)} + 2\epsilon_1) - \frac{z^{(2)}}{z^{(3)}}\,(\epsilon^{(2)} + 2\epsilon_2) \\ &\doteq \epsilon^{(3)} + \frac{x_1^2}{x_1^2 - x_2^2}\,\epsilon^{(1)} - \frac{x_2^2}{x_1^2 - x_2^2}\,\epsilon^{(2)} + \frac{2x_1^2}{x_1^2 - x_2^2}\,\epsilon_1 - \frac{2x_2^2}{x_1^2 - x_2^2}\,\epsilon_2,\end{aligned}$$

dove $\epsilon^{(1)}$, $\epsilon^{(2)}$, $\epsilon^{(3)}$ sono gli errori locali delle operazioni. Quindi l'errore algoritmico è

$$\epsilon_{alg1} \doteq \epsilon^{(3)} + \frac{x_1^2}{x_1^2 - x_2^2}\,\epsilon^{(1)} - \frac{x_2^2}{x_1^2 - x_2^2}\,\epsilon^{(2)}.$$

La stessa funzione può essere calcolata anche con il seguente algoritmo:

$$\begin{aligned}v^{(1)} &= x_1 + x_2 \\ v^{(2)} &= x_1 - x_2 \\ v^{(3)} &= v^{(1)} \cdot v^{(2)}.\end{aligned}$$

Dal grafo corrispondente si ricava l'errore totale:

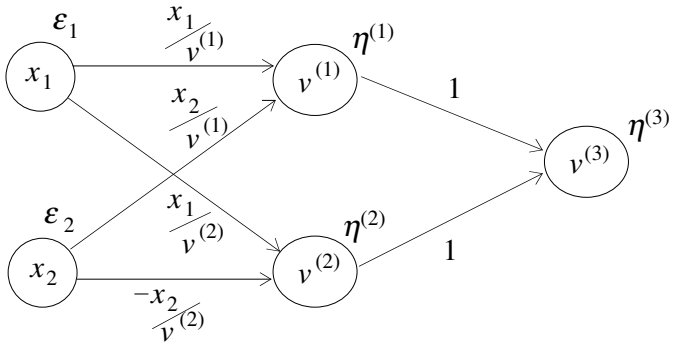
$$\begin{aligned}\epsilon_{tot2} &\doteq \eta^{(3)} + \left[\eta^{(1)} + \frac{x_1}{v^{(1)}}\,\epsilon_1 + \frac{x_2}{v^{(1)}}\,\epsilon_2\right] + \left[\eta^{(2)} + \frac{x_1}{v^{(2)}}\,\epsilon_1 - \frac{x_2}{v^{(2)}}\,\epsilon_2\right] \\ &\doteq \eta^{(1)} + \eta^{(2)} + \eta^{(3)} + \frac{2x_1^2}{x_1^2 - x_2^2}\,\epsilon_1 - \frac{2x_2^2}{x_1^2 - x_2^2}\,\epsilon_2,\end{aligned}$$

dove $\eta^{(1)}$, $\eta^{(2)}$, $\eta^{(3)}$ sono gli errori locali delle operazioni. Quindi l'errore algoritmico è

$$\epsilon_{alg2} \doteq \eta^{(1)} + \eta^{(2)} + \eta^{(3)}.$$

Si confrontano i due errori algoritmici. Risulta

$$|\epsilon_{alg1}| < \left(1 + \frac{x_1^2 + x_2^2}{|x_1^2 - x_2^2|}\right) u, \quad |\epsilon_{alg2}| < 3u,$$



perciò il secondo algoritmo è più stabile del primo se $x_1^2 + x_2^2 > |x_1^2 - x_2^2|$. □

L'analisi dell'errore condotta con il grafo permette di valutare sia l'errore algoritmico che l'errore inerente; poiché però per l'errore inerente conviene utilizzare la (1.13), si può semplificare l'analisi utilizzando il grafo per il calcolo del solo errore algoritmico, assumendo nulli gli errori di rappresentazione dei dati, come sarà fatto negli esempi che seguono.

1.12 Errore nel calcolo di una somma

Particolarmente importante è lo studio della propagazione dell'errore nel calcolo della funzione

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i.$$

Supponiamo che $x_i \neq 0$ per ogni i e che $f(\boldsymbol{x}) \neq 0$. Dalla (1.13) si ha

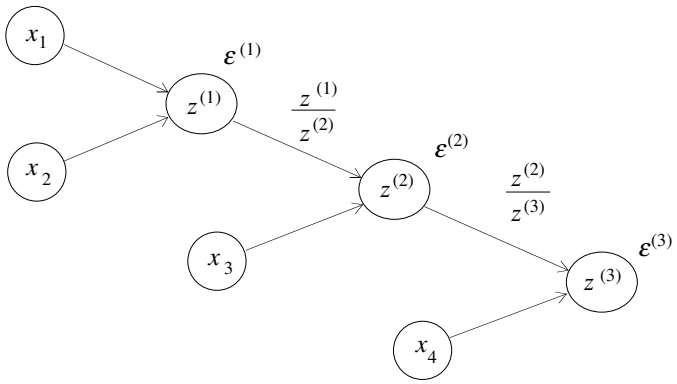
$$\epsilon_{in} = \frac{1}{f(\boldsymbol{x})} \sum_{i=1}^{n} x_i \epsilon_i.$$

Anche se $|\epsilon_i| < u$ per $i = 1, \ldots, n$, nel caso generale non si possono dare limitazioni dell'errore inerente che non dipendano dai dati x_i. Se però i dati sono tutti dello stesso segno, si può dare una limitazione molto semplice. Supponiamo ad esempio che i dati siano tutti positivi. Allora risulta

$$|\epsilon_{in}| \leq \frac{1}{f(\boldsymbol{x})} \sum_{i=1}^{n} x_i |\epsilon_i| < \frac{u}{f(\boldsymbol{x})} \sum_{i=1}^{n} x_i = u.$$

Il primo algoritmo è quello che segue l'ordine naturale degli addendi

$$\begin{aligned} &z^{(1)} = x_1 + x_2, \\ &z^{(i)} = z^{(i-1)} + x_{i+1}, \quad i = 2, \ldots, n-1, \\ &f(\boldsymbol{x}) = z^{(n-1)}. \end{aligned}$$



Nella prossima figura è riportato il grafo corrispondente al caso $n = 4$. Poiché l'errore inerente è già stato determinato, nel grafo non sono state riportate le indicazioni relative agli errori di rappresentazione degli x_i e ai corrispondenti coefficienti di amplificazione. Per l'errore algoritmico si ottiene dal grafo:

$$\epsilon_{alg1} \doteq \frac{1}{f(\boldsymbol{x})} \sum_{i=1}^{n-1} \Big(\sum_{j=1}^{i+1} x_j \Big) \epsilon^{(i)}.$$

In generale, anche per l'errore algoritmico non è possibile dare limitazioni che non dipendano dai dati. Se però i dati sono tutti positivi, risulta

$$\sum_{j=1}^{i+1} x_j \leq f(\boldsymbol{x}), \quad \text{per } i = 1, 2, \ldots, n-1,$$

e vale la limitazione

$$|\epsilon_{alg1}| \ < \ (n-1)u. \tag{1.17}$$

Una migliore limitazione dell'errore si può ottenere, sempre nel caso che i dati siano tutti positivi, ricorrendo all'algoritmo seguente (*algoritmo di addizione in parallelo*), descritto per semplicità per $n = 2^p$, con p intero positivo: si calcolino i valori $v_j^{(i)}$, $j = 1, \ldots, n/2^i$, $i = 0, \ldots, p$, così definiti:

$$\begin{aligned}
&v_j^{(0)} = x_j, \quad j = 1, \ldots, n; \\
&v_j^{(i)} = v_{2j-1}^{(i-1)} + v_{2j}^{(i-1)}, \quad j = 1, \ldots, \frac{n}{2^i}, \quad i = 1, \ldots, p, \\
&f(\boldsymbol{x}) = v_1^{(p)}.
\end{aligned}$$

Nella prossima figura è riportato il grafo corrispondente al caso $p = 3$. L'errore algoritmico è

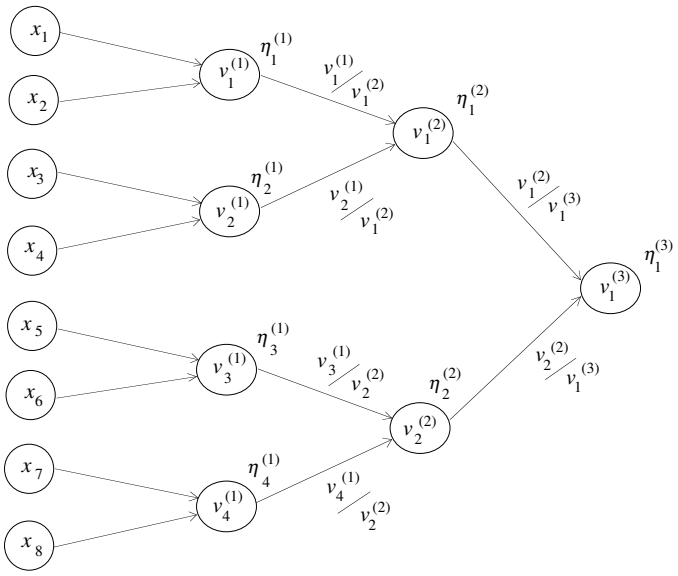
$$\epsilon_{alg2} \doteq \frac{1}{f(\boldsymbol{x})} \sum_{i=1}^{p} \sum_{j=1}^{n/2^i} v_j^{(i)} \eta_j^{(i)},$$

e poiché

$$\sum_{j=1}^{n/2^i} v_j^{(i)} = f(\boldsymbol{x}), \quad \text{per } i = 1, 2, \ldots, p,$$

si ottiene

$$|\epsilon_{alg2}| < \max_{i,j} |\eta_j^{(i)}| \sum_{i=1}^{p} \sum_{j=1}^{n/2^i} \frac{v_j^{(i)}}{f(\boldsymbol{x})} \leq \max_{i,j} |\eta_j^{(i)}| \; p < u \log_2 n. \tag{1.18}$$



Confrontando (1.17) con (1.18) si vede che, se i dati sono tutti positivi, l'algoritmo di addizione in parallelo è più stabile, come risulta ad esempio nel caso seguente.

Esempio. Gli algoritmi precedenti sono utilizzati per calcolare la somma degli n numeri $x_i = \text{arr}(i^{-3/2})$ per $i = 1, \ldots, n$ e $n = 2^{14}$. Indicati con ϵ_1 e ϵ_2 gli errori relativi delle somme effettivamente calcolate, prima seguendo l'ordine naturale $i = 1, 2, \ldots, n$, poi con l'algoritmo in parallelo, risulta:

$$\epsilon_1 = 0.21\ 10^{-4} \qquad \epsilon_2 = 0.57\ 10^{-7}. \qquad \square$$

1.13 Errore nel calcolo di una funzione non razionale.

Se la funzione f non è razionale, è necessario approssimarla con una funzione razionale $g(x)$: tale approssimazione introduce un *errore analitico*

$$\epsilon_{an}(\boldsymbol{x}) = \frac{g(\boldsymbol{x}) - f(\boldsymbol{x})}{f(\boldsymbol{x})}\,.$$

Detta ancora ψ la funzione effettivamente calcolata al posto della $g(x)$, se $g(\widetilde{\boldsymbol{x}}) \neq 0$, con procedimento analogo a quello utilizzato nel par. 1.9 si ha

$$\epsilon_{tot} = \frac{\psi(\widetilde{\boldsymbol{x}}) - f(\boldsymbol{x})}{f(\boldsymbol{x})} \doteq \epsilon_{in} + \epsilon_{alg} + \epsilon_{an}(\widetilde{\boldsymbol{x}}),$$

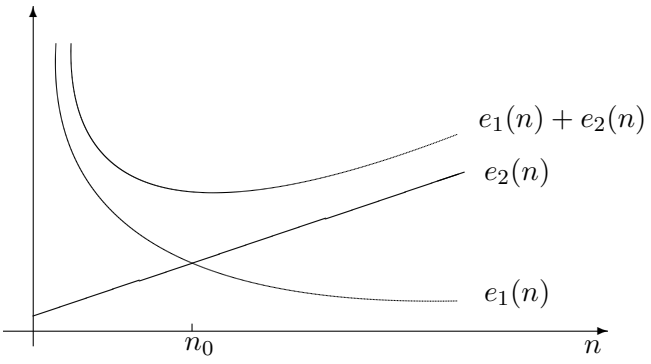
dove in questo caso è

$$\epsilon_{alg} = \frac{\psi(\widetilde{\boldsymbol{x}}) - g(\widetilde{\boldsymbol{x}})}{g(\widetilde{\boldsymbol{x}})}\,.$$

Nel seguito si suppone che $\epsilon_{an}(\widetilde{\boldsymbol{x}}) \doteq \epsilon_{an}(\boldsymbol{x})$ e, come già si è fatto per ϵ_{in} e ϵ_{alg}, si omette l'indicazione dell'argomento di ϵ_{an}. Risulta allora

$$\epsilon_{tot} \doteq \epsilon_{in} + \epsilon_{alg} + \epsilon_{an}.$$

Il modo più semplice per ottenere un'approssimazione razionale di una funzione non razionale è quello di ricorrere alla formula di Taylor. Considerando per semplicità il caso in cui la f sia una funzione di una sola variabile x, possiamo approssimare $f(x)$ con un polinomio di grado n ottenuto troncando all'$(n+1)$-esimo termine la formula di Taylor di $f(x)$). In generale l'errore analitico ϵ_{an} tende a diminuire quanto più elevato è n, mentre l'errore algoritmico tende ad aumentare con n. Indicando con $e_1(n)$ ed $e_2(n)$ le maggiorazioni di $|\epsilon_{an}|$ e $|\epsilon_{alg}|$ al variare di n, il diverso comportamento dei due errori può essere qualitativamente rappresentato come nella figura. Quindi conviene scegliere un valore di n vicino a n_0, perché per n molto più grande



di n_0, ad un maggior volume di calcolo non corrisponde in generale una diminuzione dell'errore effettivamente prodotto.

Esempio. Si vuole approssimare il valore di $\log 0.6$. La funzione logaritmo può essere approssimata con un polinomio troncando la serie di Taylor all'n-esimo termine nel modo seguente:

$$\log(1+x) = \sum_{i=1}^{n} (-1)^{i-1}\,\frac{x^i}{i} + r(x), \quad \text{per } |x| < 1,$$

dove

$$r(x) = \frac{(-1)^n x^{n+1}}{(n+1)\,(1+\xi)^{n+1}}\,, \quad |\xi| < |x|.$$

L'errore analitico è dato da

$$\epsilon_{an} = \frac{r(x)}{\log(1+x)}\,,$$

e quindi $|\epsilon_{an}|$ è una funzione decrescente al crescere di n e tende a zero per $n \to \infty$. Nel nostro caso è $x = -0.4$. Poiché per $-1 < x < 0$ è $|\log(1+x)| > |x|$ e $1 + x < 1 + \xi < 1$, risulta

$$|\epsilon_{an}| < \frac{|x|^n}{(n+1)(1+x)^{n+1}}\,.$$

Per il calcolo si usa l'algoritmo

$$\begin{aligned} &s_0 = 0, \quad t_0 = -1, \\ &t_i = -x\, t_{i-1}, \quad s_i = s_{i-1} + \frac{t_i}{i}\,, \quad \text{per } i = 1, \ldots, n. \end{aligned}$$

Utilizzando i risultati del par. 1.12, si ha che l'errore algoritmico è dato da

$$\epsilon_{alg} \doteq \frac{1}{s_n} \sum_{i=1}^{n} \Big(\frac{t_i}{i}\, \epsilon_i + s_i \epsilon^{(i)} \Big),$$

in cui ϵ_i è l'errore da cui è affetto $\dfrac{t_i}{i}$ e $\epsilon^{(i)}$ è l'errore locale della i-esima addizione, e quindi

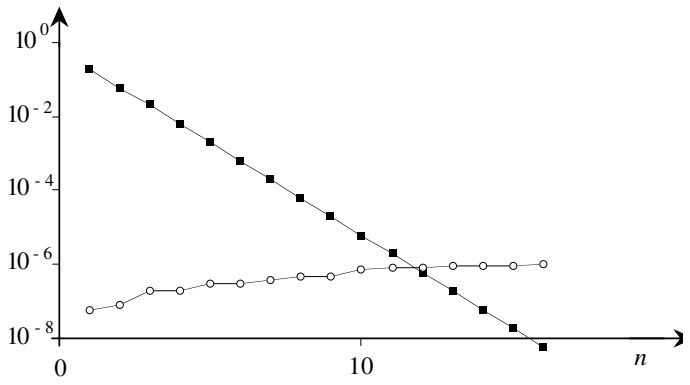
$$\epsilon_1 = 0, \quad |\epsilon_i| < iu, \quad |\epsilon^{(i)}| < u, \quad \text{per} \quad i = 1, \ldots, n.$$

Poiché gli s_i sono tutti positivi, è

$$\begin{aligned} |\epsilon_{alg}| \; &< \frac{u}{s_n} \sum_{i=1}^{n} (t_i + s_i) = \frac{u}{s_n} \left[\sum_{i=1}^{n} |x|^i + \sum_{i=1}^{n} \sum_{j=1}^{i} \frac{|x|^i}{i} \right] \\ &= \frac{u}{s_n} \left[\sum_{i=1}^{n} |x|^i + \sum_{i=1}^{n} \sum_{j=i}^{n} \frac{|x|^i}{i} \right] = \frac{(n+1)u}{s_n} \sum_{i=1}^{n} \frac{|x|^i}{i} \; = (n+1)u. \end{aligned}$$

L'errore algoritmico è quindi maggiorato in modulo da una funzione crescente con n. Nella figura sono riportati i grafici dei moduli degli errori analitico (con i quadratini neri) e algoritmico (con i pallini) effettivamente prodotti per $x = -0.4$. La scelta più conveniente per n è $n_0 = 12$. □

Molte funzioni non razionali fra le più comuni (come le funzioni trigonometriche, esponenziale, logaritmo e radice quadrata) possono essere calcolate utilizzando programmi inclusi in librerie di software, che implementano algoritmi per i quali gli errori algoritmici e analitici corrispondenti sono limitati superiormente in modulo da quantità dell'ordine della precisione di macchina. La valutazione dell'errore totale



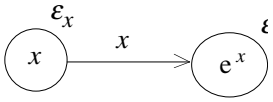
da cui è affetta una funzione non razionale può ancora essere fatta usando i grafi: ad ogni utilizzazione di una funzione di libreria si fa corrispondere un nodo, a cui arrivano tanti archi quanti sono gli argomenti della funzione.

Esempio. Per calcolare e^x si suppone di avere a disposizione una funzione di libreria $\mathrm{EXP}(x)$, che per ogni numero di macchina x soddisfi la relazione

$$\mathrm{EXP}(x) = \mathrm{e}^x(1+\epsilon), \quad |\epsilon| = |\epsilon_{alg} + \epsilon_{an}| < u.$$

Nel grafo ad ogni nodo corrispondente ad un valore di e^x arriva un solo arco con il coefficiente di amplificazione

$$c_x = \frac{x f'(x)}{f(x)} = \frac{x\mathrm{e}^x}{\mathrm{e}^x} = x,$$



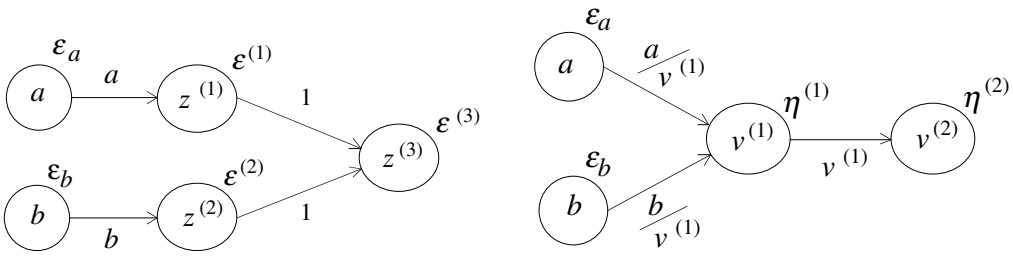
quindi $\epsilon_{tot} \doteq \epsilon + x\,\epsilon_x$.

Supponiamo ora di voler calcolare $f(a,b) = \mathrm{e}^a\,\mathrm{e}^b = \mathrm{e}^{a+b}$. Si possono usare i due algoritmi seguenti:

$$\begin{aligned} z^{(1)} &= \mathrm{e}^a \\ z^{(2)} &= \mathrm{e}^b \\ z^{(3)} &= z^{(1)} \cdot z^{(2)}, \end{aligned} \qquad \begin{aligned} v^{(1)} &= a + b \\ v^{(2)} &= \mathrm{e}^{v^{(1)}}. \end{aligned}$$

Ad un primo esame si potrebbe pensare che se a e b hanno segno discorde, il secondo algoritmo possa essere meno stabile del primo. Vediamo se questa idea è confermata o meno, studiando gli errori algoritmici. Nella figura sono riportati i grafi corrispondenti. Per il primo algoritmo è

$$\epsilon_{tot1} = \epsilon^{(1)} + \epsilon^{(2)} + \epsilon^{(3)} + a\epsilon_a + b\epsilon_b,$$



e quindi $|\epsilon_{alg1}| < 3u$. Per il secondo algoritmo è

$$\epsilon_{tot2} = \eta^{(2)} + v^{(1)}(\eta^{(1)} + \frac{a}{v^{(1)}}\ \epsilon_a + \frac{b}{v^{(1)}}\ \epsilon_b) = (a+b)\eta^{(1)} + \eta^{(2)} + a\epsilon_a + b\epsilon_b,$$

e quindi $|\epsilon_{alg2}| < (1+|a+b|)u$. Il secondo algoritmo risulta quindi più stabile quando $|a+b|$ è piccolo, meno stabile quando $|a+b|$ è elevato, contrariamente a quello che si era pensato all'inizio. Infatti il possibile fenomeno di cancellazione nel calcolo di $a+b$ non si trasmette né nell'errore inerente, né in quello algoritmico. □

1.14 Lo standard IEEE.

Lo standard IEEE è adottato per la rappresentazione floating point dalle principali case costruttrici di personal computer. Lo standard accetta tre categorie di dati:

(1) numeri normalizzati,

(2) numeri non normalizzati,

(3) numeri speciali.

La base usata è $\beta = 2$ e vi sono tre possibili precisioni: semplice, doppia, estesa.

Iniziamo con la precisione semplice: i numeri sono allocati in parole di 32 bit, così ripartiti:

1 bit per il segno del numero,

8 bit per l'esponente,

23 bit per la mantissa.

(1) I numeri normalizzati corrispondono a configurazioni dell'esponente in cui non tutti i bit sono uguali a 0 e non tutti i bit sono uguali a 1. Poiché per questi numeri è sempre $d_1 = 1$, la prima cifra non viene rappresentata, cioè in questo standard le cifre della mantissa che vengono rappresentate sono $d_2, \ldots, d_{24}$. Questo artificio consente di aumentare di poco la precisione, che viene a corrispondere a $t = 24$ cifre significative.

Per quanto riguarda l'esponente, è $m = 125$ e $M = 128$, cioè il minimo numero positivo normalizzato che viene rappresentato è $\omega = (0.1)_2\ 2^{-125} = 2^{-126}$ (approssimativamente $1.2\ 10^{-38}$) e il massimo è $\Omega = (0.1\ldots1)_2\ 2^{128} = (1-2^{24})2^{128}$ (approssimativamente $3.4\ 10^{38}$). L'effettiva rappresentazione dell'esponente p negli 8 bit ad

esso riservati viene fatta in traslazione, sommando 126 a p. Così il minimo esponente -125 viene rappresentato con $(1)_2 = 1$ e il massimo con $(11111110)_2 = 254$.

(2) I numeri non normalizzati corrispondono alla configurazione di tutti i bit uguali a 0 nel campo dell'esponente. Per questi dati è $d_1 = 0$ e nel campo della mantissa vengono rappresentate le cifre $d_2, \ldots, d_{24}$. In questo modo vengono rappresentati il numero 0 e i numeri

$$2^{-125} \sum_{i=k}^{24} d_i \, 2^{-i},$$

dove $k \geq 2$ è il più piccolo intero per cui $d_k \neq 0$. Il minimo numero positivo non normalizzato che può essere rappresentato è 2^{-149}, per cui i numeri positivi non normalizzati appartengono all'intervallo $[2^{-149}, 2^{-126})$. I numeri non normalizzati sono meno accurati di quelli normalizzati, in quanto rappresentati (a partire dal primo bit non nullo) con un numero di cifre minore di t. L'accuratezza è tanto più piccola quanto maggiore è k.

Il numero 0 risulta rappresentato con tutti bit nulli nei campi dell'esponente e della mantissa. Non è invece fissato il segno, che può essere sia $+$ che $-$. Per questo si usa dire che lo zero ha una doppia rappresentazione $+0$ e -0 (che ovviamente corrispondono allo stesso valore matematico).

Con l'introduzione nello standard IEEE dei numeri non normalizzati si è in parte risolto il problema dell'underflow. Se però il numero da rappresentare è più piccolo di 2^{-149}, il valore assegnato è 0.

(3) I numeri speciali corrispondono alla configurazione di tutti i bit uguali a 1 nel campo dell'esponente. Hanno un significato diverso a seconda della presenza o meno di bit diversi da zero nel campo della mantissa. Vengono utilizzati nel caso in cui nel corso dei calcoli si verifichino delle situazioni eccezionali.

Se i bit della mantissa sono tutti uguali a zero, il numero viene considerato come $+\infty$ se ha il segno positivo e come $-\infty$ se ha il segno negativo. Se almeno uno dei bit del campo della mantissa è diverso da 0, la quantità rappresentata non viene considerata un numero e viene indicata con il nome NaN (= *not a number*).

I valori $-\infty$ e $+\infty$ vengono assegnati nel caso in cui si verifichi una situazione di overflow, cioè si debba rappresentare un numero al di fuori dell'intervallo $[-\Omega, \Omega]$, e nel caso di una divisione per 0 di un numero $x \neq 0$. Un risultato ∞ viene anche fornito nel caso di un'operazione in cui almeno uno dei due operandi sia ∞, se ciò è in accordo con la teoria dei limiti, come ad esempio per $x + \infty$, $\infty/0$ e $x * \infty$ se $x \neq 0$. Ma nel caso di espressioni del tipo ∞/∞, $0 * \infty$ e $+\infty - (+\infty)$ non si può assegnare ∞ come risultato e infatti lo standard IEEE assegna un NaN. Ogni successiva operazione su un dato NaN ha come risultato un NaN.

La presenza nello standard IEEE di questi dati speciali consente di portare a termine l'esecuzione di un programma anche quando si verificano delle situazioni eccezionali (prima che questo standard fosse introdotto, l'esecuzione di un programma

poteva essere interrotta ad esempio nel caso di un overflow o di una divisione per zero).

La seguente tabella riassume la rappresentazione dei numeri in precisione semplice.

numero	campo dell'esponente	valore del numero
non normalizzato	$(00000000)_2$	$(.0\, d_2 \ldots d_{24})\ 2^{-125}$
normalizzato	$(00000001)_2$ $\vdots$ $(11111110)_2$	$(.1\, d_2 \ldots d_{24})\ 2^{-125}$ $\vdots$ $(.1\, d_2 \ldots d_{24})\ 2^{+128}$
speciale	$(11111111)_2$	$\pm\infty$ oppure NaN

La rappresentazione di un numero in precisione doppia e in precisione estesa è analoga a quella in precisione semplice. Per la precisione doppia viene usata una doppia parola di 64 bit, con 11 bit per l'esponente e 52 per la mantissa. Pertanto si ha $t = 53$, $m = 1021$ e $M = 1024$, e

numero	campo dell'esponente	valore del numero
non normalizzato	$(00000000000)_2$	$(.0\, d_2 \ldots d_{53})\ 2^{-1021}$
normalizzato	$(00000000001)_2$ $\vdots$ $(11111111110)_2$	$(.1\, d_2 \ldots d_{53})\ 2^{-1021}$ $\vdots$ $(.1\, d_2 \ldots d_{53})\ 2^{+1024}$
speciale	$(11111111111)_2$	$\pm\infty$ oppure NaN

Per la precisione estesa viene usata una parola di 80 bit, con 15 bit per l'esponente e 64 per la mantissa. L'unica differenza per la precisione estesa, rispetto a quella semplice e doppia, è la rappresentazione anche della cifra d_1.

Capitolo 2

Equazioni non lineari

Risolvere un'equazione in una variabile è uno dei problemi più comuni della matematica. Consideriamo il seguente esempio.

Esempio. Un famoso problema dell'antichità, noto come *problema di Archimede*, pone il seguente quesito: è dato un recipiente emisferico di raggio r. Si vuole sapere qual è l'altezza h del segmento sferico, il cui volume S è la metà del volume V del recipiente.

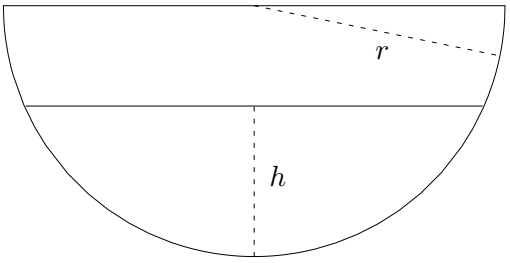


Figura 2.1: Il problema di Archimede.

Dalla geometria è noto che

$$V = \frac{2}{3}\,\pi r^3 \quad \text{e} \quad S = \frac{1}{3}\,\pi h^2(3r - h).$$

Il valore h cercato, che deve necessariamente essere compreso fra 0 e r, soddisfa la condizione

$$\frac{S}{V} = \frac{h^2(3r - h)}{2r^3} = \frac{1}{2}\,.$$

Ponendo $x = \dfrac{h}{r}$, si ottiene l'equazione

$$x^3 - 3x^2 + 1 = 0,$$

di cui si cerca una soluzione, che chiameremo α, compresa fra 0 e 1. □

Per un'equazione di terzo grado come quella dell'esempio esiste la formula risolutiva, che dà un'espressione esplicita delle soluzioni, ma si tratta di una formula assai complicata che richiede di operare con aritmetica complessa e di calcolare radici cubiche. D'altra parte, nella quasi totalità, una generica equazione della forma

$$f(x) = 0, \tag{2.1}$$

dove $f(x)$ è una qualsiasi funzione di x, non possiede formule risolutive, per cui si deve ricorrere a metodi iterativi che consentano di approssimare le soluzioni con una precisione prestabilita.

Una cosa importante da fare prima di affrontare la risoluzione di un'equazione con un qualunque metodo è quella di farsi un'idea dell'andamento della funzione $f(x)$, per determinare il numero delle soluzioni e *separare* ogni soluzione, cioè individuare, per ogni soluzione, un intervallo che non ne contenga altre. Questa prima indagine risulta molto semplificata se ci si può aiutare con un grafico della $f(x)$. Buoni intervalli di separazione possono essere difficili da determinare se vi sono soluzioni molto vicine fra di loro o se le soluzioni hanno molteplicità pari (cioè se sono anche punti di massimo o minimo), perché ci si basa di solito sui cambiamenti di segno della funzione.

I primi metodi che esamineremo richiedono che la $f(x)$ sia almeno continua in un intervallo $[a, b]$ (con $a < b$) e che assuma valori di segno opposto agli estremi, cioè che $f(a)f(b) < 0$. Si suppone inoltre che α sia l'unica soluzione compresa fra a e b.

2.1 Metodo di bisezione.

Il metodo più immediato per approssimare α è il metodo di *bisezione*, che procede suddividendo, ad ogni passo, l'intervallo $[a, b]$ a metà e determinando in quale dei due sottointervalli si trova la soluzione.

Si pone $a_0 = a,\ b_0 = b$. Per $i = 0, 1, \ldots$ si calcolano

$$x_{i+1} = \frac{a_i + b_i}{2} \quad \text{e} \quad f(x_{i+1});$$

se $f(a_i)f(x_{i+1}) < 0$, si pone $a_{i+1} = a_i$ e $b_{i+1} = x_{i+1}$;

se $f(a_i)f(x_{i+1}) > 0$, si pone $a_{i+1} = x_{i+1}$ e $b_{i+1} = b_i$;

se $f(x_{i+1}) = 0$, è $\alpha = x_{i+1}$.

Se la condizione $f(x_{i+1}) = 0$ non è mai verificata, il procedimento definisce una successione di intervalli $[a_i, b_i]$ contenenti α, ciascuno di lunghezza metà del precedente. Quindi la successione dei x_i converge ad α.

Il procedimento viene interrotto utilizzando un *criterio di arresto*: di solito si impone che

$$b_i - a_i < \epsilon, \tag{2.2}$$

dove $\epsilon > 0$ è una tolleranza prefissata o, se l'intervallo $[a_i, b_i]$ non contiene lo zero, l'analoga condizione sull'errore relativo

$$b_i - a_i < \epsilon \min\{|a_i|, |b_i|\}.$$

Questa condizione risulta più in sintonia con la caratteristica dell'aritmetica di macchina di mantenere una limitazione uniforme dell'errore relativo.

In teoria con questo procedimento si può approssimare una soluzione con un errore più piccolo di qualunque tolleranza prefissata; in realtà però se i calcoli sono eseguiti con precisione finita, l'errore non può scendere al di sotto di una certa soglia, determinata dalla precisione usata. Quindi la tolleranza ϵ non può essere scelta arbitrariamente piccola, e poiché non è facile stabilire a priori quanto elevato potrà essere l'errore di arrotondamento, è sempre bene imporre anche una limitazione superiore al numero di passi permessi: si eviterà in tal modo che il programma *vada in ciclo*. Tipicamente ad un programma per la risoluzione dell'equazione (2.1) si devono fornire i seguenti dati:

- la funzione $f(x)$ e gli estremi a e b dell'intervallo di separazione (si suppone che $f(a)$ e $f(b)$ siano diversi da zero e abbiano segno opposto),

- limitazioni superiori *tol* per l'errore e n per il numero di passi del metodo.

Il seguente algoritmo, scritto in Matlab, dà un'implementazione molto schematica del metodo di bisezione. Un indicatore `ind` segnala quando non è stato possibile ottenere un'approssimazione della soluzione con la tolleranza `tol` richiesta. In uscita i vettori `x` e `fx` contengono le iterate calcolate e i corrispondenti valori della $f(x)$.

```
d=sign(f(a)); i=0; ind=1;
while ind & i<n
  i=i+1;
  x0=(a+b)/2;
  f0=f(x0);
  if d*f0>0 a=x0; else b=x0; end
  ind= b-a>tol;
  x(i)=x0;
  fx(i)=f0;
end;
```

Il metodo di bisezione presenta, rispetto ai metodi che vedremo in seguito, il vantaggio di poter dire a priori quanti passi occorrono per ottenere l'approssimazione richiesta. Infatti, poiché

$$b_i - a_i = \frac{b-a}{2^i},$$

non tenendo conto dell'effetto degli errori di arrotondamento, la condizione (2.2) è verificata all'n-esimo passo, dove n è il minimo intero tale che

$$n > \log_2 \left(\frac{b-a}{\epsilon} \right).$$

Esempio. Per risolvere l'equazione del problema di Archimede si traccia il grafico della funzione $f(x) = x^3 - 3x^2 + 1$ nell'intervallo [0,1]. Appare evidente che l'e-

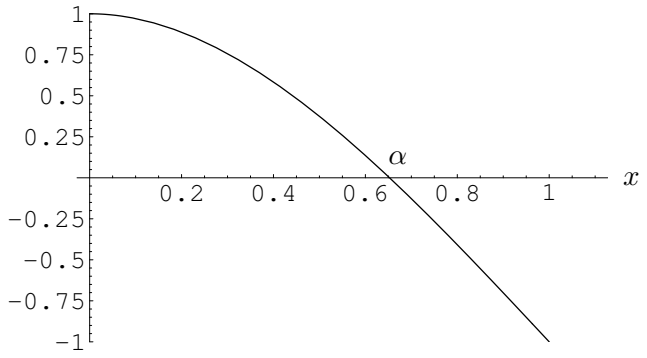


Figura 2.2: Grafico di $x^3 - 3x^2 + 1$.

quazione ha una sola soluzione reale $\alpha \in (0,1)$. Poiché $f(0) > 0$ e $f(1) < 0$, si può applicare il metodo di bisezione. Ponendo `tol` uguale a 10^{-3} si ottiene la successione

i	x_i	$f(x_i)$
1	0.50000	$3.7500\ 10^{-1}$
2	0.75000	$-2.6563\ 10^{-1}$
3	0.62500	$7.2266\ 10^{-2}$
4	0.68750	$-9.3018\ 10^{-2}$
5	0.65625	$-9.3689\ 10^{-3}$
6	0.64063	$3.1712\ 10^{-2}$
7	0.64844	$1.1236\ 10^{-2}$
8	0.65234	$9.4932\ 10^{-4}$
9	0.65430	$-4.2058\ 10^{-3}$
10	0.65332	$-1.6273\ 10^{-3}$

L'ampiezza dell'intervallo, che inizialmente è 1, dopo 10 passi è ridotta a $2^{-10} < 10^{-3}$, cioè $x_{10} = 0.65332$ approssima α con un errore assoluto minore di 10^{-3}. Con

la tolleranza 10^{-6}, al ventesimo passo si ottiene il punto $x_{20} = 0.65270$ in cui la funzione vale meno di 10^{-6}. □

Il metodo di bisezione ha uno svantaggio: è lento, perché l'errore ad ogni passo si riduce solo della metà. Ci sono altri metodi più efficaci, in quanto riescono di solito a far decrescere l'errore più rapidamente. Però in alcuni casi difficili il metodo di bisezione può essere l'unico strumento veramente utile.

2.2 Metodo delle secanti.

Con una piccola modifica dal metodo di bisezione si ottiene il metodo delle *secanti*. La differenza consiste nel fatto che il punto x_i non viene costruito come il punto medio dell'intervallo $[a_i, b_i]$ ma come il punto di intersezione con l'asse x del segmento che unisce i due estremi del grafico della funzione, cioè i punti $(a_i, f(a_i))$ e $(b_i, f(b_i))$.

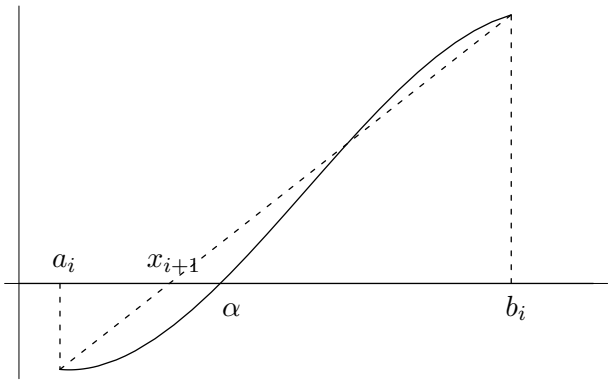


Figura 2.3: Metodo delle secanti.

L'equazione della retta passante per gli estremi è

$$y = f(a_i) + \frac{f(b_i) - f(a_i)}{b_i - a_i}\,(x - a_i),$$

il punto x_{i+1} in cui la retta interseca l'asse delle x si ottiene ponendo $y = 0$

$$x_{i+1} = a_i - \frac{f(a_i)(b_i - a_i)}{f(b_i) - f(a_i)}\,.$$

Si continua poi come per il metodo di bisezione, calcolando il valore $f(x_{i+1})$ e determinando quale dei due sottointervalli individuati da x_{i+1} contiene la soluzione α. Come nel metodo di bisezione, ogni passo del metodo delle secanti richiede una sola valutazione della funzione $f(x)$ se si ha l'accortezza di utilizzare al generico passo i valori di $f(x)$ calcolati precedentemente.

A differenza del metodo di bisezione, la successione delle lunghezze degli intervalli $[a_i, b_i]$ in generale non tende a zero, perché da un certo punto in poi uno dei due

estremi dell'intervallo a_i oppure b_i rimane invariato. Questo accade fin dall'inizio se la funzione $f(x)$ è concava o convessa nell'intervallo $[a, b]$, come si vede nella figura 2.4. In questo caso la successione x_i converge ad α in modo monotono, crescente

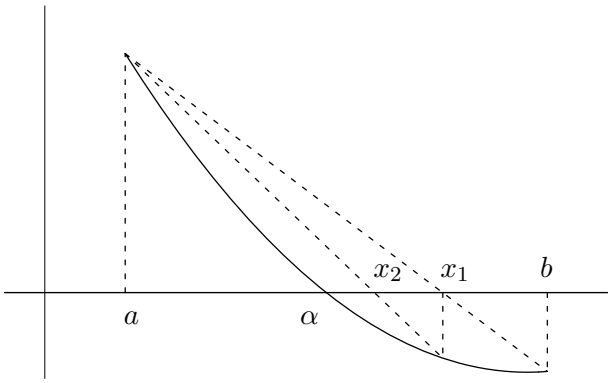


Figura 2.4: Metodo delle secanti: caso di funzione $f(x)$ convessa.

oppure decrescente. Perciò il controllo sull'approssimazione del risultato deve essere fatto per mezzo della distanza di x_{i+1} da x_i utilizzando il criterio di arresto

$$|x_{i+1} - x_i| < \epsilon, \tag{2.3}$$

dove $\epsilon > 0$ è una tolleranza prefissata, legata alla precisione con cui si vuole approssimare la soluzione.

Se $f \in C^1[a, b]$, poiché $f(\alpha) = 0$ dalla formula di Taylor si ha

$$f(x_i) = (x_i - \alpha)\, f'(\xi),$$

dove ξ è un opportuno punto vicino ad α. Perciò se $f'(x)$ varia poco quando x è vicino ad α, i successivi valori $f(x_i)$ calcolati danno un'idea di come diminuisce l'errore.

Il seguente algoritmo dà un'implementazione molto schematica del metodo delle secanti. Un indicatore `ind` segnala quando non è stato possibile ottenere un'approssimazione della soluzione con la tolleranza `tol` richiesta. In uscita i vettori `x` e `fx` contengono le iterate calcolate e i corrispondenti valori della $f(x)$.

```
fa=f(a);
fb=f(b);
x0=a;
i=0;
ind=1;
while ind & i<n
    i=i+1;
    x1=a-fa*(b-a)/(fb-fa);
    f1=f(x1);
```

```
      if fa*f1>0 a=x1; fa=f1;
      else b=x1; fb=f1;
      end
      ind=abs(x1-x0)>tol;
      x0=x1;
      x(i)=x1;
      fx(i)=f1;
  end;
```

Esempio. Riprendiamo l'equazione del problema di Archimede e applichiamo il metodo delle secanti con la stessa tolleranza `tol` di prima. Un confronto con la

i	x_i	$f(x_i)$
1	0.50000	$3.7500\ 10^{-1}$
2	0.63636	$4.2825\ 10^{-2}$
3	0.65130	$3.7093\ 10^{-3}$
4	0.65259	$3.1166\ 10^{-4}$
5	0.65269	$2.6116\ 10^{-5}$

tabella dei valori ottenuti con il metodo di bisezione mostra quanto il metodo delle secanti possa essere migliore di quello di bisezione. In soli 5 passi, e quindi con 7 valutazioni di $f(x)$, si è ottenuta un'approssimazione migliore di quella che aveva richiesto 11 valutazioni di $f(x)$ con il metodo di bisezione. □

Nel caso della successione calcolata sopra, l'estremo b_i resta invariato. Un semplice studio grafico consente di determinare facilmente quale dei due estremi resta fisso se la $f(x)$ è concava o convessa. Nel caso in cui la $f(x)$ sia derivabile due volte, la situazione è quella illustrata dal seguente teorema.

Teorema. *Sia $f \in C^2[a,b]$ e $f''(x) \neq 0$ per ogni $x \in [a,b]$. Si indichi con x_0 l'estremo a o b tale che $f(x_0)f''(x_0) < 0$. Allora la successione generata dal metodo delle secanti è monotona convergente ad α.*

2.3 Metodo delle tangenti.

Il terzo metodo che consideriamo è il *metodo delle tangenti*, che sotto certe ipotesi è molto più efficace dei due visti finora, ma che si può applicare solo se la funzione $f(x)$ è derivabile nell'intervallo $[a,b]$. Questo metodo differisce da quello delle secanti perché la retta che si considera è tangente alla curva di equazione $y = f(x)$. Il metodo

procede nel modo seguente: si sceglie nell'intervallo $[a, b]$ un punto che chiameremo x_0 e si considera la retta tangente alla curva nel punto $(x_0, f(x_0))$, di equazione

$$y = f(x_0) + f'(x_0)(x - x_0).$$

Il punto x_1 in cui la retta interseca l'asse delle x si ottiene ponendo $y = 0$

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}\,.$$

Se il punto x_0 è stato scelto opportunamente, il punto x_1 appartiene ancora all'in-

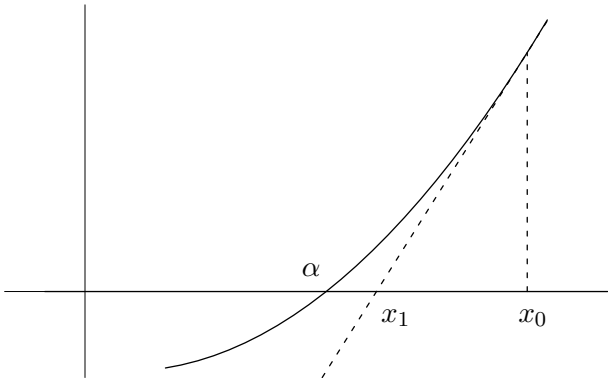


Figura 2.5: Metodo delle tangenti.

tervallo $[a, b]$ e risulta compreso fra α e x_0. Il metodo prosegue sostituendo x_1 al posto di x_0 e calcolando un altro punto x_2 e così via. In pratica si applica la formula

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}\,, \quad \text{per} \quad i = 0, 1, \ldots \tag{2.4}$$

Ogni passo del metodo richiede quindi due valutazioni di funzione: una della funzione $f(x)$ e una della derivata.

Il seguente algoritmo dà un'implementazione molto schematica del metodo delle tangenti. Sono richieste la funzione `f`, la derivata `f1`, il punto iniziale `x0`, il numero massimo `n` di passi permessi e la tolleranza `tol`. Il punto iniziale può coincidere con uno degli estremi. La condizione di arresto è la (2.3). Per semplicità non vi è, ma sarebbe bene che vi fosse, un ulteriore controllo che ad ogni passo verifichi che x_i appartiene all'intervallo $[a, b]$.

```
f0=feval(f,x0);
i=0; ind=1;
while ind & x0 >= a & x0 <= b & i<n
  i=i+1;
  x1=x0-f0/feval(f1,x0);
```

```
    ind=abs(x1-x0)>tol;
    x0=x1; f0=feval(f,x1);
    x(i)=x1; fx(i)=f0;
  end;
```

Una buona scelta del punto x_0 è fondamentale, perché se il punto x_0 è scelto in modo sbagliato si possono ottenere risultati del tutto imprevedibili: è infatti possibile che il metodo generi una successione x_i che tende ad un'altra soluzione della stessa equazione (se esiste), oppure che non ha un limite finito.

Esempio. Riprendiamo l'equazione del problema di Archimede, in cui

$$f(x) = x^3 - 3x^2 + 1, \qquad f'(x) = 3x^2 - 6x.$$

Applicando il metodo delle tangenti a partire da $x_0 = 1$ si ottiene Quindi con tre

i	x_i	$f(x_i)$
1	0.66667	$-3.7037\ 10^{-2}$
2	0.65278	$-1.9558\ 10^{-4}$
3	0.65270	$-5.7248\ 10^{-9}$

passi si è ottenuto lo stesso risultato che con gli altri due metodi. Se però scegliamo $x_0 = 0.1$ si ottiene $x_1 = 1.8035$ che è fuori dall'intervallo. Infatti con questa scelta di x_0 la successione ottenuta ha come limite il punto -0.53209, che è un'altra soluzione della stessa equazione, ma inaccettabile per il nostro problema. □

La scelta del punto x_0 risulta molto facilitata se la funzione $f(x)$ è concava o convessa nell'intervallo $[a, b]$. Un semplice studio grafico consente di determinare

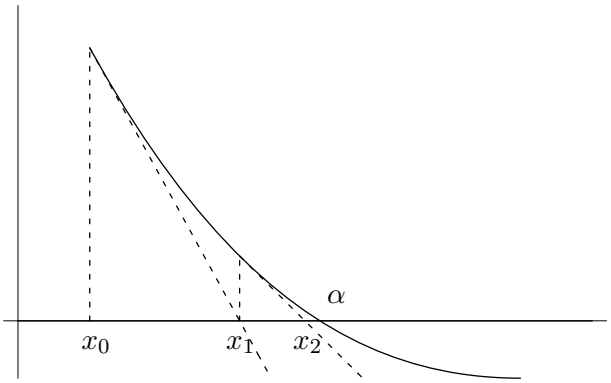


Figura 2.6: Metodo delle tangenti: caso di funzione $f(x)$ convessa.

facilmente vicino a quale dei due estremi vada scelto il punto x_0. Nel caso in cui la funzione $f(x)$ sia derivabile due volte, la situazione è quella illustrata dal seguente teorema.

Teorema. *Sia $f \in C^2[a,b]$ e $f'(x), f''(x) \neq 0$ per ogni $x \in [a,b]$, escluso al più il punto α. Si indichi, se esiste, con x_0 un punto di $[a,b]$ tale che $f(x_0)f''(x_0) > 0$. Allora la successione generata dal metodo delle tangenti è monotona convergente ad α.*

2.4 Metodo delle corde.

Il metodo delle tangenti in generale converge molto rapidamente, ma ad ogni passo si devono calcolare due diverse funzioni: la $f(x)$ e la $f'(x)$. Questo non è un problema nel caso delle funzioni algebriche o di funzioni la cui derivata non sia troppo complicata, ma può diventarlo nel caso opposto. Per ovviare in parte a questo inconveniente si può apportare la seguente modifica: si mantiene lo stesso valore del denominatore per un numero prefissato di passi (per esempio 5). In questi passi si calcola solo un nuovo valore di $f(x)$. Geometricamente questo vuol dire che in questi passi la retta che si traccia non è tangente al grafico di $f(x)$ ma mantiene lo stesso coefficiente angolare. Poi un nuovo calcolo di $f'(x)$ modifica il coefficiente angolare tracciando una nuova tangente.

È anche possibile mantenere costante il coefficiente angolare della retta durante tutto il procedimento iterativo. In questo caso il metodo, detto delle *corde*, calcola ad ogni iterazione il valore

$$x_{i+1} = x_i - \frac{f(x_i)}{m},$$

dove $m \neq 0$ è una costante scelta opportunamente, in relazione ai valori assunti da $f'(x)$ in un intorno di α. In generale la convergenza è più lenta di quella del metodo delle tangenti.

2.5 Calcolo della radice n-esima di un numero.

Il metodo delle tangenti è fra i metodi più usati del calcolo numerico: è il metodo di elezione quando la funzione $f(x)$ è derivabile e la sua derivata non è eccessivamente complicata da calcolare. Vediamolo all'opera in un problema classico, quello del calcolo della radice n-esima di un numero. L'equazione

$$f(x) = x^n - k = 0, \quad k > 0$$

ha per n intero e per $x > 0$ una sola soluzione reale $\alpha = \sqrt[n]{k}$.
Dalla (2.4) si ha

$$x_{i+1} = \frac{1}{n}\left[(n-1)x_i + \frac{k}{x_i^{n-1}}\right], \quad i = 0, 1, \ldots \tag{2.5}$$

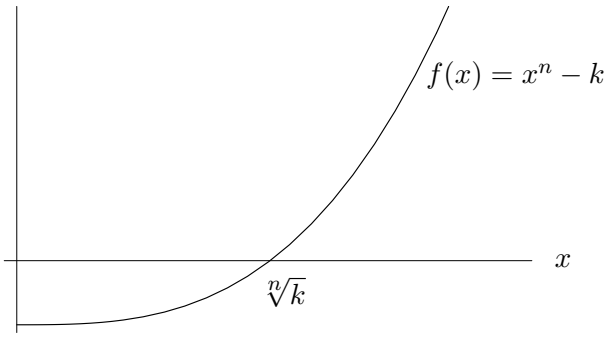


Figura 2.7: Calcolo di $\sqrt[n]{k}$.

Poiché $f'(x)$ e $f''(x)$ non si annullano per $x > 0$ e

$$f''(x) = n(n-1)x^{n-2} > 0, \quad \text{per ogni } x > 0,$$

per x_0 si può scegliere un qualunque punto a destra di α e si ottiene una successione x_i monotona decrescente. Ovviamente non conviene prendere un x_0 molto grande, per evitare di dover fare troppi passi.

Applichiamo la (2.5) ad un caso importante: quello della radice quadrata. Per $n = 2$ la (2.5) diventa

$$x_{i+1} = \frac{1}{2}\left[x_i + \frac{k}{x_i}\right], \quad i = 0, 1, \ldots$$

Questo algoritmo per calcolare radici quadrate è noto come formula di Erone, matematico alessandrino del 2^o secolo. In realtà il metodo era già noto ai babilonesi. È molto efficiente, come possiamo vedere applicandolo, per esempio, al calcolo di $\sqrt{2}$. Scegliendo $x_0 = 2$ si ottiene

i	x_i	$f(x_i)$
1	1.5000	$2.5000\ 10^{-1}$
2	1.4167	$6.9444\ 10^{-3}$
3	1.4142	$6.0073\ 10^{-6}$
4	1.4142	$4.510\ 10^{-12}$

2.6 Metodi iterativi.

Fino a questo punto abbiamo visto tre metodi per approssimare soluzioni di equazioni: il metodo di bisezione, il metodo delle secanti e quello delle tangenti. Il secondo

ci ha fornito una buona approssimazione della soluzione dell'equazione del problema di Archimede più rapidamente del primo, il terzo più rapidamente degli altri due. Per poter dare una spiegazione di questo comportamento dobbiamo fare uno studio più approfondito dei metodi iterativi.

Il metodo delle tangenti espresso dalla (2.4) è un caso particolare del più generale *metodo iterativo* della forma

$$x_{i+1} = g(x_i), \quad i = 0, 1, \ldots \tag{2.6}$$

con cui, a partire da un valore iniziale x_0, è possibile approssimare le soluzioni dell'equazione

$$x = g(x). \tag{2.7}$$

La funzione $g(x)$ è detta *funzione di iterazione* e le soluzioni di (2.7) sono anche dette *punti fissi* di $g(x)$. Nel seguito supporremo per semplicità che $g \in C^1[a, b]$, anche se per applicare il metodo (2.6) basta che la $g(x)$ sia continua. Geometricamente le soluzioni dell'equazione (2.7) corrispondono alle ascisse dei punti di intersezione delle curve definite da $y = x$ e $y = g(x)$. Nel caso della figura 2.8 vi sono tre soluzioni, che sono state indicate con le lettere α, β e γ. Naturalmente se si vuole

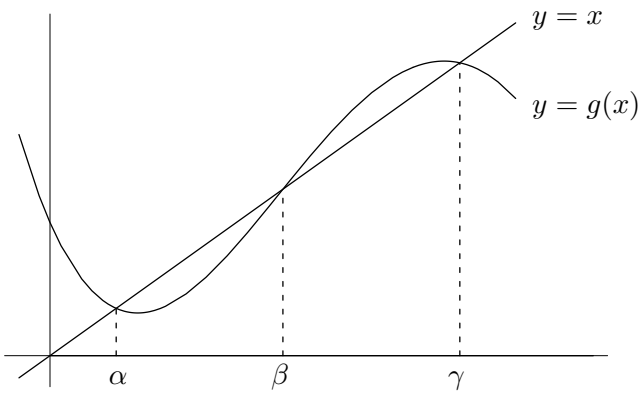


Figura 2.8: Soluzioni dell'equazione $x = g(x)$.

utilizzare un metodo della forma (2.6) per approssimare le soluzioni dell'equazione (2.1), le due equazioni (2.1) e (2.7) devono essere equivalenti. Ad esempio, per il metodo delle tangenti è

$$g(x) = x - \frac{f(x)}{f'(x)}, \tag{2.8}$$

e se $f'(x) \neq 0$ in α, in un intorno di α l'equazione

$$x = x - \frac{f(x)}{f'(x)}$$

è equivalente alla (2.1).

Esempio. L'equazione

$$x^3 - 3x^2 + 1 = 0$$

del problema di Archimede può essere scritta nella forma (2.7) in molti modi. Eccone solo alcuni:

$$(a) \quad x = x^3 - 3x^2 + x + 1,$$

$$(b) \quad x = 3 - \frac{1}{x^2},$$

$$(c) \quad x = \frac{1}{3}\left(x^2 + \frac{1}{x}\right).$$

La prima equazione è equivalente a quella data per ogni x, la seconda e la terza lo sono negli intervalli che non contengono lo zero (notare che 0 non è soluzione dell'equazione data).

Scegliamo come metodo iterativo quello che si ottiene dall'equazione (c). Poiché cerchiamo la soluzione compresa fra 0 e 1, assumiamo come punto iniziale $x_0 = 0.5$. Si ottiene

i	x_i	$x_i - x_{i-1}$
1	0.75000	$2.5000\ 10^{-1}$
2	0.63194	$-1.1806\ 10^{-1}$
3	0.66059	$2.8646\ 10^{-2}$
4	0.65006	$-1.0531\ 10^{-2}$
5	0.65363	$3.5739\ 10^{-3}$
6	0.65238	$-1.2506\ 10^{-3}$
7	0.65282	$4.3317\ 10^{-4}$

È evidente che la successione ottenuta tende ad α. □

2.7 Convergenza di un metodo iterativo.

Se la successione ottenuta con la (2.6) è convergente, posto

$$\alpha = \lim_{i\to\infty} x_i,$$

risulta

$$\alpha = \lim_{i\to\infty} x_{i+1} = \lim_{i\to\infty} g(x_i) = g(\lim_{i\to\infty} x_i) = g(\alpha),$$

quindi α è soluzione dell'equazione (2.7). Naturalmente se la successione ottenuta con il metodo (2.6) non è convergente, i valori della successione non sono di alcuna utilità.

Per ottenere una successione convergente dobbiamo scegliere con cura sia il metodo iterativo sia il punto iniziale. È infatti possibile che un metodo iterativo produca

una successione convergente a partire da un punto iniziale e non convergente a partire da un altro. Così come è possibile che un metodo non produca mai successioni convergenti o produca successioni che convergono a soluzioni diverse da quella che vogliamo calcolare.

Per studiare la convergenza di un metodo iterativo della forma (2.6) è fondamentale il seguente teorema.

Teorema del punto fisso. *Siano α un punto fisso di $g(x)$ e S un intorno circolare chiuso di α tale che*

$$|g'(x)| < 1, \quad per \quad x \in S. \tag{2.9}$$

Se scegliamo un punto $x_0 \in S$, ogni x_i definito dalla (2.6) appartiene ad S e

$$\lim_{i\to\infty} x_i = \alpha.$$

Dim. Sia $\lambda = \max_{x\in S} |g'(x)|$ e ρ il raggio di S. Dalla (2.9) risulta $\lambda < 1$. Si dimostra per induzione che

$$|x_i - \alpha| \leq \lambda^i \rho \leq \rho. \tag{2.10}$$

Per $i = 0$ la (2.10) è vera. Si suppone che la (2.10) sia vera fino all'indice i; per il teorema del valor medio, dalla (2.6) si ha

$$x_{i+1} - \alpha = g(x_i) - g(\alpha) = g'(\xi_i)\ (x_i - \alpha), \quad \xi_i \in S, \tag{2.11}$$

da cui

$$|x_{i+1} - \alpha| = |g'(\xi_i)|\ |x_i - \alpha|.$$

Poiché

$$|g'(\xi_i)| \leq \lambda < 1,$$

per l'ipotesi induttiva è

$$|x_{i+1} - \alpha| \leq \lambda^{i+1} \rho,$$

quindi la (2.10) vale per ogni $i \geq 0$. Dalla (2.10) poi, passando al limite, si ottiene

$$\lim_{i\to\infty} |x_i - \alpha| = 0. \qquad \square$$

Nelle ipotesi del teorema, α è l'unica soluzione della (2.7) in S. Infatti, se per assurdo ce ne fosse un'altra $\beta \neq \alpha$, esisterebbe $\xi \in S$ tale che

$$|\alpha - \beta| = |g(\alpha) - g(\beta)| = |g'(\xi)||\alpha - \beta| < |\alpha - \beta|,$$

che è assurdo.

Del teorema si può dare un'interpretazione geometrica. La figura 2.9 mostra due casi in cui è verificata la condizione (2.9). Nei due casi si ha:

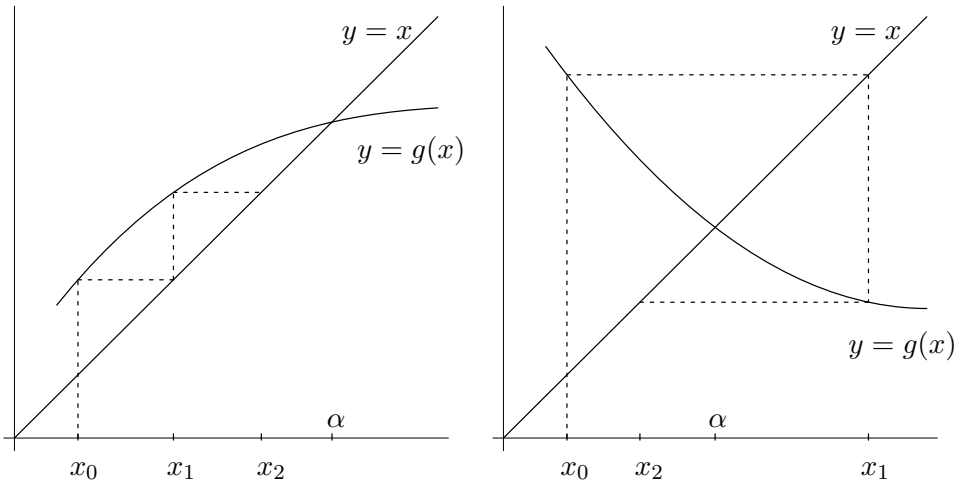


Figura 2.9: Caso $0 \leq g'(\alpha) < 1$ Caso $-1 < g'(\alpha) < 0$.

a) se $0 < g'(x) < 1$, la successione ottenuta è monotona crescente se $x_0 < \alpha$, decrescente se $x_0 > \alpha$;

b) se $-1 < g'(x) < 0$, la successione ottenuta ha elementi alternativamente maggiori e minori di α ed è quindi costituita da due sottosuccessioni monotone, una crescente e l'altra decrescente. Verrà chiamata *successione alternata*.

Poiché α non è noto, non possiamo verificare se la condizione (2.9) vale in un intorno circolare di α, ma dobbiamo limitarci a considerare un intervallo chiuso $[a, b]$ contenente α nella sua parte interna. Le ipotesi del teorema valgono nel massimo intorno circolare chiuso di α contenuto in $[a, b]$, e la convergenza è certamente assicurata se il valore iniziale x_0 è l'estremo dell'intervallo più vicino ad α.

Se nell'intervallo $[a, b]$ è $g'(x) > 0$, poiché in tal caso la successione è monotona, si può scegliere come valore iniziale indifferentemente uno dei due estremi. Inoltre in questo caso è sufficiente che la condizione (8) sia verificata in $[a, \alpha)$, assumendo $x_0 = a$, oppure in $(\alpha, b]$, assumendo $x_0 = b$.

Se nell'intervallo $[a, b]$ è $g'(x) < 0$, la successione risulta alternata e in tal caso, se non è possibile stabilire quale dei due estremi sia il più vicino ad α, si può scegliere come valore iniziale, ad esempio, $x_0 = a$: se $x_1 \in [a, b]$, la successione converge, se invece $x_1 \notin [a, b]$, basta scegliere $x_0 = b$, per ottenere certamente una successione convergente.

Se nell'intervallo $[a, b]$ la (2.9) non vale, cioè $|g'(x)|$ è maggiore di 1 in qualche punto di $[a, b]$, occorre valutare se è possibile restringere l'intervallo in modo da ottenerne uno in cui la (2.9) valga, oppure se questo non è possibile perché $|g'(\alpha)| \geq 1$. Nella figura 2.10 sono mostrati due casi in cui è $|g'(\alpha)| > 1$ e non vi è convergenza.

Le diverse situazioni che si possono verificare sono bene illustrate dalla terminologia che si usa: se $|g'(\alpha)| < 1$ il punto α viene detto *attrattivo* per la funzione

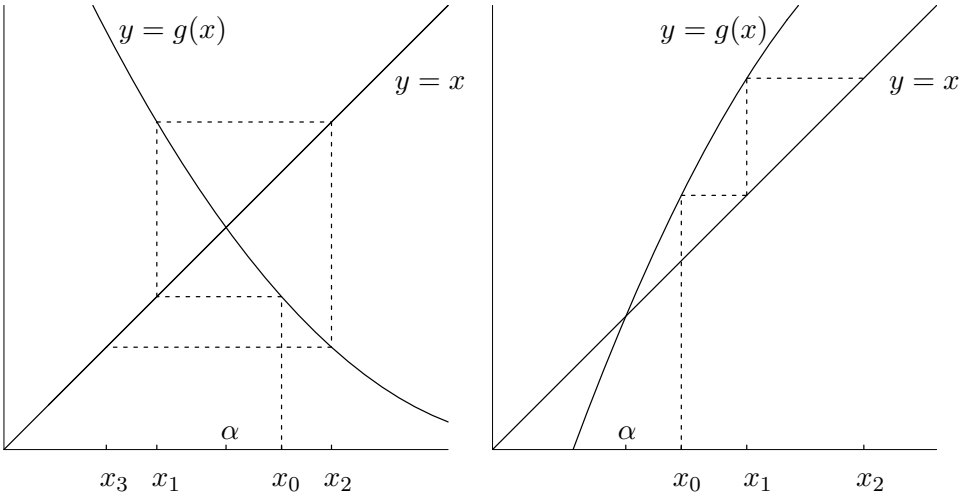


Figura 2.10: Caso $g'(\alpha) < -1$ Caso $g'(\alpha) > 1$.

$g(x)$ e il metodo iterativo (2.6) viene detto *(localmente) convergente*: infatti esiste un intorno circolare S di α tale che tutte le successioni costruite con il metodo a partire da punti di S convergono ad α. Se invece $|g'(\alpha)| > 1$, il punto α viene detto *repulsivo*.

Esempio. Dei tre metodi iterativi considerati nel paragrafo 2.6 per risolvere l'equazione

$$x^3 - 3x^2 + 1 = 0,$$

solo l'ultimo è localmente convergente. Infatti si ha per $0.5 < x < 1$

$$(a) \quad g(x) = x^3 - 3x^2 + x + 1, \quad g'(x) = 3x^2 - 6x + 1, \quad g'(x) < -1;$$

$$(b) \quad g(x) = 3 - \frac{1}{x^2}, \quad g'(x) = \frac{2}{x^3}, \quad g'(x) > 1;$$

$$(c) \quad g(x) = \frac{1}{3}\left(x^2 + \frac{1}{x}\right), \quad g'(x) = \frac{1}{3}\left(2x - \frac{1}{x^2}\right), \quad -1 < g'(x) < 1.$$

Quindi l'ultimo metodo è convergente in ogni intorno circolare chiuso di α contenuto in $(1/2, 1)$. Si può anche scegliere come x_0 un qualunque punto di $[1/2, 1]$, purché x_1 appartenga ancora all'intervallo. Il segno di $g'(x)$ spiega l'andamento della successione ottenuta: inizialmente la successione risulta decrescente perché vicino ad 1 $g'(x)$ è positiva, poi diventa alternata perché $g'(x)$ è negativa in un piccolo intorno di α. □

Il metodo iterativo (2.6) è convergente anche in ipotesi più deboli di quelle richieste dal teorema del punto fisso. Si può ad esempio dimostrare che la successione (2.6) è convergente anche se la condizione (2.9) vale in tutto l'intorno S eccetto il punto α (per continuità deve essere $|g'(\alpha)| = 1$). Si noti però che in tal caso la successione converge molto lentamente.

Esempio. L'equazione

$$x = \sin x$$

ha la soluzione $\alpha = 0$. Il metodo iterativo

$$x_{i+1} = g(x_i) = \sin x_i, \quad i = 0, 1, \ldots,$$

è convergente per ogni x_0 appartenente ad un intorno circolare di α di raggio minore o uguale a $\pi/2$. Si ha infatti $|g'(x)| < 1$ per $x \in [-\pi/2, \pi/2]$ e $x \neq \alpha$. Ponendo $x_0 = 1$, la successione $\{x_i\}$ converge molto lentamente.

i	x_i	$x_i - x_{i-1}$
1	0.84147	$-1.5853\ 10^{-1}$
2	0.74562	$-9.5847\ 10^{-2}$
3	0.67843	$-6.7194\ 10^{-2}$
4	0.62757	$-5.0859\ 10^{-2}$
.	.	.
.	.	.
86	0.18143	$-1.0104\ 10^{-3}$
87	0.18044	$-9.9377\ 10^{-4}$

□

Successioni che convergono così lentamente non sono di alcuna utilità pratica. Per questa ragione non vengono in realtà mai usati metodi iterativi in cui $|g'(\alpha)|$ sia uguale a 1 o molto vicino a 1.

2.8 Criteri di arresto

Il criterio più comunemente usato per decidere a quale iterazione arrestare il metodo (2.6) è il (2.3), che però non sempre garantisce che la soluzione sia effettivamente approssimata con un errore assoluto minore di ϵ. Infatti dalla (2.11) per ogni i si ha

$$x_{i+1} - x_i = (x_{i+1} - \alpha) - (x_i - \alpha) = (x_i - \alpha)(g'(\xi_i) - 1), \quad \xi_i \in S$$

e quindi

$$|x_i - \alpha| = \frac{|x_{i+1} - x_i|}{|g'(\xi_i) - 1|} < \frac{\epsilon}{|g'(\xi_i) - 1|}\,.$$

Perciò per un dato ϵ, l'errore può risultare tanto più grande quanto più $g'(x)$ è vicino a 1 per x vicino ad α. Una stima un po' grossolana di $g'(\alpha)$ si può dare esaminando la successione delle differenze $x_{i+1} - x_i$. Si ha infatti che

$$\frac{x_{i+1} - \alpha}{x_i - \alpha} = g'(\xi) \quad \text{e} \quad x_{i+1} - x_i = (x_i - \alpha)(g'(\xi) - 1),$$

quindi se x_i è sufficientemente vicino ad α e se $g'(x)$ non varia molto nell'intorno di α in cui stiamo lavorando, si ha

$$\frac{x_{i+2} - x_{i+1}}{x_{i+1} - x_i} \sim \frac{(x_{i+1} - \alpha)(g'(\alpha) - 1)}{(x_i - \alpha)(g'(\alpha) - 1)} \sim g'(\alpha). \tag{2.12}$$

Se risulta che $g'(\alpha)$ è vicino ad 1, probabilmente occorreranno molte iterazioni prima che la (2.3) sia verificata, e comunque la soluzione sarà ancora lontana.

Esempio. Nel caso della successione calcolata nell'esempio del paragrafo 2.6 la condizione (2.3) risulta verificata per $\epsilon = 10^{-3}$ se $i = 7$ e si ha $x_7 = 0.65282$. Poiché $g'(x)$ in un intorno di x_7 è negativa, la successione è alternata e il valore di x_7 ottenuto approssima α con un errore assoluto minore di ϵ.

Se per risolvere la stessa equazione si usa invece il metodo

$$x_{i+1} = g(x_i), \quad \text{con} \quad g(x) = \frac{1}{30}\left(x^2 + \frac{1}{x}\right) + \frac{9}{10}\,x,$$

perché la (2.3) sia verificata con lo stesso ϵ occorrono 21 iterazioni. Notiamo che la successione converge lentamente, e infatti dalla stima (2.12) risulta che la successione dei rapporti delle differenze è crescente e alla 21−esima iterazione supera 0.86. Dalla (1.11) segue perciò che x_{21} è affetto da un errore superiore ad ϵ. Nella fattispecie è $|x_{21} - \alpha| \sim 0.6\ 10^{-2}$. □

Al posto della (2.3), con cui si controlla l'errore assoluto dell'approssimazione, si possono utilizzare altre condizioni di arresto. Per esempio si può usare la condizione

$$|x_{i+1} - x_i| < \epsilon\ \min(|x_i|, |x_{i+1}|), \quad \text{se} \quad |x_i|, |x_{i+1}| \neq 0,$$

con cui si controlla l'errore relativo, o la condizione

$$|f(x_i)| < \epsilon, \tag{2.13}$$

con cui si controlla l'approssimazione della soluzione attraverso i valori della funzione $f(x)$. Se si usa la (2.13), poiché per ogni i esiste un η tale che

$$f(x_i) = f(x_i) - f(\alpha) = f'(\eta)(x_i - \alpha), \quad |\eta - \alpha| < |x_i - \alpha|,$$

si ha

$$|x_i - \alpha| = \frac{|f(x_i)|}{|f'(\eta)|} < \frac{\epsilon}{|f'(\eta)|}\,.$$

Quindi, a parità di valori di ϵ, l'errore assoluto può risultare tanto più grande quanto più piccolo è $|f'(\eta)|$.

Oltre ai criteri descritti, è necessario prevedere un numero massimo di iterazioni consentite, in modo che l'esecuzione di un programma che implementa un metodo iterativo non impieghi eccessive risorse di calcolo nei casi di convergenza lenta.

A causa degli errori di arrotondamento che si commettono nel calcolo di $g(x_i)$ la successione effettivamente calcolata $\widetilde{x}_i$ può non essere convergente anche quando sono soddisfatte le ipotesi del teorema del punto fisso. Tuttavia è possibile dimostrare che i valori $\widetilde{x}_i$ effettivamente calcolati appartengono ad intorni di α con raggio via via decrescente.

Sia δ_i l'errore assoluto introdotto nel calcolo effettivo alla i-esima iterazione:

$$\widetilde{x}_i = g(\widetilde{x}_{i-1}) + \delta_i,$$

tale che

$$|\delta_i| \leq \delta, \quad \text{per } i = 0, 1, \ldots$$

Nelle ipotesi del teorema del punto fisso, posto

$$\lambda = \max_{x \in S} |g'(x)| < 1 \quad \text{e} \quad \sigma = \frac{\delta}{1-\lambda},$$

se $\sigma < \rho$, $x_0 \in S$ e $\widetilde{x}_0 = x_0$, si può dimostrare che

$$|\widetilde{x}_i - \alpha| \leq \sigma + \lambda^i(\rho - \sigma), \quad \text{per } i = 0, 1, \ldots$$

Quindi i valori $\widetilde{x}_i$ effettivamente calcolati possono non avvicinarsi arbitrariamente ad α, ma da un certo indice i in poi approssimano α con un errore assoluto non superiore a σ.

2.9 Ordine di convergenza.

Per confrontare metodi iterativi diversi che approssimano la stessa soluzione α di una equazione, se ne deve valutare la velocità di convergenza. Lo studio della velocità di convergenza viene generalmente ricondotto a quello dell'ordine di convergenza del metodo.

Sia x_i una successione convergente ad α e sia $x_i \neq \alpha$ per ogni i. Se esiste un numero reale $p \geq 1$, tale che

$$\lim_{i \to \infty} \frac{|x_{i+1} - \alpha|}{|x_i - \alpha|^p} = \gamma, \quad \text{con} \begin{cases} 0 < \gamma < 1 & \text{se } p = 1, \\ \gamma > 0 & \text{se } p > 1, \end{cases} \tag{2.14}$$

si dice che la successione ha *ordine di convergenza* p. La costante γ è detta *fattore di convergenza*.

Se $p = 1$, si dice anche che la convergenza è *lineare*,

se $p > 1$, si dice anche che la convergenza è *superlineare*.

Se nel limite (2.14) è $\gamma = 1$ quando $p = 1$, si dice che la convergenza è *sublineare*.

Un *metodo iterativo* convergente ad α si dice di *ordine* p se tutte le successioni ottenute al variare del punto iniziale in un intorno di α convergono con ordine di convergenza p.

Nelle ipotesi del teorema del punto fisso, per la (2.11) esiste ξ_i tale che

$$\frac{x_{i+1}-\alpha}{x_i-\alpha} = g'(\xi_i), \quad |\xi-\alpha| < |x_i-\alpha|,$$

e passando al limite si ha

$$\lim_{i\to\infty} \frac{|x_{i+1}-\alpha|}{|x_i-\alpha|} = |g'(\alpha)|.$$

Ne segue che se $0 < |g'(\alpha)| < 1$, il metodo (2.6) ha convergenza lineare.

Nel caso $|g'(\alpha)| = 1$, il metodo, se fosse convergente, avrebbe convergenza sublineare. Se $g'(\alpha) = 0$, l'ordine è legato a quello della prima derivata della $g(x)$ che non si annulla in α, come risulta dal seguente teorema.

Teorema. *Se in un intorno S di α è $g(x) \in C^p[S]$, con $p \geq 2$ intero, e*

$$g'(\alpha) = g''(\alpha) = \ldots = g^{(p-1)}(\alpha) = 0, \; g^{(p)}(\alpha) \neq 0,$$

il metodo (2.6) ha ordine di convergenza p.

Dim. Dalla formula di Taylor si ha

$$x_{i+1}-\alpha = g(x_i)-g(\alpha) = \frac{(x_i-\alpha)^p}{p!}\, g^{(p)}(\xi), \quad |\xi-\alpha| < |x_i-\alpha|,$$

da cui

$$\lim_{i\to\infty} \frac{|x_{i+1}-\alpha|}{|x_i-\alpha|^p} = \frac{1}{p!}\, |g^{(p)}(\alpha)| \neq 0,$$

e quindi la successione ha ordine di convergenza p. □

Si osservi che l'ordine di convergenza p può essere anche un numero non intero: in tal caso, posto $q = \lfloor p \rfloor$, se $g(x) \in C^q[a,b]$, si ha che

$$g'(\alpha) = g''(\alpha) = \ldots = g^{(q)}(\alpha) = 0,$$

e che la funzione $g(x)$ non ha la derivata $(q+1)$-esima continua in α.

Esempio. L'equazione

$$x = \sqrt{|x|^3}$$

ha soluzione $\alpha = 0$. La successione $\{x_i\}$ definita da

$$x_{i+1} = \sqrt{|x_i|^3}$$

è tale che $|x_{i+1}| < |x_i|$ per $|x_i| < 1$ e quindi è convergente per $x_0 \in (-1,1)$. Inoltre si ha

$$\frac{|x_{i+1}|}{|x_i|^{3/2}} = 1.$$

Perciò l'ordine di convergenza è $p = 3/2$. □

Dalla (1.13) segue che per un metodo lineare o superlineare esiste una costante $\rho > 0$ tale che, per i sufficientemente grande, vale la relazione

$$|x_{i+1} - \alpha| \leq \rho |x_i - \alpha|^p, \tag{16}$$

con $\rho < 1$ se $p = 1$. Questa relazione misura la riduzione ad ogni iterazione dell'errore assoluto che si commette approssimando α con x_{i+1}. È evidente che tale riduzione è tanto più elevata quanto maggiore è l'ordine di convergenza, e, a parità di ordine di convergenza, quanto minore è il fattore di convergenza. Se il metodo è sublineare non vale alcuna relazione della forma (1.14): la riduzione dell'errore rallenta quanto più ci si avvicina ad α. Ne è un esempio il metodo iterativo $x_{i+1} = \sin x_i$, studiato nel par. 2.7.

2.10 Ordine del metodo delle tangenti.

Determinare l'ordine del metodo delle tangenti è facile se

$$f \in C^2[a,b], \qquad f'(\alpha) \neq 0, \qquad f''(\alpha) \neq 0. \tag{2.15}$$

Si ha infatti dalla formula di Taylor

$$\begin{aligned} f(x) &= f'(\alpha)(x-\alpha) + f''(\xi)\,\tfrac{(x-\alpha)^2}{2}, \quad |\xi - \alpha| < |x - \alpha|, \\ f'(x) &= f'(\alpha) + f''(\eta)(x-\alpha), \quad |\eta - \alpha| < |x - \alpha|. \end{aligned}$$

Dalla (2.8) si ha

$$g(x) - \alpha = \frac{(x-\alpha)f'(x) - f(x)}{f'(x)} = \frac{(x-\alpha)^2}{f'(x)} \left[f''(\eta) - \frac{f''(\xi)}{2} \right],$$

e quindi

$$\frac{g(x) - \alpha}{(x-\alpha)^2} = \frac{2f''(\eta) - f''(\xi)}{2f'(x)}.$$

Passando al limite si ha $\lim_{x\to\alpha} f'(x) = f'(\alpha)$ e $\lim_{x\to\alpha} f''(\xi) = \lim_{x\to\alpha} f''(\eta) = f''(\alpha)$, per cui

$$\lim_{i\to\infty} \frac{x_{i+1} - \alpha}{(x_i - \alpha)^2} = \lim_{i\to\infty} \frac{g(x_i) - \alpha}{(x_i - \alpha)^2} = \frac{f''(\alpha)}{2f'(\alpha)},$$

e questo limite è finito e diverso da zero. Quindi il metodo delle tangenti nelle ipotesi (2.15) ha ordine 2.

In ipotesi diverse dalle (2.15) la determinazione dell'ordine del metodo delle tangenti è un po' più complicata. Intanto bisogna distinguere il caso in cui α è soluzione *multipla* dal caso in cui α è soluzione *semplice*.

Se per un intero $r > 0$ è $f(x) \in C^r[a,b]$, una soluzione α si dice di *molteplicità* r se esiste finito e non nullo il

$$\lim_{x\to\alpha} \frac{f(x)}{(x-\alpha)^r}.$$

In tal caso risulta

$$f(\alpha) = f'(\alpha) = \ldots = f^{(r-1)}(\alpha) = 0, \quad f^{(r)}(\alpha) \neq 0.$$

Vale allora il seguente teorema.

Teorema. *Sia $\alpha \in [a,b]$ soluzione di $f(x) = 0$, e sia $f'(x) \neq 0$ per $x \in [a,b] - \{\alpha\}$.*
a) Se α ha molteplicità 1 e $f(x) \in C^2[a,b]$, allora il metodo delle tangenti è convergente con ordine almeno 2. In particolare l'ordine è 2 se $f''(\alpha) \neq 0$.
b) Se α ha molteplicità finita $r \geq 2$ e se $f(x) \in C^r[a,b]$, allora il metodo delle tangenti ha convergenza lineare. □

Esempio. Per calcolare $\alpha = \sqrt[3]{2}$ si può applicare il metodo delle tangenti direttamente all'equazione

$$f(x) = x^3 - 2 = 0,$$

come si è fatto più in generale nel par. 2.5. Poiché $f'(\alpha) \neq 0$ e $f''(\alpha) \neq 0$, il metodo che si ottiene

$$x_{i+1} = \frac{2}{3}\left(x_i + \frac{1}{x_i^2}\right), \tag{2.16}$$

risulta di ordine 2. Inoltre, poiché

$$f(x) = x^3 - \alpha^3 = (x - \alpha)\,(x^2 + \alpha x + \alpha^2),$$

per x_i vicino ad α è

$$f(x_i) \sim (x_i - \alpha)\, 3\alpha^2,$$

per cui le due successioni $|f(x_i)|$ e $|x_i - \alpha|$ hanno lo stesso comportamento per $x_i \to \alpha$. Quindi per monitorare il comportamento di $|x_i - \alpha|$ si può esaminare quello di $|f(x_i)|$. A questo scopo si applica il metodo delle tangenti utilizzando una precisione di calcolo maggiore di quella usata negli esempi precedenti e specificando una tolleranza molto minore.

i	x_i	$f(x_i)$
1	1.50000000000000	1.37500000000000
2	1.29629629629630	$1.78275669359345\ 10^{-1}$
3	1.26093222474175	$4.81928579256641\ 10^{-3}$
4	1.25992186056593	$3.86058273083023\ 10^{-6}$
5	1.25992104989539	$2.48379095069140\ 10^{-12}$

Esaminando la terza colonna, è evidente che gli errori $|x_i - \alpha|$ decrescono in modo quadratico.

Lo stesso α è anche soluzione dell'equazione

$$f_1(x) = x^2 - \frac{2}{x} = 0.$$

In questo caso risulta $f_1'(\alpha) \neq 0$ e $f_1''(\alpha) = 0$, perciò il metodo delle tangenti applicato alla f_1

$$x_{i+1} = \frac{x_i}{2}\,\frac{x_i^3 + 4}{x_i^3 + 1}$$

è di ordine superiore a 2 (è facile verificare che è del terzo ordine). Si ottiene

i	x_i	$f(x_i)$
1	1.33333333333333	2.77777777777778 10^{-1}
2	1.26007326007326	5.75318426072169 10^{-4}
3	1.25992104989635	5.59641222253049 10^{-12}
4	1.25992104989487	2.22044604925031 10^{-16}

Esaminando la terza colonna, è evidente che gli errori $|x_i - \alpha|$ decrescono in modo cubico fino al limite della precisione di macchina. In questo secondo caso sono state richieste meno iterazioni per ottenere l'accuratezza richiesta. Si deve però tenere presente che ogni iterazione ha un costo maggiore che nel primo caso. □

2.11 Ordine dei metodi delle corde e delle secanti.

Per il metodo delle corde è

$$g(x) = x - \frac{f(x)}{m}\,.$$

Se $f \in C^1[S]$, dal teorema del punto fisso segue che la successione x_i ottenuta dal metodo è convergente se

$$|g'(x)| = \left|1 - \frac{f'(x)}{m}\right| < 1$$

in tutto un intorno circolare della soluzione α in cui si sceglie il punto x_0. Si ottengono quindi le seguenti condizioni sufficienti per la convergenza

$$f'(x) \neq 0 \quad \text{e} \quad mf'(x) > 0 \quad \text{per} \quad x \in S,$$
$$\text{e} \quad |m| > \tfrac{1}{2}\ \max_{x\in S} |f'(x)|.$$

Per quanto riguarda l'ordine, risulta che se $m \neq f'(\alpha)$ è $g'(\alpha) \neq 0$, per cui il metodo delle corde è del primo ordine. Se invece $m = f'(\alpha)$ e $f \in C^2[S]$, il metodo è almeno del secondo ordine.

Per trovare l'ordine di convergenza del metodo delle secanti, consideriamo il caso in cui da un certo indice in poi uno dei due estremi resti invariato, cioè il caso in cui il metodo assume la forma (2.6)

$$x_{i+1} = x_i - \frac{f(x_i)(x_i - c)}{f(x_i) - f(c)},$$

dove c sta ad indicare l'estremo che resta invariato. La funzione di iterazione del metodo risulta quindi

$$g(x) = x - \frac{f(x)(x-c)}{f(x)-f(c)},$$

ed è

$$g'(x) = f(c)\,\frac{f'(x)(x-c)-f(x)+f(c)}{[f(x)-f(c)]^2},$$

da cui

$$g'(\alpha) = 1 + \frac{f'(\alpha)}{f(c)}\,(\alpha - c).$$

Poiché in generale questa quantità è diversa da zero, il metodo delle secanti è del primo ordine.

Vi è una variante del metodo delle secanti che consiste nel considerare all'i-esima iterazione la retta che passa per i punti $(x_i, f(x_i))$ e $(x_{i-1}, f(x_{i-1}))$. Il metodo che ne risulta è il seguente:

$$x_{i+1} = x_i - \frac{f(x_i)(x_i - x_{i-1})}{f(x_i) - f(x_{i-1})}.$$

Poiché questo metodo non appartiene alla classe dei metodi individuati dalla (2.6), i teoremi che abbiamo visto non possono essere applicati. È possibile comunque dimostrare che il metodo è convergente se x_0 è abbastanza vicino alla soluzione e che l'ordine di convergenza è

$$p = \frac{1+\sqrt{5}}{2} \sim 1.618 \quad \text{(sezione aurea)}.$$

Capitolo 3

Elementi di algebra lineare

3.1 Vettori.

In matematica si incontrano spesso insiemi di elementi su cui sono definite delle operazioni che godono di particolari proprietà. Uno dei più importanti è lo spazio vettoriale di cui andiamo adesso a studiare l'esempio più classico.

Sia $\mathbf{R}$ l'insieme dei numeri reali; gli elementi di $\mathbf{R}$ sono anche detti *scalari*. Fissato un intero $n \geq 1$, si chiama $\mathbf{R}^n$ l'insieme delle n-uple $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, con $x_i \in \boldsymbol{R}$ per $i = 1, \ldots, n$. Le x_i sono dette *componenti*. Gli elementi di $\boldsymbol{R}^n$ possono essere visti come generalizzazione delle coppie ordinate di numeri reali con cui si rappresentano i punti del piano cartesiano. Ad esempio in $\boldsymbol{R}^3$ si rappresentano i punti dello spazio.

Su $\boldsymbol{R}^n$ si definiscono le operazioni di

- addizione: $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{y}$, con $\boldsymbol{x},\ \boldsymbol{y} \in \boldsymbol{R}^n$, è la n-upla le cui componenti sono

$$z_i = x_i + y_i, \quad \text{per} \quad i = 1, \ldots, n,$$

- moltiplicazione per scalare: $\boldsymbol{y} = \alpha\boldsymbol{x}$, con $\boldsymbol{x} \in \boldsymbol{R}^n$ e $\alpha \in \boldsymbol{R}$, è la n-upla le cui componenti sono

$$y_i = \alpha x_i \quad \text{per} \quad i = 1, \ldots, n.$$

Queste operazioni verificano alcune proprietà fondamentali. Ad esempio, l'addizione è commutativa e associativa, esiste l'elemento neutro dell'addizione, esiste l'elemento opposto, la moltiplicazione per scalare è distributiva, esiste l'elemento neutro della moltiplicazione per scalare. Un insieme così fatto viene chiamato *spazio vettoriale* e i suoi elementi sono detti *vettori*. Noi useremo la convenzione di scrivere i vettori per *colonna*, in questo modo

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

e indicheremo con $\boldsymbol{x}^T$ il vettore *riga*, cioè il vettore le cui componenti sono scritte in orizzontale, in questo modo

$$\boldsymbol{x}^T = [x_1\,,\, x_2\,,\, \ldots\,,\, x_n].$$

Il *prodotto scalare* di due vettori $\boldsymbol{x}$ e $\boldsymbol{y} \in \boldsymbol{R}^n$ è definito da

$$\boldsymbol{x}^T\boldsymbol{y} = [x_1, \ldots, x_n] \begin{bmatrix} y_1 \\ \ldots \\ y_n \end{bmatrix} = \sum_{i=1}^{n} x_i\, y_i.$$

Il prodotto scalare gode delle seguenti proprietà:

1. $\boldsymbol{x}^T\boldsymbol{x} \geq 0$, ed è nullo se e solo se $\boldsymbol{x} = \boldsymbol{0}$;
2. $\boldsymbol{x}^T\boldsymbol{y} = \boldsymbol{y}^T\boldsymbol{x}$;
3. $\boldsymbol{x}^T(\alpha\boldsymbol{y}) = \alpha\, \boldsymbol{x}^T\boldsymbol{y}$ per $\alpha \in \boldsymbol{R}$;
4. $\boldsymbol{x}^T(\boldsymbol{y} + \boldsymbol{z}) = \boldsymbol{x}^T\boldsymbol{y} + \boldsymbol{x}^T\boldsymbol{z}$ per $\boldsymbol{z} \in \boldsymbol{R}^n$.

Vale inoltre la disuguaglianza di *Cauchy-Schwarz*

$$(\boldsymbol{x}^T\boldsymbol{y})^2 \leq (\boldsymbol{x}^T\boldsymbol{x})\ (\boldsymbol{y}^T\boldsymbol{y}).$$

Se $\boldsymbol{x}^T\boldsymbol{y} = 0$, i due vettori $\boldsymbol{x}$ e $\boldsymbol{y}$ sono detti *ortogonali*.

Dati k vettori $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ di $\boldsymbol{R}^n$ ed k scalari $\alpha_1, \ldots, \alpha_k$, con $\alpha_i \in \boldsymbol{R}$, il vettore

$$\alpha_1\, \boldsymbol{v}_1 + \cdots + \alpha_k\, \boldsymbol{v}_k$$

è detto *combinazione lineare* di $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$. Gli α_i sono i *coefficienti* della combinazione.

Si dice che i k vettori non nulli $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ sono *linearmente dipendenti* se esiste una loro combinazione lineare nulla con coefficienti non tutti nulli. In caso contrario i vettori sono detti *linearmente indipendenti*.

Si dice che i k vettori $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ sono *ortogonali* se

$$\boldsymbol{x}_i^T\boldsymbol{x}_j = \delta_{ij}, \quad \text{dove} \quad \delta_{ij} = \begin{cases} 1 & \text{per } i = j, \\ 0 & \text{per } i \neq j, \end{cases}$$

e sono *ortonormali* se sono ortogonali ed inoltre $\boldsymbol{x}_i^T\boldsymbol{x}_i = 1$, cioè se sono *normalizzati*. Si osservi che k vettori ortogonali sono anche linearmente indipendenti.

Esempio. I vettori

$$\boldsymbol{x} = [1,\, 1,\, -1]^T \qquad \text{e} \qquad \boldsymbol{y} = [1,\, 1,\, 1]^T,$$

sono linearmente indipendenti, ma non ortogonali: infatti $\boldsymbol{x}^T\boldsymbol{y} = 1 \neq 0$. I vettori

$$\boldsymbol{u} = \frac{1}{\sqrt{3}}\, \boldsymbol{x} \qquad \text{e} \qquad \boldsymbol{v} = \frac{1}{\sqrt{8}}\, [-2,\, 2,\, 0]^T$$

sono ortonormali: infatti $\boldsymbol{u}^T\boldsymbol{u} = 1$, $\boldsymbol{v}^T\boldsymbol{v} = 1$ e $\boldsymbol{u}^T\boldsymbol{v} = 0$. □

Trovare se k vettori assegnati sono linearmente indipendenti oppure no può essere complicato se n è grande. Il metodo più semplice per farlo è quello di scrivere i vettori come colonne di una matrice, di cui poi si calcola il rango (come vedremo nel prossimo paragrafo).

n vettori linearmente indipendenti $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \boldsymbol{R}^n$ costituiscono una *base* di $\boldsymbol{R}^n$: ogni vettore $\boldsymbol{x} \in \boldsymbol{R}^n$ può essere espresso come combinazione lineare dei vettori della base

$$\boldsymbol{x} = \sum_{i=1}^{n} \alpha_i \, \boldsymbol{v}_i.$$

Una base viene detta *ortonormale* se i vettori $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ sono ortonormali. Fra le basi ortonormali, particolarmente importante è la cosiddetta *base canonica*, formata dai vettori $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$, dove $\boldsymbol{e}_i$ è il vettore di n componenti nulle, eccetto la i-esima che è uguale a 1. Infatti si può sempre scrivere

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = x_1\boldsymbol{e}_1 + x_2\boldsymbol{e}_2 + \cdots + x_n\boldsymbol{e}_n.$$

3.2 Matrici.

Con $\boldsymbol{R}^{n\times n}$ si indica l'insieme delle *matrici* quadrate di ordine n, cioè delle tabelle di numeri reali disposti su n righe ed n colonne

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}.$$

Gli elementi a_{ij} tali che $i = j$ vengono detti elementi *diagonali* o *principali* di A e formano la *diagonale principale* di A.

Una matrice $A \in \boldsymbol{R}^{n\times n}$ è:

- *diagonale* se $a_{ij} = 0$ per $i \neq j$;
- *scalare* se è diagonale e $a_{ii} = \alpha \in \boldsymbol{R}$;
- *triangolare superiore (inferiore)* se $a_{ij} = 0$ per $i > j$ (per $i < j$);
- *triangolare superiore (inferiore) in senso stretto* se $a_{ij} = 0$ per $i \geq j$ (per $i \leq j$);
- *tridiagonale* se $a_{ij} = 0$ per $|i - j| > 1$.

Si definiscono le seguenti operazioni fra matrici:

- *addizione di matrici*: $C = A + B$ è la matrice i cui elementi sono

$$c_{ij} = a_{ij} + b_{ij}, \quad \text{per} \quad i, j = 1, \ldots, n,$$

- *moltiplicazione di matrice per scalare*: $B = \alpha\, A$ è la matrice i cui elementi sono

$$b_{ij} = \alpha\, a_{ij}, \quad \text{per} \quad i, j = 1, \ldots, n.$$

Valgono le proprietà di associatività e commutatività per l'addizione e di distributività della moltiplicazione per scalare rispetto all'addizione. L'elemento *neutro* per l'addizione è la matrice con tutti gli elementi nulli, che viene indicata con O_n o semplicemente con O se dal contesto risulta chiaramente qual è l'ordine.

Si definisce *prodotto (righe per colonne)* di due matrici la matrice $C = A\,B$, i cui elementi sono

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}, \quad \text{per} \quad i, j = 1, \ldots, n.$$

La moltiplicazione fra matrici gode della proprietà associativa, di quella distributiva rispetto all'addizione, ma non di quella commutativa, cioè in generale

$$A\,B \neq B\,A.$$

La matrice scalare di ordine n avente gli elementi principali uguali a 1 è detta matrice *identica* e viene indicata con I_n o semplicemente con I se dal contesto risulta chiaramente qual è l'ordine. Tale matrice verifica le relazioni

$$IA = AI = A \quad \text{per ogni matrice} \quad A \in \boldsymbol{R}^{n\times n},$$

ed è quindi l'elemento neutro della moltiplicazione.

Data una matrice $A \in \boldsymbol{R}^{n\times n}$, si definisce matrice *trasposta* di A la matrice $B = A^T$ i cui elementi sono

$$b_{ij} = a_{ji}, \quad \text{per} \quad i, j = 1, \ldots, n.$$

Ad esempio

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \quad A^T = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix}.$$

Valgono le proprietà

$$(A^T)^T = A, \quad (A + B)^T = A^T + B^T, \quad (A\,B)^T = B^T A^T.$$

Una matrice si dice

- *simmetrica* se $A^T = A$;
- *ortogonale* se $A^T A = A\,A^T = I$.

Ad esempio la matrice

$$G = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}, \qquad \phi \in \boldsymbol{R},$$

è ortogonale. Infatti

$$G\,G^T = G^T G = \begin{bmatrix} \sin^2\phi + \cos^2\phi & 0 \\ 0 & \sin^2\phi + \cos^2\phi \end{bmatrix} = I.$$

Le colonne di una matrice ortogonale formano una base ortonormale di $\boldsymbol{R}^n$.

Si definisce *rango* di A il numero di colonne linearmente indipendenti di A. Vale la proprietà

$$\text{rango di } A = \text{rango di } A^T,$$

per cui il rango di A è anche uguale al numero di righe linearmente indipendenti di A.

Finora abbiamo considerato solo matrici quadrate, ma è possibile anche considerare matrici con m righe e n colonne, con $m \neq n$. In particolare anche i vettori di $\boldsymbol{R}^n$ possono essere visti come matrici di una sola colonna. È così possibile definire il

– prodotto di matrice per vettore: $\boldsymbol{y} = A\boldsymbol{x}$ è il vettore i cui elementi sono

$$y_i = \sum_{j=1}^{n} a_{ij} x_j, \quad \text{per} \quad i = 1, \ldots, m,$$

– prodotto esterno di vettori: $A = \boldsymbol{x}\,\boldsymbol{y}^T$ è la matrice i cui elementi sono

$$a_{ij} = x_i y_j, \quad \text{per} \quad i, j = 1, \ldots, n.$$

È facile verificare che la matrice $\boldsymbol{x}\,\boldsymbol{y}^T$, se è diversa da zero, ha rango 1. Infatti tutte le sue colonne sono proporzionali, per cui una sola colonna, ad esempio la prima, è linearmente indipendente.

Ad una matrice quadrata A si può associare un opportuno numero reale, detto *determinante*. Il concetto di determinante può essere introdotto in modo assiomatico, attraverso le proprietà che lo caratterizzano oppure, in modo più semplice, per mezzo della seguente *regola di Laplace*: per ogni a_{ij} si considera la sottomatrice A_{ij} di A ottenuta cancellando la i-esima riga e la j-esima colonna. Fissato un qualunque indice di riga i, si definisce il *determinante* di A

$$\det A = \begin{cases} a_{11} \quad \text{se } n = 1, \\ \displaystyle\sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det A_{ij} \quad \text{se } n > 1. \end{cases} \tag{3.1}$$

La quantità $c_{ij} = (-1)^{i+j} a_{ij} \det A_{ij}$ è detta *cofattore* di a_{ij}.

Si può dimostrare che la quantità definita in (3.1) non dipende dal particolare indice i scelto, per cui la (3.1) è in effetti una buona definizione. Per chiarire meglio come si calcola un determinante vediamo qualche caso particolare.

Per $n = 2$ il determinante risulta

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \text{e} \quad \det A = a_{11}a_{22} - a_{12}a_{21}.$$

Per $n > 2$ conviene sviluppare il determinante scegliendo una riga che contenga quanti più zeri possibile. Ad esempio, sia

$$A = \begin{bmatrix} 3 & 1 & 1 & 0 \\ 2 & -1 & 3 & 1 \\ 0 & 0 & -3 & 4 \\ -1 & -2 & 0 & 2 \end{bmatrix}.$$

Scegliamo $i = 3$, cioè sviluppiamo il determinante rispetto alla terza riga. Dalla (3.1) si ha

$$\det A = -3 \det A_{33} - 4 \det A_{34},$$

dove

$$A_{33} = \begin{bmatrix} 3 & 1 & 0 \\ 2 & -1 & 1 \\ -1 & -2 & 2 \end{bmatrix}, \quad A_{34} = \begin{bmatrix} 3 & 1 & 1 \\ 2 & -1 & 3 \\ -1 & -2 & 0 \end{bmatrix}.$$

A_{33} e A_{34} hanno ordine 3: il loro determinante viene calcolato applicando ancora la regola di Laplace, alla prima riga per A_{33} e alla terza riga per A_{34}. Quindi $\det A_{33}$ e $\det A_{34}$ si possono esprimere mediante determinanti di ordine 2, che si sanno calcolare direttamente. Si ha perciò

$$\begin{aligned} \det A_{33} &= 3[(-1)(2) - (1)(-2)] - (1)[(2)(2) - (1)(-1)] = -5, \\ \det A_{34} &= (-1)[(1)(3) - (1)(-1)] - (-2)[(3)(3) - (1)(2)] = 10, \end{aligned}$$

e quindi $\det A = -25$.

È chiaro che se la matrice A ha ordine elevato e non contiene molti elementi nulli il calcolo di $\det A$ attraverso i cofattori risulta molto costoso. Nel prossimo capitolo studieremo il metodo di Gauss che consente di calcolare il determinante in modo molto meno costoso.

Siano A, $B \in \boldsymbol{R}^{n\times n}$, $\alpha \in \boldsymbol{R}$. Valgono le seguenti proprietà:

$\det A = \prod_{i=1}^{n} a_{ii}$ se A è diagonale o triangolare;

$\det I = 1$;

$\det A^T = \det A$

$\det(A\,B) = \det A \det B$ (regola di *Binet*);

$\det B = \alpha \det A$, se B è ottenuta da A moltiplicandone una riga (o colonna) per α;

$\det(\alpha A) = \alpha^n \det A$;

$\det B = -\det A$, se B è ottenuta da A scambiando fra loro due righe (o colonne);

$\det B = \det A$, se B è ottenuta da A aggiungendo ad una riga (o colonna) un'altra riga (o colonna) moltiplicata per un numero;

$\det A = 0$, se due o più righe (o colonne) di A sono linearmente dipendenti.

Poiché $\det A = \det A^T$, la regola di Laplace per il calcolo del determinante di A può essere applicata sommando nella (3.1) rispetto all'indice di riga invece che all'indice di colonna, cioè sviluppando il determinante rispetto a una colonna invece che rispetto a una riga.

Si definisce *matrice inversa* di A una matrice $B \in \mathbf{R}^{n\times n}$ tale che

$$A\,B = B\,A = I.$$

Una matrice A per cui non esiste una matrice inversa è detta *singolare*. Una matrice A non singolare ha un'unica inversa che viene indicata con A^{-1}.

Valgono le proprietà

$$(A^{-1})^{-1} = A, \quad (A\,B)^{-1} = B^{-1}A^{-1}, \quad (A^T)^{-1} = (A^{-1})^T.$$

Se A è simmetrica, anche A^{-1} lo è. Se A è ortogonale, allora $A^{-1} = A^T$.

Si definisce *aggiunta* di A la matrice $\mathrm{adj}A$ il cui elemento (i,j)-esimo è il cofattore c_{ji}. È facile verificare che

$$A\,\mathrm{adj}A = (\det A)\,I,$$

e quindi

$$A^{-1} = \frac{1}{\det A}\,\mathrm{adj}A.$$

Ne segue che la matrice A è non singolare se e solo se il suo determinante è diverso da zero. Dalla regola di Binet, poichè $\det I = 1$, si ha $\det A^{-1} = 1/\det A$.

Nel prossimo capitolo studieremo il metodo di Gauss con cui si calcola l'inversa senza passare attraverso la costruzione dell'aggiunta.

3.3 Sistemi lineari.

Data una matrice $A \in \boldsymbol{R}^{n\times n}$ e un vettore $\boldsymbol{b} \in \boldsymbol{R}^n$, si chiama *sistema lineare* un sistema di n equazioni della forma

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \quad \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases} \tag{3.2}$$

dove $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$ è detto *vettore delle incognite*. Con notazione più compatta il sistema (3.2) viene anche scritto

$$A\boldsymbol{x} = \boldsymbol{b}.$$

Risolvere un sistema lineare significa calcolare, se esiste, un vettore $\boldsymbol{x}$ che soddisfa le (3.2). La matrice A si dice *matrice dei coefficienti*, il vettore $\boldsymbol{b}$ *vettore dei termini noti*, la matrice $[A \,|\, \boldsymbol{b}]$, ottenuta affiancando alla matrice A la colonna $\boldsymbol{b}$, si dice *matrice aumentata*. Se $\boldsymbol{b} = \boldsymbol{0}$ il sistema si dice *omogeneo*. Il sistema si dice *consistente* se ha almeno una soluzione. Vale il seguente teorema fondamentale.

Teorema di Rouché-Capelli. *Se la matrice A e la matrice aumentata hanno lo stesso rango, il sistema (3.2) è consistente.*

Se A è non singolare (cioè esiste A^{-1}), il sistema ha un'unica soluzione, che può essere espressa come

$$\boldsymbol{x} = A^{-1}\boldsymbol{b}. \tag{3.3}$$

Se A è singolare ma il sistema è consistente, il sistema ha infinite soluzioni. Se il sistema è omogeneo, la matrice dei coefficienti e la matrice aumentata hanno ovviamente lo stesso rango e quindi il sistema è consistente. Se il sistema è omogeneo e A è non singolare, l'unica soluzione è quella nulla.

Vi sono moltissimi modi di risolvere un sistema lineare: alcuni sono adatti a sistemi di tipo generale, altri a sistemi con matrici di forma particolare, oppure a matrici di dimensioni particolarmente elevate. Nel prossimo capitolo se ne esamineranno alcuni fra quelli più usati.

3.4 Autovalori e autovettori.

Sia $\boldsymbol{R}^n$ lo spazio vettoriale dei vettori ad n componenti reali e $\boldsymbol{R}^{n\times n}$ lo spazio vettoriale delle matrici quadrate di ordine n ad elementi reali. Una matrice associata ad una applicazione lineare $f : \boldsymbol{R}^n \to \boldsymbol{R}^n$ dipende dalla base scelta in $\boldsymbol{R}^n$. Un problema interessante che si presenta è il seguente: esiste in $\boldsymbol{R}^n$ una base per cui la matrice associata ad f ha forma diagonale? La risposta, che non è sempre affermativa, dipende da un insieme particolare di scalari e di vettori legati alla matrice e detti rispettivamente autovalori e autovettori. Da tali grandezze dipendono anche fenomeni di propagazione dell'errore e proprietà di convergenza nei problemi lineari.

Sia $A \in \boldsymbol{R}^{n\times n}$. Un numero λ per cui esiste un vettore $\boldsymbol{x} \neq \boldsymbol{0}$ tale che valga la relazione

$$A\boldsymbol{x} = \lambda\boldsymbol{x} \tag{3.4}$$

è detto *autovalore* di A ed $\boldsymbol{x}$ è detto *autovettore* corrispondente a λ. L'insieme degli autovalori di A costituisce lo *spettro* di A e il modulo massimo $\rho(A)$ degli autovalori è detto *raggio spettrale* di A.

L'esistenza degli autovalori e degli autovettori di A è una naturale conseguenza del fatto che gli autovalori sono le soluzioni di un'equazione algebrica. Infatti il sistema (3.4), che si può scrivere anche nella forma

$$(A - \lambda I)\boldsymbol{x} = \mathbf{0}, \tag{3.5}$$

è omogeneo e ammette soluzioni non nulle se e solo se

$$\det(A - \lambda I) = 0. \tag{3.6}$$

Il determinante della matrice

$$A - \lambda I = \begin{bmatrix} a_{11} - \lambda & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} - \lambda & \dots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} - \lambda \end{bmatrix}$$

è un polinomio $p(\lambda)$ di grado n, i cui coefficienti dipendono dagli elementi di A. Il polinomio $p(\lambda)$ è detto *polinomio caratteristico* di A e l'equazione $p(\lambda) = 0$ è detta *equazione caratteristica* di A. Gli autovalori di A sono tutte e sole le soluzioni dell'equazione caratteristica.

Per il teorema fondamentale dell'algebra l'equazione caratteristica ha nel campo complesso n radici, tenendo conto della loro molteplicità. Quindi una matrice di ordine n ha, tenendo conto della loro molteplicità, n autovalori nel campo complesso. Poiché la matrice A ha elementi reali, l'equazione caratteristica ha coefficienti reali e se A ha un autovalore λ non reale, allora ha anche l'autovalore $\overline{\lambda}$.

Esempio. Il polinomio caratteristico della matrice

$$A = \begin{bmatrix} 1 & 3 \\ 3 & 1 \end{bmatrix}$$

è

$$p(\lambda) = \det(A - \lambda I) = \det \begin{bmatrix} 1 - \lambda & 3 \\ 3 & 1 - \lambda \end{bmatrix} = \lambda^2 - 2\lambda - 8.$$

L'equazione caratteristica corrispondente

$$\lambda^2 - 2\lambda - 8 = 0$$

ha come radici $\lambda_1 = -2$ e $\lambda_2 = 4$, che sono gli autovalori di A.

Fissiamo un autovalore, ad esempio $\lambda_1 = -2$ e calcoliamo gli autovettori corrispondenti risolvendo il sistema (3.5) che in questo caso diventa

$$\begin{bmatrix} 3 & 3 \\ 3 & 3 \end{bmatrix} \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \mathbf{0}.$$

Questo sistema è omogeneo, quindi è risolubile. Però la matrice dei coefficienti è singolare, quindi vi sono infiniti autovettori corrispondenti a λ_1 che dipendono da

una costante di proporzionalità: se $\boldsymbol{x}$ è autovettore di A, anche $k\boldsymbol{x}$, con $k \neq 0$, è autovettore di A, corrispondente allo stesso autovalore.

Nel nostro caso, dalla prima equazione si ottiene $x_1 + x_2 = 0$, da cui $x_1 = -x_2$, e ne segue che qualunque vettore della forma

$$\boldsymbol{x}_1 = k \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad \text{con} \quad k \neq 0,$$

è autovettore di A corrispondente all'autovalore $\lambda_1 = -2$. Gli autovettori corrispondenti a $\lambda_2 = 4$ si calcolano risolvendo il sistema

$$\begin{bmatrix} -3 & 3 \\ 3 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \mathbf{0}.$$

Dalla prima equazione si ottiene $-x_1 + x_2 = 0$, da cui $x_1 = x_2$, e ne segue che qualunque vettore della forma

$$\boldsymbol{x}_1 = k \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{con} \quad k \neq 0,$$

è autovettore di A corrispondente all'autovalore $\lambda_2 = 4$.

Se ripetiamo lo stesso procedimento con la matrice

$$A = \begin{bmatrix} 2 & -1 \\ 1 & 2 \end{bmatrix},$$

si ricava dall'equazione caratteristica

$$\lambda^2 - 4\lambda + 5 = 0$$

che gli autovalori sono $\lambda_1 = 2+i$ e $\lambda_2 = 2-i$. Risolvendo i sistemi (3.5) si ottengono gli autovettori $\boldsymbol{x}_1$ corrispondente a λ_1 e $\boldsymbol{x}_2$ corrispondente a λ_2 dove

$$\boldsymbol{x}_1 = k \begin{bmatrix} i \\ 1 \end{bmatrix} \quad \text{e} \quad \boldsymbol{x}_2 = k \begin{bmatrix} -i \\ 1 \end{bmatrix} \quad \text{con} \quad k \neq 0. \qquad \square$$

Per trovare gli autovalori di una matrice non è però conveniente passare attraverso la costruzione e la risoluzione dell'equazione caratteristica. Questa strada sarebbe troppo costosa, soprattutto se fossero richiesti solo alcuni autovalori, come in effetti accade nella pratica. Ad esempio, vedremo nel prossimo capitolo come si può determinare l'autovalore di modulo massimo con un metodo iterativo a basso costo.

Fra gli elementi della matrice A e i coefficienti del polinomio caratteristico di A vi sono alcune relazioni interessanti. In particolare, il coefficiente c_1 di λ^{n-1} è uguale alla traccia di A, cioè alla somma degli elementi diagonali di A, e il termine noto c_n è uguale al determinante di A. Dalle relazioni che legano i coefficienti e le radici di un'equazione algebrica risulta allora che:

$$\sum_{i=1}^{n} \lambda_i = \text{traccia}\, A \quad \text{e} \quad \prod_{i=1}^{n} \lambda_i = \det A.$$

Una conseguenza immediata è che una matrice A è singolare se e solo se ha almeno un autovalore nullo.

Siano λ un autovalore di A e $\boldsymbol{x}$ un autovettore corrispondente. Valgono le seguenti proprietà.

1. Gli autovalori di una matrice A diagonale o triangolare (superiore o inferiore) sono uguali agli elementi diagonali. Infatti la matrice $A - \lambda I$ è ancora diagonale o triangolare e quindi

$$\det(A - \lambda I) = \prod_{i=1}^{n}(a_{ii} - \lambda).$$

2. Un autovalore λ di A è anche autovalore di A^T. Infatti

$$\det(A - \lambda I) = \det(A - \lambda I)^T = \det(A^T - \lambda I),$$

e quindi se $\det(A - \lambda I) = 0$ è $\det(A^T - \lambda I) = 0$.

3. λ^i è autovalore di A^i e $\boldsymbol{x}$ è un autovettore corrispondente. Infatti

$$A^i\boldsymbol{x} = A^{i-1}A\boldsymbol{x} = A^{i-1}\lambda\boldsymbol{x} = \lambda A^{i-1}\boldsymbol{x} = \lambda A^{i-2}A\boldsymbol{x} = \ldots = \lambda^i\boldsymbol{x}.$$

4. Se $B = b_0 I + b_1 A + \ldots + b_k A^k$, dove $b_0, b_1, \ldots, b_k \in \boldsymbol{R}$, allora $\mu = b_0 + b_1\lambda + \ldots + b_k\lambda^k$ è autovalore di B e $\boldsymbol{x}$ è un autovettore corrispondente. Infatti

$$B\boldsymbol{x} = b_0\boldsymbol{x} + b_1 A\boldsymbol{x} + \ldots + b_k A^k\boldsymbol{x} = b_0\boldsymbol{x} + b_1\lambda\boldsymbol{x} + \ldots + b_k\lambda^k\boldsymbol{x} = \mu\boldsymbol{x}.$$

5. Se A è non singolare, allora $1/\lambda$ è autovalore di A^{-1} con $\boldsymbol{x}$ autovettore corrispondente. Infatti da

$$A\boldsymbol{x} = \lambda\boldsymbol{x}, \qquad \boldsymbol{x} \neq \boldsymbol{0},$$

si ha

$$\boldsymbol{x} = \lambda A^{-1}\boldsymbol{x}$$

e quindi

$$\lambda \neq 0 \quad \text{e} \quad A^{-1}\boldsymbol{x} = \frac{1}{\lambda}\,\boldsymbol{x}.$$

Esempio. La matrice

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \tag{3.7}$$

ha gli autovalori $\lambda_1 = 1 - \sqrt{2}$, $\lambda_2 = 1$, $\lambda_3 = 1 + \sqrt{2}$ e i corrispondenti autovettori sono

$$\boldsymbol{x}_1 = \begin{bmatrix} 1 \\ -\sqrt{2} \\ 1 \end{bmatrix}, \quad \boldsymbol{x}_2 = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}, \quad \boldsymbol{x}_3 = \begin{bmatrix} 1 \\ \sqrt{2} \\ 1 \end{bmatrix}.$$

La matrice A^{-1} ha gli autovalori $1/\lambda_1 = -\sqrt{2}-1,\ 1/\lambda_2 = 1,\ 1/\lambda_3 = \sqrt{2}-1$ e gli stessi autovettori di A. La matrice

$$B = 3A^2 - A + 2I = \begin{bmatrix} 7 & 5 & 3 \\ 5 & 10 & 5 \\ 3 & 5 & 7 \end{bmatrix}, \tag{3.8}$$

ha gli autovalori

$$\mu_i = 3\lambda_i^2 - \lambda_i + 2, \quad \text{per} \quad i = 1, 2, 3,$$

cioè $\mu_1 = 10 - 5\sqrt{2},\ \mu_2 = 4,\ \mu_3 = 10 + 5\sqrt{2}$ e gli stessi autovettori di A. □

3.5 Matrici simili.

Se in $\boldsymbol{R}^n$ consideriamo due basi diverse, ad una stessa applicazione lineare f risultano associate due matrici diverse A e B, che però sono legate da una relazione del tipo

$$A = P\,B\,P^{-1}, \tag{3.9}$$

dove P è una matrice non singolare che descrive il cambiamento di base. Le due matrici A e $B \in \mathbf{R}^{n\times n}$ vengono dette *simili* e la trasformazione (3.9) viene detta *trasformazione per similitudine*. Se la matrice P è reale ed è tale che $P^T P = I$ (cioè le colonne di P formano una base ortonormale di $\boldsymbol{R}^n$) la trasformazione viene detta *ortogonale*. La trasformazione per similitudine è una relazione di equivalenza, in quanto gode delle proprietà riflessiva, simmetrica e transitiva.

Le proprietà degli autovalori e autovettori sono proprietà intrinseche dell'applicazione lineare e non legate solamente alla matrice che la rappresenta in una particolare base. Vale infatti il seguente teorema.

Teorema. *Due matrici simili hanno gli stessi autovalori.*

Dim. Se A e B sono due matrici simili, dalla (3.9) segue che

$$\begin{aligned} &\det(A - \lambda I) = \det(P\,B\,P^{-1} - \lambda P\,P^{-1}) = \det\left[P\,(B - \lambda I)\,P^{-1}\right] \\ &= \det P \det(B - \lambda I) \det(P^{-1}) = \det(B - \lambda I) \end{aligned}$$

per cui le due matrici hanno lo stesso polinomio caratteristico e quindi hanno gli stessi autovalori. □

Da questo teorema risulta che se due matrici sono simili, hanno uguali la traccia e il determinante.

Una matrice A simile ad una matrice diagonale D si dice *diagonalizzabile*.

Teorema. *Sia A una matrice di ordine n. Allora le due condizioni:*

a) A ha n autovettori linearmente indipendenti,

b) A è diagonalizzabile,

sono equivalenti. Inoltre le colonne della matrice P, per cui $P^{-1}AP$ è diagonale, sono gli autovettori di A.

Dim. a) $\Rightarrow$ b) Siano $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ n autovettori linearmente indipendenti, corrispondenti agli autovalori $\lambda_1, \ldots, \lambda_n$. Siano D la matrice diagonale avente λ_i come i-esimo elemento diagonale, e P la matrice la cui i-esima colonna è uguale a $\boldsymbol{x}_i$. Dalla relazione

$$A\boldsymbol{x}_i = \lambda_i \boldsymbol{x}_i, \quad i = 1, 2, \ldots, n,$$

si ha anche che

$$A\,P = P\,D. \tag{3.10}$$

Essendo P non singolare, perché formata da colonne linearmente indipendenti, esiste P^{-1}; quindi dalla (3.10) si ha

$$A = P\,D\,P^{-1}.$$

b) $\Rightarrow$ a) Siano $A = P\,D\,P^{-1}$ e D matrice diagonale con gli autovalori di A come elementi diagonali. Allora risulta $AP = PD$. Indicando con $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n$ le colonne di P, si ha:

$$A\,[\boldsymbol{p}_1 \,|\, \boldsymbol{p}_2 \,|\, \ldots \,|\, \boldsymbol{p}_n] = [\lambda_1 \boldsymbol{p}_1 \,|\, \lambda_2 \boldsymbol{p}_2 \,|\, \ldots \,|\, \lambda_n \boldsymbol{p}_n]$$

e quindi

$$A\boldsymbol{p}_i = \lambda_i \boldsymbol{p}_i, \quad i = 1, 2, \ldots, n.$$

Perciò le colonne di P sono n autovettori di A, che risultano linearmente indipendenti, perché P è non singolare. □

Esempio. Le matrici A definita in (3.7) e $B = 3A^2 - A + 2I$ definita in (3.8) hanno entrambe tre autovalori distinti e gli stessi tre autovettori linearmente indipendenti. Quindi sono diagonalizzabili dalla stessa trasformazione per similitudine. Posto

$$P = \begin{bmatrix} 1 & 1 & 1 \\ -\sqrt{2} & 0 & \sqrt{2} \\ 1 & -1 & 1 \end{bmatrix},$$

si ha

$$P^{-1} = \begin{bmatrix} 1/4 & -\sqrt{2}/4 & 1/4 \\ 1/2 & 0 & -1/2 \\ 1/4 & \sqrt{2}/4 & 1/4 \end{bmatrix}$$

e

$$A = P \begin{bmatrix} 1-\sqrt{2} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1+\sqrt{2} \end{bmatrix} P^{-1},$$

$$B = P \begin{bmatrix} 10-5\sqrt{2} & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 10+5\sqrt{2} \end{bmatrix} P^{-1}.$$

□

Poiché i metodi che si usano per calcolare effettivamente gli autovalori di una matrice funzionano meglio (e in qualche caso anche solo) quando la matrice è diagonalizzabile, è importante studiare la classe delle matrici diagonalizzabili. Il seguente teorema ci dà un primo risultato interessante.

Teorema. *Autovettori corrispondenti ad autovalori distinti sono linearmente indipendenti.* □

Ne segue che se A ha n autovalori tutti distinti, allora ha n autovettori linearmente indipendenti e quindi è diagonalizzabile.

Se A non ha n autovalori distinti, può avere n autovettori linearmente indipendenti oppure no. Per esempio, la matrice identica I di $\boldsymbol{R}^{n\times n}$, che ha il solo autovalore 1, ha tutti i vettori $\boldsymbol{e}_i$ della base canonica di $\boldsymbol{R}^n$ come autovettori, infatti è

$$I\,\boldsymbol{e}_i = \boldsymbol{e}_i, \quad \text{per} \quad i = 1, \ldots, n.$$

Invece la matrice

$$A = \begin{bmatrix} 1 & 1 & & & \\ & 1 & 1 & & \\ & & \ddots & \ddots & \\ & & & 1 & 1 \\ & & & & 1 \end{bmatrix},$$

ha il solo autovalore 1 a cui corrispondono solo autovettori della forma $\boldsymbol{x} = k\boldsymbol{e}_1$, $k \neq 0$. In questo caso la matrice A non è diagonalizzabile.

Vi è un'importante classe di matrici che sono sicuramente diagonalizzabili anche se hanno autovalori non tutti distinti. Si tratta delle matrici simmetriche. Vale infatti il seguente teorema.

Teorema. *Sia A una matrice simmetrica. Allora*

(a) gli autovalori di A sono reali,

(b) A è diagonalizzabile per trasformazione ortogonale. □

Poichè le colonne della matrice P per cui $P^{-1}\,A\,P$ è diagonale sono gli autovettori di A, da questo teorema segue che una matrice simmetrica ha n autovettori ortonormali.

Esempio. La matrice

$$A = \begin{bmatrix} 11 & 8 & 2 \\ 8 & 5 & -10 \\ 2 & -10 & 2 \end{bmatrix}$$

è simmetrica. I suoi autovalori sono $\lambda_1 = -9$, $\lambda_2 = 9$ e $\lambda_3 = 18$. I corrispondenti autovettori normalizzati sono

$$\boldsymbol{x}_1 = \frac{1}{3}\begin{bmatrix} -1 \\ 2 \\ 2 \end{bmatrix}, \quad \boldsymbol{x}_2 = \frac{1}{3}\begin{bmatrix} 2 \\ -1 \\ 2 \end{bmatrix}, \quad \boldsymbol{x}_3 = \frac{1}{3}\begin{bmatrix} -2 \\ -2 \\ 1 \end{bmatrix}.$$

È facile verificare che gli autovettori sono ortogonali. Quindi è

$$A = P\,D\,P^T,$$

dove

$$P = \frac{1}{3}\begin{bmatrix} -1 & 2 & -2 \\ 2 & -1 & -2 \\ 2 & 2 & 1 \end{bmatrix}, \qquad D = \begin{bmatrix} -9 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 18 \end{bmatrix}.$$

□

3.6 Localizzazione degli autovalori.

Il calcolo effettivo degli autovalori di una matrice non è facile, soprattutto quando la matrice è di grandi dimensioni. Si deve ricorrere a metodi numerici, che in generale sono iterativi e quindi richiedono di possedere una stima, magari non molto precisa, dell'autovalore che si vuole approssimare. La stima più semplice sfrutta i cerchi di Gerschgorin.

Sia $A \in \boldsymbol{R}^{n\times n}$. I cerchi del piano complesso

$$K_i = \{\, z \in \mathbf{C} \;:\; |z - a_{ii}| \le \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}| \,\},\ i = 1, 2, \ldots, n,$$

di centro a_{ii} e raggio $r_i = \displaystyle\sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$ sono detti *cerchi di Gerschgorin.*

Teorema di Gerschgorin. *Gli autovalori della matrice A di ordine n sono tutti contenuti in*

$$\bigcup_{i=1,\ldots,n} K_i.$$

Dim. Sia λ un autovalore di A e $\boldsymbol{x}$ un autovettore corrispondente, ossia

$$A\boldsymbol{x} = \lambda\,\boldsymbol{x},\ \ \boldsymbol{x} \neq \boldsymbol{0}.$$

Allora si ha:

$$\sum_{j=1}^{n} a_{ij}\,x_j = \lambda\,x_i,\ \ i = 1, \ldots, n,$$

da cui

$$(\lambda - a_{ii})x_i = \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}\,x_j,\ \ i = 1, \ldots, n. \tag{3.11}$$

Sia x_p la componente di $\boldsymbol{x}$ di massimo modulo, cioè quella per cui

$$|x_p| = \max_{j=1,\ldots,n} |x_j| \neq 0, \tag{3.12}$$

e, ponendo $i = p$ nella (3.11), si ha:

$$(\lambda - a_{pp})x_p = \sum_{\substack{j=1 \\ j\neq p}}^{n} a_{pj}\, x_j,$$

da cui:

$$|\lambda - a_{pp}|\ |x_p| \leq \sum_{\substack{j=1 \\ j\neq p}}^{n} |a_{pj}|\, |x_j|,$$

e, per la (3.12),

$$|\lambda - a_{pp}|\ |x_p| \leq \sum_{\substack{j=1 \\ j\neq p}}^{n} |a_{pj}|\ |x_p|.$$

Infine, dividendo per $|x_p| > 0$, si ottiene

$$|\lambda - a_{pp}| \leq \sum_{\substack{j=1 \\ j\neq p}}^{n} |a_{pj}|,$$

e quindi $\lambda \in K_p$. Si osservi che, poiché a priori non è noto il valore dell'indice p, è possibile solo dire che λ appartiene all'unione di tutti i cerchi K_i. □

Poiché il teorema precedente può essere applicato anche alla matrice A^T, che ha gli stessi autovalori della matrice A, risulta che gli autovalori di A appartengono anche all'unione dei cerchi

$$H_i = \{\, z \in \mathbf{C} \ :\ |z - a_{ii}| \leq \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ji}|\,\},\ i = 1, 2, \ldots, n,$$

e quindi gli autovalori di A appartengono all'insieme

$$(\bigcup_{i=1,\ldots,n} K_i) \bigcap (\bigcup_{i=1,\ldots,n} H_i).$$

Si può poi dimostrare che se l'unione M_1 di k cerchi di Gerschgorin è disgiunta dall'unione M_2 dei rimanenti $n - k$, allora k autovalori appartengono a M_1 e $n - k$ autovalori appartengono a M_2.

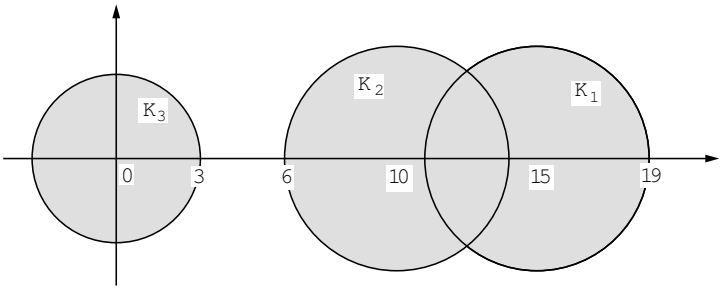
Esempio. I cerchi di Gerschgorin della matrice

$$A = \begin{bmatrix} 15 & -2 & 2 \\ 1 & 10 & -3 \\ -2 & 1 & 0 \end{bmatrix}$$

sono

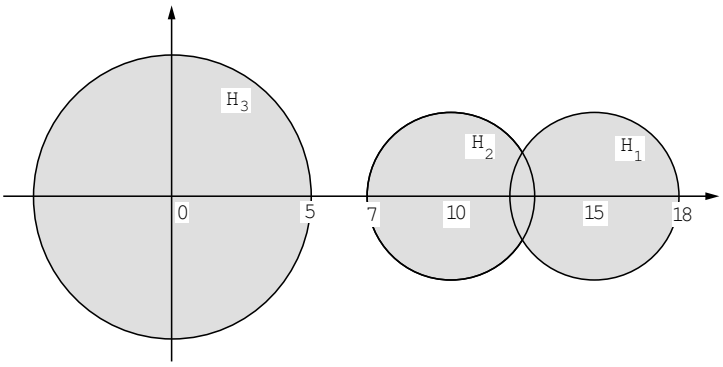
$$K_1 = \{z \in \mathbf{C} \ : \ |z - 15| \leq 4\},$$
$$K_2 = \{z \in \mathbf{C} \ : \ |z - 10| \leq 4\},$$
$$K_3 = \{z \in \mathbf{C} \ : \ |z| \leq 3\}.$$

Per il teorema gli autovalori stanno nelle aree grigie.

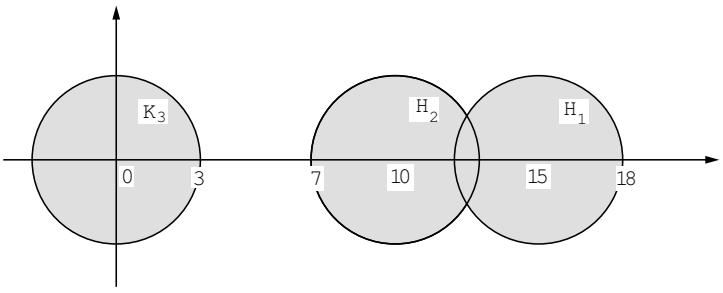


Consideriamo poi i cerchi associati alla matrice A^T.

$$H_1 = \{z \in \mathbf{C} \ : \ |z - 15| \leq 3\},$$
$$H_2 = \{z \in \mathbf{C} \ : \ |z - 10| \leq 3\},$$
$$H_3 = \{z \in \mathbf{C} \ : \ |z| \leq 5\},$$



Quindi gli autovalori di A stanno nell'intersezione dei due insiemi di cerchi.



Dei tre autovalori, uno è contenuto in K_3, mentre gli altri due appartengono ad $H_1 \cup H_2$. I due autovalori contenuti in $H_1 \cup H_2$ possono essere reali o complessi e hanno modulo compreso fra 7 e 18. L'autovalore contenuto in K_3 è reale; infatti, se avesse parte immaginaria non nulla, anche il suo coniugato dovrebbe essere un autovalore di A, essendo zero di un polinomio a coefficienti reali, e apparterrebbe ancora a K_3, mentre sappiamo che in K_3 c'è un solo autovalore. □

Una classe importante di matrici è quella delle matrici a predominanza diagonale, che si presentano spesso nella pratica. Una matrice $A \in \boldsymbol{R}^{n\times n}$ si dice *a predominanza diagonale in senso stretto* se per ogni $i = 1, \ldots, n$ risulta

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}} |a_{ij}|.$$

La definizione di predominanza diagonale in senso stretto si può dare anche per colonne, considerando le somme per colonne anziché per righe e in tal caso si specifica che la predominanza diagonale in senso stretto è per colonne.

Teorema. *Una matrice a predominanza diagonale in senso stretto è non singolare.*
Dim. Se A è a predominanza diagonale in senso stretto i cerchi di Gerschgorin, avendo raggio minore della distanza del centro dall'origine del piano complesso, non possono includere l'origine, e quindi A non può avere un autovalore nullo. □

Poiché gli autovalori della matrice A sono uguali a quelli della matrice A^T, la tesi del teorema vale anche nel caso in cui la predominanza diagonale in senso stretto della matrice sia per colonne.

3.7 Norme.

La norma è una generalizzazione della lunghezza di un vettore $\boldsymbol{x} \in \boldsymbol{R}^n$, data dall'espressione

$$\sqrt{x_1^2 + \cdots + x_n^2}.$$

Si chiama *norma vettoriale* una funzione che ad ogni $\boldsymbol{x} \in \boldsymbol{R}^n$ associa un numero reale, indicato con $\|\boldsymbol{x}\|$, e che verifica le seguenti proprietà

(a) $\|\boldsymbol{x}\| \geq 0$ e $\|\boldsymbol{x}\| = 0$ se e solo se $\boldsymbol{x} = \boldsymbol{0}$,

(b) $\|\alpha\boldsymbol{x}\| = |\alpha|\ \|\boldsymbol{x}\|$ per ogni $\alpha \in \boldsymbol{R}$,

(c) $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$ per ogni $\boldsymbol{y} \in \boldsymbol{R}^n$.

La proprietà (c) corrisponde alla ben nota *disuguaglianza triangolare*, per la quale in un triangolo la somma delle lunghezze di due lati è maggiore od uguale alla lunghezza del terzo lato. Sfruttando le tre proprietà si può dimostrare che la norma è una funzione uniformemente continua.

Norme vettoriali comunemente usate sono:

$$\begin{aligned}
&\text{norma 1} \qquad \|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i| \\
&\text{norma 2} \qquad \|\boldsymbol{x}\|_2 = \sqrt{\boldsymbol{x}^T\boldsymbol{x}} = \sqrt{\sum_{i=1}^{n} x_i^2} \\
&\text{norma } \infty \qquad \|\boldsymbol{x}\|_\infty = \max_{i=1,\dots,n} |x_i|.
\end{aligned}$$

Dimostriamo, come esempio, che la norma ∞ verifica le proprietà (a), (b) e (c) della definizione (in modo analogo si può procedere per le altre norme).

(a) Poiché $|x_i| \geq 0$ per $i = 1, \dots, n$, ne segue che $\max_{i=1,\dots,n} |x_i| \geq 0$ e quindi $\|\boldsymbol{x}\|_\infty \geq 0$; inoltre se $\max_{i=1,\dots,n} |x_i| = 0$, deve essere $|x_i| = 0$ per $i = 1, \dots, n$, e viceversa, quindi $\|\boldsymbol{x}\|_\infty = 0$ se e solo se $\boldsymbol{x} = \boldsymbol{0}$;

(b) $\|\alpha\boldsymbol{x}\|_\infty = \max_{i=1,\dots,n} |\alpha x_i| = \max_{i=1,\dots,n} |\alpha|\,|x_i| = |\alpha| \max_{i=1,\dots,n} |x_i| = |\alpha|\,\|\boldsymbol{x}\|_\infty$;

(c) $$\begin{aligned}\|\boldsymbol{x}+\boldsymbol{y}\|_\infty &= \max_{i=1,\dots,n} |x_i + y_i| \leq \max_{i=1,\dots,n} (|x_i| + |y_i|) \leq \max_{i=1,\dots,n} |x_i| + \max_{i=1,\dots,n} |y_i| \\ &= \|\boldsymbol{x}\|_\infty + \|\boldsymbol{y}\|_\infty.\end{aligned}$$

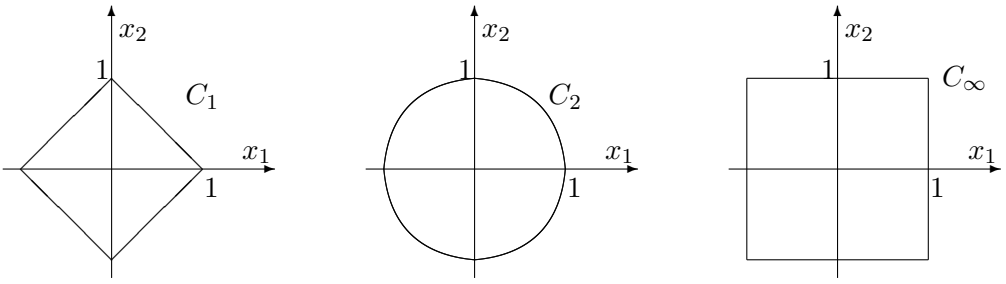
Le tre norme sono legate dalle relazioni

$$\begin{aligned}
|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_2 &\leq \sqrt{n}\,\|\boldsymbol{x}\|_\infty, \\
\|\boldsymbol{x}\|_2 \;\leq\; \|\boldsymbol{x}\|_1 &\leq \sqrt{n}\,\|\boldsymbol{x}\|_2.
\end{aligned}$$

La figura mostra gli insiemi

$$\begin{aligned}
C_1 &= \{\boldsymbol{x} \in \boldsymbol{R}^2 : \|\boldsymbol{x}\|_1 = 1\}, \\
C_2 &= \{\boldsymbol{x} \in \boldsymbol{R}^2 : \|\boldsymbol{x}\|_2 = 1\}, \\
C_\infty &= \{\boldsymbol{x} \in \boldsymbol{R}^2 : \|\boldsymbol{x}\|_\infty = 1\},
\end{aligned}$$

detti cerchi unitari di $\boldsymbol{R}^2$.



Si chiama *norma matriciale* una funzione che ad ogni $A \in \mathbf{R}^{n\times n}$ associa un numero reale, indicato con $\|A\|$, e che verifica le seguenti proprietà

(a) $\|A\| \geq 0$ e $\|A\| = 0$ se e solo se $A = O$,

(b) $\|\alpha A\| = |\alpha| \, \|A\|$ per ogni $\alpha \in \mathbf{R}$,

(c) $\|A + B\| \leq \|A\| + \|B\|$ per ogni $B \in \mathbf{R}^{n\times n}$,

(d) $\|AB\| \leq \|A\| \, \|B\|$ per ogni $B \in \mathbf{R}^{n\times n}$.

Quando in una stessa espressione matematica compaiono insieme norme di vettori e di matrici (e questo accade spesso in espressioni lineari) è importante che sia rispettata una relazione di compatibilità fra norme: una norma vettoriale $\| \, . \, \|_v$ e una norma matriciale $\| \, . \, \|_m$ si dicono *compatibili* se

$$\|A\boldsymbol{x}\|_v \leq \|A\|_m \, \|\boldsymbol{x}\|_v \quad \text{per ogni} \quad \boldsymbol{x} \in \mathbf{R}^n.$$

Norme matriciali comunemente usate sono

$$\text{norma 1} \qquad \|A\|_\infty = \max_{j=1,\ldots,n} \sum_{i=1}^{n} |a_{ij}|$$

$$\text{norma 2} \qquad \|A\|_2 = \sqrt{\rho(A^T A)}$$

$$\text{norma } \infty \qquad \|A\|_\infty = \max_{i=1,\ldots,n} \sum_{j=1}^{n} |a_{ij}|.$$

Queste norme verificano le seguenti proprietà:

1. Sono compatibili con le corrispondenti norme vettoriali.
2. Poiché $\|AB\| \leq \|A\| \, \|B\|$, per ogni intero $m \geq 1$ risulta
$$\|A^m\| \leq \|A\|^m.$$
3. $\|I\| = 1$.
4. Se $A \in \mathbf{R}^{n\times n}$ è non singolare, poiché $\|I\| = \|A^{-1}A\| \leq \|A^{-1}\| \, \|A\|$, risulta
$$\|A^{-1}\| \geq \frac{1}{\|A\|}.$$
5. $\rho(A) \leq \|A\|$. Infatti siano λ un autovalore e $\boldsymbol{x}$ il corrispondente autovettore normalizzato rispetto alla norma $\| \, . \, \|$, cioè con $\|\boldsymbol{x}\| = 1$. Quindi $|\lambda| = \|A\boldsymbol{x}\|$, da cui segue che
$$|\lambda| \leq \|A\| \, \|\boldsymbol{x}\| = \|A\|.$$
Questa relazione vale per ogni autovalore λ di A e quindi anche per quello di modulo massimo.
6. $\dfrac{1}{\sqrt{n}} \, \|A\|_\infty \leq \|A\|_2 \leq \sqrt{n} \, \|A\|_\infty, \qquad \dfrac{1}{\sqrt{n}} \, \|A\|_1 \leq \|A\|_2 \leq \sqrt{n} \, \|A\|_1,$
7. $\max\limits_{i,j} |a_{ij}| \leq \|A\|_2 \leq n \max\limits_{i,j} |a_{ij}|,$
8. Se A è una matrice simmetrica, risulta
$$\|A\|_1 = \|A\|_\infty, \qquad \|A\|_2 = \sqrt{\rho(A^T A)} = \sqrt{\rho(A^2)} = \sqrt{\rho^2(A)} = \rho(A).$$

Capitolo 4

Metodi per sistemi lineari e autovalori

Il primo problema che si affronta in questo capitolo è quello della risoluzione del sistema lineare

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{4.1}$$

dove $A \in \boldsymbol{R}^{n\times n}$ e $\boldsymbol{x}, \boldsymbol{b} \in \boldsymbol{R}^n$. Si suppone che la matrice A sia non singolare, per cui il sistema ammette una e una sola soluzione.

Il metodo più usato per risolvere i sistemi lineari è il metodo di Gauss, noto anche come metodo di *sostituzione*, che studieremo per primo. Il metodo di Gauss è un metodo *diretto*: se non ci fossero errori di rappresentazione dei dati e di arrotondamento nei calcoli, la soluzione del sistema verrebbe calcolata esattamente. Naturalmente, qualunque metodo si usi, si deve tener conto che la soluzione è affetta da un errore inerente, dovuto agli errori di rappresentazione dei dati, e da un errore algoritmico, dovuto all'arrotondamento nei calcoli.

Oltre ai metodi diretti, per risolvere un sistema lineare si possono usare anche i *metodi iterativi*, che vedremo successivamente. Per questi si deve tener conto anche dell'errore analitico.

Nello studio dei diversi metodi esamineremo anche il loro *costo computazionale*, cioè il numero di operazioni aritmetiche richieste. In pratica come misura di questo costo si considera solo il numero delle operazioni moltiplicative (moltiplicazioni e divisioni) richieste, in quanto il numero delle operazioni additive (addizioni e sottrazioni) è generalmente dello stesso ordine. Una valutazione più accurata dovrebbe prendere in considerazione anche le operazioni necessarie alla gestione dei dati del problema nella memoria del calcolatore (calcolo degli indici, permutazioni, ecc.).

Il costo computazionale è dato come funzione della dimensione n della matrice A. Di tale funzione si riportano solo i termini di ordine più elevato in n, usando il simbolo $\simeq$, che si legge appunto *uguale a meno di termini di ordine inferiore*. Si ha

quindi

$$\sum_{i=1}^{n} i \simeq \frac{n^2}{2} \qquad \text{e} \qquad \sum_{i=1}^{n} i^2 \simeq \frac{n^3}{3}.$$

4.1 Condizionamento.

Nel caso dei sistemi lineari, lo studio dell'errore inerente, e quindi del condizionamento del problema, viene fatto *perturbando* i dati del problema, cioè gli elementi della matrice A e del vettore $\boldsymbol{b}$, ed esaminando gli effetti indotti da queste perturbazioni sulla soluzione. Per semplicità si analizza il caso in cui il solo vettore $\boldsymbol{b}$ è perturbato. Si suppone che $\boldsymbol{b} \neq \boldsymbol{0}$ per cui $\boldsymbol{x} \neq \boldsymbol{0}$.

Indichiamo con $\delta\boldsymbol{b}$ il vettore delle perturbazioni dei termini noti del sistema (4.1) e con $\boldsymbol{x} + \delta\boldsymbol{x}$ la soluzione del sistema perturbato

$$A\,(\boldsymbol{x} + \delta\boldsymbol{x}) = \boldsymbol{b} + \delta\boldsymbol{b}.$$

Tenendo conto della (4.1) si ha

$$A\,\delta\boldsymbol{x} = \delta\boldsymbol{b},$$

da cui

$$\delta\boldsymbol{x} = A^{-1}\,\delta\boldsymbol{b},$$

e passando alle norme si ottiene

$$\|\delta\boldsymbol{x}\| = \|A^{-1}\delta\boldsymbol{b}\| \leq \|A^{-1}\|\,\|\delta\boldsymbol{b}\|. \tag{4.2}$$

D'altra parte per la (4.1) è

$$\|\boldsymbol{b}\| = \|A\,\boldsymbol{x}\| \leq \|A\|\,\|\boldsymbol{x}\|$$

da cui

$$\frac{1}{\|\boldsymbol{x}\|} \leq \frac{\|A\|}{\|\boldsymbol{b}\|}. \tag{4.3}$$

Moltiplicando (4.2) e (4.3) membro a membro, si ottiene

$$\frac{\|\delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \;\leq\; \mu(A)\,\frac{\|\delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|}, \tag{4.4}$$

dove $\mu(A) = \|A\|\,\|A^{-1}\|$ è il *numero di condizionamento* della matrice A. Il numero di condizionamento è sempre maggiore o uguale a 1, infatti:

$$\mu(A) = \|A\|\,\|A^{-1}\| \geq \|AA^{-1}\| = 1.$$

Poniamo $\epsilon_b = \|\delta\boldsymbol{b}\|/\|\boldsymbol{b}\|$ la perturbazione relativa del vettore $\boldsymbol{b}$ e $\epsilon_x = \|\delta\boldsymbol{x}\|/\|\boldsymbol{x}\|$ la perturbazione relativa indotta sul vettore $\boldsymbol{x}$. La (4.4) viene così riformulata

$$\epsilon_x \leq \mu(A)\,\epsilon_b.$$

Quindi se $\mu(A)$ assume valori piccoli, piccole perturbazioni sui dati inducono piccole perturbazioni sulla soluzione e il problema è ben condizionato: in questo caso la matrice del sistema si dice *ben condizionata*; se $\mu(A)$ assume valori grandi, allora piccole variazioni sui dati possono indurre grandi perturbazioni nella soluzione e il problema può essere mal condizionato: in questo caso la matrice del sistema si dice *mal condizionata*. Se ad esempio $\mu(A) = 1000$, l'errore ϵ_x può essere 1000 volte quello presente nei dati.

Se A è una matrice simmetrica non singolare, utilizzando la norma 2, si ha che

$$\mu_2(A) = \|A\|_2\|A^{-1}\|_2 = \frac{\lambda_{\max}}{\lambda_{\min}},$$

in cui $\lambda_{\max}$ e $\lambda_{\min}$ sono rispettivamente il massimo e il minimo modulo degli autovalori di A. Quindi A è tanto meglio condizionata quanto più vicini sono fra loro i suoi autovalori ed è mal condizionata se un autovalore è in modulo molto piccolo rispetto agli altri.

Esempio. Un esempio classico di matrice mal condizionata è la matrice di *Hilbert* definita da

$$a_{ij} = \frac{1}{i+j-1}, \quad i,j = 1,\ldots,n.$$

Per $n = 5$ si ha

$$A^{(5)} = \begin{bmatrix} 1 & 1/2 & 1/3 & 1/4 & 1/5 \\ 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \end{bmatrix}$$

$$[A^{(5)}]^{-1} = \begin{bmatrix} 25 & -300 & 1050 & -1400 & 630 \\ -300 & 4800 & -18900 & 26880 & -12600 \\ 1050 & -18900 & 79380 & -117600 & 56700 \\ -1400 & 26880 & -117600 & 179200 & -88200 \\ 630 & -12600 & 56700 & -88200 & 44100 \end{bmatrix}.$$

Per ogni valore di n la matrice $B^{(n)} = [A^{(n)}]^{-1}$ ha elementi $b_{ij}^{(n)}$ interi, tali che $|b_{ij}^{(n)}|$ è, per ogni i e j, una funzione crescente di n. Nella tabella che segue vengono riportati i valori del numero di condizionamento $\mu_2(A) = \|A\|_2\|A^{-1}\|_2$, in norma 2, e $\mu_\infty(A) = \|A\|_\infty\|A^{-1}\|_\infty$, in norma ∞, della matrice di Hilbert per valori di n da 2 a 10. Asintoticamente il numero di condizionamento in norma 2 di $A^{(n)}$ risulta essere una funzione crescente di n dell'ordine di $e^{3.5n}$. □

n	$\mu_2(A^{(n)})$	$\mu_\infty(A^{(n)})$
2	19.28	27
3	$5.241\ 10^2$	$7.480\ 10^2$
4	$1.551\ 10^4$	$2.837\ 10^4$
5	$4.766\ 10^5$	$9.436\ 10^5$
6	$1.495\ 10^7$	$2.907\ 10^7$
7	$4.754\ 10^8$	$9.852\ 10^8$
8	$1.526\ 10^{10}$	$3.387\ 10^{10}$
9	$4.932\ 10^{11}$	$1.099\ 10^{12}$
10	$1.603\ 10^{13}$	$3.535\ 10^{13}$

4.2 Sistemi lineari con matrice triangolare.

La risoluzione del sistema (4.1) è particolarmente semplice quando la matrice A è triangolare. Se, ad esempio, A è triangolare superiore, il sistema (4.1) è

$$\begin{cases} a_{ii}x_i + \displaystyle\sum_{j=i+1}^{n} a_{ij}x_j = b_i, \quad i = 1, \ldots, n-1, \\ a_{nn}x_n = b_n. \end{cases}$$

Poiché A è non singolare, è $a_{ii} \neq 0$, per $i = 1, \ldots, n$, e si ha:

$$\begin{cases} x_n = \dfrac{b_n}{a_{nn}}, \\ x_i = \dfrac{1}{a_{ii}} \Big[b_i - \displaystyle\sum_{j=i+1}^{n} a_{ij}x_j \Big], \quad i = n-1, \ldots, 1, \end{cases}$$

quindi la risoluzione procede calcolando nell'ordine $x_n, x_{n-1}, \ldots, x_1$: all'$i$-esimo passo, per calcolare x_i, vengono utilizzate le componenti di indice maggiore di i, già calcolate. Si tratta cioè di una *sostituzione all'indietro*.

Il costo computazionale è determinato tenendo conto del fatto che la componente x_i viene calcolata con $n - i$ moltiplicazioni e 1 divisione, per cui risulta

$$\sum_{i=1}^{n}(n - i + 1) = \sum_{i=1}^{n} i \simeq \frac{n^2}{2}.$$

Se la matrice del sistema fosse triangolare inferiore, la risoluzione avverrebbe in modo analogo, con il semplice scambio dell'ordine in cui svolgere i calcoli: dalla prima componente di $\boldsymbol{x}$ verso l'ultima (*sostituzione in avanti*).

Lo stesso procedimento può essere utilizzato per calcolare la matrice inversa di una matrice $A \in \boldsymbol{R}^{n\times n}$ non singolare e triangolare. Infatti la matrice X, inversa di A, è tale che

$$AX = I,$$

e quindi la k-esima colonna di X è un vettore $\boldsymbol{x}_k$ che verifica la relazione

$$A\boldsymbol{x}_k = \boldsymbol{e}_k, \quad k = 1, 2, \ldots, n, \tag{4.5}$$

dove $\boldsymbol{e}_k$ è la k-esima colonna di I. Poiché la matrice A è triangolare, anche la matrice X risulta triangolare: in particolare, se A è triangolare superiore, il vettore $\boldsymbol{x}_k$ ha le ultime $n-k$ componenti nulle. L'inversa si ottiene risolvendo gli n sistemi (4.5) in cui si determinano solo le prime k componenti di $\boldsymbol{x}_k$. Quindi la risoluzione del k-esimo sistema lineare (4.5) richiede $k^2/2$ operazioni moltiplicative, e il costo computazionale del calcolo della matrice inversa è dato da

$$\sum_{k=1}^{n} \frac{k^2}{2} \simeq \frac{n^3}{6}.$$

4.3 Metodo di Gauss.

Il modo più semplice per risolvere un sistema lineare consiste nell'eliminare passo per passo le incognite $x_1, x_2, \ldots, x_{n-1}$ dalle equazioni successive rispettivamente alla prima, seconda, ..., $(n-1)$-esima (per questo il metodo di Gauss è detto anche *metodo di eliminazione*). Questo procedimento, quando è applicabile, trasforma il sistema dato in un altro, equivalente ad esso e avente matrice triangolare superiore, che sappiamo già risolvere.

In pratica il metodo di Gauss costruisce una successione di matrici

$$[\, A^{(1)} \mid \boldsymbol{b}^{(1)} \,] = [\, A \mid \boldsymbol{b} \,], \quad [\, A^{(2)} \mid \boldsymbol{b}^{(2)} \,], \quad \ldots, \quad [\, A^{(n)} \mid \boldsymbol{b}^{(n)} \,]$$

tali che tutti i sistemi $A^{(k)}\boldsymbol{x} = \boldsymbol{b}^{(k)}$ siano equivalenti e che l'ultima matrice $A^{(n)}$ sia triangolare superiore. Il passaggio da una matrice alla successiva avviene mediante combinazioni lineari di righe, con cui si annullano successivamente gli elementi che si trovano al di sotto della diagonale principale. Esaminiamo in dettaglio come si passa da un sistema al successivo.

Al primo passo si suppone che $a_{11}^{(1)} \neq 0$ (l'elemento $a_{11}^{(1)}$ è detto *pivot* al primo passo). La prima riga della matrice $[\, A^{(2)} \mid \boldsymbol{b}^{(2)} \,]$ è uguale a quella di $[\, A^{(1)} \mid \boldsymbol{b}^{(1)} \,]$, cioè si pone

$$a_{1j}^{(2)} = a_{1j}^{(1)}, \quad \text{per } j = 1, \ldots, n, \quad \text{e} \quad b_1^{(2)} = b_1^{(1)}.$$

Sulle altre righe si operano delle opportune combinazioni lineari: esattamente alla i-esima riga, con $i = 2, \ldots, n$, viene sottratta la prima riga moltiplicata per il fattore $m_{i1} = a_{i1}^{(1)}/a_{11}^{(1)}$, cioè

$$a_{ij}^{(2)} = a_{ij}^{1)} - m_{i1}a_{1j}^{(1)}, \quad \text{per } j = 1, \ldots, n, \quad \text{e} \quad b_i^{(2)} = b_i^{(1)} - m_{i1}b_1^{(1)}.$$

Alla fine del primo passo tutti gli elementi della prima colonna con indice di riga maggiore di 1 risultano nulli.

Il procedimento viene ripetuto con le colonne successive. Quindi al generico k-esimo passo si opera nel modo seguente: la matrice $A^{(k)}$ è della forma:

$$A^{(k)} = \left[\begin{array}{cc} B^{(k)} & C^{(k)} \\ O & D^{(k)} \end{array}\right] \begin{array}{l} \} \quad k-1 \text{ righe} \\ \} \quad n-k+1 \text{ righe,} \end{array}$$

in cui $B^{(k)}$ è la sottomatrice triangolare superiore di ordine $k-1$, e si suppone che $a_{kk}^{(k)} \neq 0$ (l'elemento $a_{kk}^{(k)}$ è detto *pivot* al k-esimo passo). Le prime k righe della matrice $[\, A^{(k+1)} |\, \boldsymbol{b}^{(k+1)} \,]$ sono uguali a quelle di $[\, A^{(k)} |\, \boldsymbol{b}^{(k)} \,]$, cioè per $i = 1, \ldots, k$ si pone

$$a_{ij}^{(k+1)} = a_{ij}^{(k)}, \quad \text{per } j = 1, \ldots, n, \quad \text{e} \quad b_i^{(k+1)} = b_i^{(k)}.$$

Alla i-esima riga, con $i = k+1, \ldots, n$, viene sottratta la k-esima riga moltiplicata per il fattore $m_{ik} = a_{ik}^{(k)} / a_{kk}^{(k)}$, cioè

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, \quad \text{per } j = 1, \ldots, n, \text{ e } b_i^{(k+1)} = b_i^{(k)} - m_{ik} b_k^{(k)}. \tag{4.6}$$

Alla fine del k-esimo passo tutti gli elementi della k-esima colonna con indice di riga maggiore di k risultano nulli, cioè la matrice $A^{(k+1)}$ ha la forma

$$A^{(k+1)} = \left[\begin{array}{cc} B^{(k+1)} & C^{(k+1)} \\ O & D^{(k+1)} \end{array}\right] \begin{array}{l} \} \quad k \text{ righe} \\ \} \quad n-k \text{ righe,} \end{array}$$

in cui $B^{(k+1)}$ è ancora triangolare superiore.

La matrice finale $A^{(n)}$, che si ottiene all'$(n-1)$-esimo passo, è triangolare superiore.

Il procedimento descritto può essere portato a termine (senza effettuare scambi di righe) se e solo se $a_{kk}^{(k)} \neq 0$ per $1 \leq k \leq n-1$. Il seguente teorema, che dà una condizione sufficiente affiché ciò accada, fa riferimento alle sottomatrici principali di testa di A (una sottomatrice A_k di ordine k è principale di testa se i suoi elementi sono gli a_{ij} per $i, j = 1, \ldots, k$).

Teorema. *Se le sottomatrici principali di testa A_k di A sono non singolari per $1 \leq k \leq n-1$, allora il metodo di Gauss è applicabile senza effettuare scambi di righe.*

Dim. Si dimostra che $a_{kk}^{(k)} \neq 0$ per $1 \leq k \leq n-1$. Per $k = 1$ è $a_{11}^{(1)} = a_{11} \neq 0$ per ipotesi. Per $2 \leq k \leq n-1$ la sottomatrice principale di testa di ordine k della matrice $A^{(k)}$ è

$$B^{(k+1)} = \left[\begin{array}{cc} B^{(k)} & \boldsymbol{v} \\ \mathbf{0}^T & a_{kk}^{(k)} \end{array}\right],$$

dove $\boldsymbol{v}$, $\mathbf{0} \in \boldsymbol{R}^{k-1}$ e $\mathbf{0}$ è il vettore nullo. È $\det B^{(k+1)} = a_{kk}^{(k)} \det B^{(k)}$. Le matrici $B^{(k)}$ e $B^{(k+1)}$ sono ottenute rispettivamente da A_{k-1} e A_k effettuando combinazioni lineari di righe che lasciano invariato il determinante. Quindi

$$a_{kk}^{(k)} = \frac{\det A_k}{\det A_{k-1}} \neq 0. \qquad \square$$

Esempio. È dato il sistema $A\boldsymbol{x} = \boldsymbol{b}$, dove

$$A = \begin{bmatrix} -2 & 4 & -1 & -1 \\ 4 & -9 & 0 & 5 \\ -4 & 5 & -5 & 5 \\ -8 & 8 & -23 & 20 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 12 \\ -32 \\ 3 \\ -13 \end{bmatrix}.$$

La risoluzione con il metodo di Gauss procede nel modo seguente: si pone

$$[\, A^{(1)} |\, \boldsymbol{b}^{(1)} \,] = [\, A \mid \boldsymbol{b} \,] = \left[\begin{array}{cccc|c} -2 & 4 & -1 & -1 & 12 \\ 4 & -9 & 0 & 5 & -32 \\ -4 & 5 & -5 & 5 & 3 \\ -8 & 8 & -23 & 20 & -13 \end{array}\right].$$

Al primo passo il pivot è $a_{11}^{(1)} = -2 \neq 0$ e i fattori per le combinazioni lineari sono $m_{21} = a_{21}^{(1)}/a_{11}^{(1)} = -2$, $m_{31} = a_{31}^{(1)}/a_{11}^{(1)} = 2$, $m_{41} = a_{41}^{(1)}/a_{11}^{(1)} = 4$. Quindi alla seconda riga viene sottratta la prima riga moltiplicata per -2, alla terza riga viene sottratta la prima riga moltiplicata per 2, alla quarta riga viene sottratta la prima riga moltiplicata per 4 e si ottiene

$$[\, A^{(2)} |\, \boldsymbol{b}^{(2)} \,] = \left[\begin{array}{cccc|c} -2 & 4 & -1 & -1 & 12 \\ 0 & -1 & -2 & 3 & -8 \\ 0 & -3 & -3 & 7 & -21 \\ 0 & -8 & -19 & 24 & -61 \end{array}\right].$$

Al secondo passo il pivot è $a_{22}^{(2)} = -1 \neq 0$ e i fattori per le combinazioni lineari sono $m_{32} = a_{32}^{(2)}/a_{22}^{(2)} = 3$, $m_{42} = a_{42}^{(2)}/a_{22}^{(2)} = 8$. Quindi alla terza riga viene sottratta la seconda riga moltiplicata per 3 e alla quarta riga viene sottratta la seconda riga moltiplicata per 8 e si ottiene

$$[\, A^{(3)} |\, \boldsymbol{b}^{(3)} \,] = \left[\begin{array}{cccc|c} -2 & 4 & -1 & -1 & 12 \\ 0 & -1 & -2 & 3 & -8 \\ 0 & 0 & 3 & -2 & 3 \\ 0 & 0 & -3 & 0 & 3 \end{array}\right].$$

Al terzo passo il pivot è $a_{33}^{(3)} = 3 \neq 0$ e vi è una sola combinazione lineare da fare, con $m_{43} = a_{43}^{(3)}/a_{33}^{(3)} = -1$. Quindi alla quarta riga viene sommata la terza riga e si ottiene

$$[\, A^{(4)} \mid \boldsymbol{b}^{(4)} \,] = \left[\begin{array}{cccc|c} -2 & 4 & -1 & -1 & 12 \\ 0 & -1 & -2 & 3 & -8 \\ 0 & 0 & 3 & -2 & 3 \\ 0 & 0 & 0 & -2 & 6 \end{array}\right].$$

Risolvendo il sistema lineare $A^{(4)}\boldsymbol{x} = \boldsymbol{b}^{(4)}$ con il procedimento di sostituzione all'indietro si ottiene la soluzione $\boldsymbol{x} = [-2, 1, -1, -3]^T$. □

Se l'ipotesi del teorema non è verificata, si può ugualmente applicare il metodo di Gauss e calcolare la soluzione, effettuando uno scambio di righe del sistema. Infatti, se per un k accade che $a_{kk}^{(k)} = 0$, poiché la matrice A è non singolare, esiste almeno una riga di indice $r > k$, con l'elemento $a_{rk}^{(k)} \neq 0$; basta allora scambiare fra di loro la k-esima e la r-esima riga della matrice $[\, A^{(k)} \mid \boldsymbol{b}^{(k)} \,]$ e proseguire nel modo descritto.

4.4 Costo computazionale del metodo di Gauss.

Il numero delle operazioni moltiplicative richieste al k-esimo passo del metodo di Gauss è dato dal numero di operazioni moltiplicative richieste dalla (4.6). Tenendo conto che gli zeri della parte triangolare inferiore di $A^{(k+1)}$ non vengono effettivamente calcolati, per la i-esima riga, $i = k+1, \ldots, n$ sono richieste

1 divisione per calcolare m_{ik},

$n-k$ moltiplicazioni per calcolare $m_{ik}a_{kj}^{(k)}$ per $j = k+1, \ldots, n$,

1 moltiplicazione per calcolare $m_{ik}b_k^{(k)}$,

cioè $n-k+2$ operazioni moltiplicative. Quindi al k-esimo passo sono richieste $(n-k)(n-k+2)$ operazioni moltiplicative.

Per gli $n-1$ passi richiesti, il costo computazionale del metodo di Gauss è allora dato da

$$\sum_{k=1}^{n-1}(n-k)(n-k+2) \simeq \sum_{k=1}^{n-1}(n-k)^2 = \sum_{k=1}^{n-1}k^2 \simeq \frac{n^3}{3}.$$

La successiva fase di risoluzione del sistema triangolare $A^{(n)}\boldsymbol{x} = \boldsymbol{b}^{(n)}$ ha un costo computazionale dell'ordine n^2, inferiore a quello richiesto per trasformare il sistema in triangolare superiore. Quindi il costo computazionale per risolvere un sistema di ordine n con il metodo di Gauss è $n^3/3$.

4.5 Stabilità del metodo di Gauss.

Studi teorici e la sperimentazione hanno mostrato che la stabilità del metodo di Gauss dipende da quanto crescono gli elementi delle matrici $A^{(k)}$ rispetto a quelli della matrice A. In particolare, il metodo è tanto più stabile quanto minore è la crescita degli elementi.

Ad esempio, esaminiamo il sistema lineare $A\boldsymbol{x} = \boldsymbol{b}$, dove

$$A = \begin{bmatrix} \alpha & 1 \\ 1 & 0 \end{bmatrix} \quad \text{e} \quad \boldsymbol{b} = \begin{bmatrix} 1+\alpha \\ 1 \end{bmatrix}, \quad \alpha > 0,$$

la cui soluzione è $\boldsymbol{x} = [1, 1]^T$. Questo problema è ben condizionato per valori di α piccoli in modulo, infatti si ha

$$A^{-1} = \begin{bmatrix} 0 & 1 \\ 1 & -\alpha \end{bmatrix},$$

per cui il numero di condizionamento di A, $\mu_\infty(A) = (1+\alpha)^2$, è di poco superiore all'unità. È però

$$A^{(2)} = \begin{bmatrix} \alpha & 1 \\ 0 & -1/\alpha \end{bmatrix},$$

per cui $a_{22}^{(2)}$ può diventare comunque grande al diminuire di α. Sia ad esempio $\alpha = 0.3\ 10^{-3}$. Con il metodo di Gauss. operando in virgola mobile in base 10 con 3 cifre significative e arrotondamento, si ha

$$[A^{(2)}|\ \boldsymbol{b}^{(2)}] = \left[\begin{array}{cc|c} 0.3\ 10^{-3} & 1 & 0.100\ 10^{1} \\ 0 & -0.333\ 10^{4} & -0.333\ 10^{4} \end{array}\right]$$

da cui si ottiene

$$\widetilde{x}_2 = \frac{0.333\ 10^4}{0.333\ 10^4} = 1, \quad \widetilde{x}_1 = \frac{0.100\ 10^1 - 0.100\ 10^1}{0.3\ 10^{-3}} = 0.$$

L'errore elevato da cui è affetta la soluzione calcolata è causato dal fatto che l'elemento di massimo modulo di $A^{(2)}$ è più di 3000 volte l'elemento di massimo modulo di A. In generale per il metodo di Gauss la crescita degli elementi delle matrici $A^{(k)}$ rispetto alla matrice A non è limitabile a priori con una espressione che dipende solo da n, quindi il metodo di Gauss può essere instabile anche quando è applicato a problemi ben condizionati.

L'implementazione del metodo di Gauss deve perciò prevedere una strategia che limiti la crescita degli elementi di $A^{(k)}$ rispetto agli elementi di A. La più semplice è quella che utilizza la tecnica del *massimo pivot*.

Al k-esimo passo con la tecnica del massimo pivot si determina l'indice di riga r per cui

$$|a_{rk}^{(k)}| = \max_{k\le i\le n} |a_{ik}^{(k)}|,$$

e si scambiano la r-esima riga con la k-esima prima di calcolare $A^{(k+1)}$. In tal modo i fattori m_{ik} delle combinazioni lineari (4.6) hanno modulo minore o uguale a 1. Per gli elementi di $A^{(k+1)}$ si ha dalla (4.6)

$$|a_{ij}^{(k+1)}| = |a_{ij}^{(k)} - m_{ik}a_{kj}^{(k)}| \le |a_{ij}^{(k)}| + |m_{ik}|\ |a_{kj}^{(k)}| \le |a_{ij}^{(k)}| + |a_{kj}^{(k)}|;$$

indicando con $a_M^{(k)}$ il massimo modulo degli elementi di $A^{(k)}$, si ha

$$a_M^{(k+1)} \le 2a_M^{(k)},$$

e quindi

$$a_M^{(n)} \le 2a_M^{(n-1)} \le \ldots \le 2^{n-1}a_M^{(1)}. \tag{4.7}$$

Perciò con il metodo di Gauss con la variante del massimo pivot la crescita delle matrici $A^{(k)}$rispetto alla matrice A è limitata da un'espressione che dipende solo da n. Nei casi pratici la limitazione (4.7) non viene quasi mai raggiunta, per cui il

metodo di Gauss con la variante del massimo pivot è più stabile di quanto indichi la (4.7). Nel caso del sistema $A\boldsymbol{x} = \boldsymbol{b}$ dell'esempio del paragrafo precedente con $\alpha = 0.3\ 10^{-3}$, con il massimo pivot si scambiano le righe della matrice $[\ A \mid \boldsymbol{b}\ ]$. Operando con un'aritmetica in virgola mobile in base 10 con 3 cifre significative, si ottiene un risultato affetto da errore nullo.

4.6 Implementazione del metodo di Gauss.

Nell'implementazione su calcolatore le matrici $[\ A^{(k)} \mid \boldsymbol{b}^{(k)}\ ]$ condividono la stessa area di memoria: al k-esimo passo solo gli elementi delle ultime $n-k$ righe e colonne vengono modificati. Al termine la matrice $A^{(n)}$ risulta sovrapposta al triangolo superiore e alla diagonale principale della matrice A. Agli elementi del triangolo inferiore viene assegnato valore zero.

Gli scambi fra righe delle matrici $[\ A^{(k)} \mid \boldsymbol{b}^{(k)}\ ]$ che dovessero rendersi necessari non vengono eseguiti effettivamente sulla matrice, ma sono realizzati per mezzo di un vettore $\boldsymbol{v}$, i cui elementi inizialmente sono $v_i = i$, $i = 1, \ldots, n$. L'elemento a_{ij} della matrice A risulta allora memorizzato nella posizione di indice di riga v_i e colonna j. Quando è richiesto lo scambio delle righe di indice k e r della matrice A, questo scambio non viene eseguito sulla matrice ma sul vettore $\boldsymbol{v}$, scambiando fra loro gli elementi v_k e v_r.

L'implementazione del metodo di Gauss deve prevedere anche un controllo ad ogni passo sulla grandezza del pivot, per evitare che un pivot troppo piccolo causi una divisione per zero. Quando si utilizza la variante del massimo pivot, un pivot troppo piccolo indica una possibile singolarità della matrice A. Per questo, normalmente, un pivot, che risulta in modulo più piccolo di una quantità prefissata ϵ, viene considerato nullo e l'algoritmo viene interrotto. La scelta di un appropriato valore per ϵ deve tenere conto della precisione usata nel calcolo e può essre critica.

Il seguente programma Matlab implementa il metodo di Gauss con il massimo pivot. Un indicatore `ind` controlla se al k-esimo passo il pivot è troppo piccolo. Se in uscita `ind` ha il valore `true`, vuol dire che l'algoritmo si è concluso regolarmente, altrimenti il valore di `k` indica il passo in cui si è verificata l'interruzione. Il vettore `m` contiene i moltiplicatori $a_{ik}^{(k)}/a_{ik}^{(k)}$. La sostituzione all'indietro viene implementata con il prodotto scalare di sottovettori.

```
%    triangolarizzazione
v=1:n; ind=1; k=1; while ind&(k<=n-1)
    pivot=a(v(k),k);
    for i=k+1:n
        if abs(a(v(i),k))>pivot pivot=a(v(i),k);
            v([i,k])=v([k,i]);
        end
    end
```

```
    ind=abs(pivot)>tol;
    if ind m=a(v(k+1:n),k)/pivot;
            a(v(k+1:n),k+1:n)=a(v(k+1:n),k+1:n)-m*a(v(k),k+1:n);
            b(v(k+1:n))=b(v(k+1:n))-m*b(v(k));
    end
    k=k+1;
end
%    sostituzione all'indietro
x(n)=b(v(n))/a(v(n),n); for i=n-1:-1:1
    x(i)=(b(v(i))-a(v(i),i+1:n)*x(i+1:n)')/a(v(i),i);
end
```

4.7 Fattorizzazione LU.

Il metodo di Gauss consente di risolvere anche un altro problema: quello della *fattorizzazione* di una matrice A nel prodotto di una matrice L triangolare inferiore con elementi principali uguali ad 1 e di una matrice U triangolare superiore, cioè

$$A = L\,U. \tag{4.8}$$

Teorema. *Se le sottomatrici principali di testa A_k di A sono non singolari per $1 \le k \le n-1$, allora esiste la fattorizzazione LU di A ed è*

$$L = \begin{bmatrix} 1 & & & & & \\ m_{21} & \ddots & & & & \\ \vdots & \ddots & 1 & & & \\ \vdots & & m_{k+1,k} & \ddots & & \\ \vdots & & \vdots & \ddots & \ddots & \\ m_{n1} & \cdots & m_{nk} & \cdots & m_{n,n-1} & 1 \end{bmatrix}, \qquad U = A^{(n)}.$$

□

Durante l'implementazione del metodo di Gauss anche la matrice L può essere memorizzata nella stessa area di memoria della A: i moltiplicatori m_{jk}, $j = k+1, \ldots, n$, del k-esimo passo sono memorizzati nella stessa area di memoria occupata dagli elementi $a_{jk}^{(k)}$, che non vengono più utilizzati nei passi successivi. Al termine del procedimento la matrice L, esclusa la diagonale principale, i cui elementi sono uguali a 1, è memorizzata nella stessa area di memoria inizialmente occupata dalla parte strettamente triangolare inferiore di A e la matrice U è memorizzata nella stessa area di memoria inizialmente occupata dalla parte triangolare superiore di A.

Il costo computazionale della fattorizzazione LU è lo stesso del metodo di Gauss.

La fattorizzazione LU è particolarmente semplice nel caso in cui A è una matrice tridiagonale di ordine n

$$A = \begin{bmatrix} a_1 & c_1 & & & \\ b_1 & a_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & c_{n-1} \\ & & & b_{n-1} & a_n \end{bmatrix}.$$

Posto

$$\alpha_1 = a_1,$$
$$\beta_i = b_i/\alpha_i, \quad \alpha_{i+1} = a_{i+1} - \beta_i c_i \quad \text{per } i = 1, \ldots, n-1,$$

se $\alpha_i \neq 0$ per $i = 1, \ldots, n-1$, si ha:

$$LU = \begin{bmatrix} 1 & & & & \\ \beta_1 & 1 & & & \\ & \beta_2 & \ddots & & \\ & & \ddots & \ddots & \\ & & & \beta_{n-1} & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 & c_1 & & & \\ & \alpha_2 & c_2 & & \\ & & \ddots & \ddots & \\ & & & \ddots & c_{n-1} \\ & & & & \alpha_n \end{bmatrix}.$$

La fattorizzazione LU di una matrice tridiagonale richiede $2n-2$ operazioni moltiplicative.

4.8 Applicazioni del metodo di Gauss

• Calcolo del determinante. Il metodo di Gauss può essere utilizzato anche per calcolare il determinante di A. Infatti la matrice $A^{(n)}$ è ottenuta dalla A con combinazioni lineari di righe, che non alterano il determinante, e quindi ha lo stesso determinante di A. Risulta perciò

$$\det A = \det A^{(n)} = \prod_{k=1}^{n} a_{kk}^{(k)}.$$

Il costo computazionale del calcolo del determinante è $n^3/3$.

• Risoluzione contemporanea di più sistemi. Il metodo di Gauss può essere utilizzato per la risoluzione contemporanea di più sistemi lineari con la stessa matrice dei coefficienti A e diverse colonne di termini noti. Sia infatti $B \in \boldsymbol{R}^{n\times r}$ la matrice formata da r colonne di termini noti. Allora la matrice $X \in \boldsymbol{R}^{n\times r}$, soluzione del sistema

$$AX = B,$$

si ottiene costruendo la successione

$$[\,A^{(1)} \mid B^{(1)}] = [\,A \mid B\,],\ [\,A^{(2)} \mid B^{(2)}\,],\ \ldots\ ,\ [\,A^{(n)} \mid B^{(n)}\,],$$

in modo che la matrice $[\,A^{(k+1)} \mid B^{(k+1)}\,]$ sia ottenuta dalla $[\,A^{(k)} \mid B^{(k)}\,]$ con le stesse combinazioni lineari che si fanno nel caso di un solo termine noto, cioè

$$b_{ij}^{(k+1)} = b_{ij}^{(k)} - m_{ik} b_{kj}^{(k)}, \quad \text{per} \quad j = 1, \ldots, r \quad \text{e} \quad i = k+1, \ldots, n.$$

Al termine si risolvono i sistemi con matrice triangolare superiore

$$A^{(n)} X = B^{(n)},$$

usando formule analoghe a quelle del caso di una sola colonna di termini noti. Complessivamente il costo computazionale è dato da:

a) al k-esimo passo, per il calcolo di $[\,A^{(k+1)} \mid B^{(k+1)}\,]$ occorrono $(n-k)(n-k+r+1)$ operazioni moltiplicative (ad $A^{(k)}$ sono state infatti affiancate r colonne, mentre il numero di righe è rimasto invariato); per gli $n-1$ passi richiesti il costo computazionale, a meno di termini di ordine inferiore, è dato da

$$\sum_{k=1}^{n-1} (n-k)(n-k+r+1) \simeq \sum_{k=1}^{n-1} k^2 + r \sum_{k=1}^{n-1} k \simeq \frac{n^3}{3} + r\ \frac{n^2}{2};$$

b) per la risoluzione degli r sistemi lineari la cui matrice è triangolare superiore, il costo computazionale è dato da $rn^2/2$.

Quindi il costo computazionale è

$$\frac{n^3}{3} + rn^2. \tag{4.9}$$

• Calcolo della matrice inversa. Un caso particolarmente importante è quello relativo al calcolo dell'inversa di una matrice A non singolare, in cui la matrice B è la matrice identica di ordine n

$$AX = I.$$

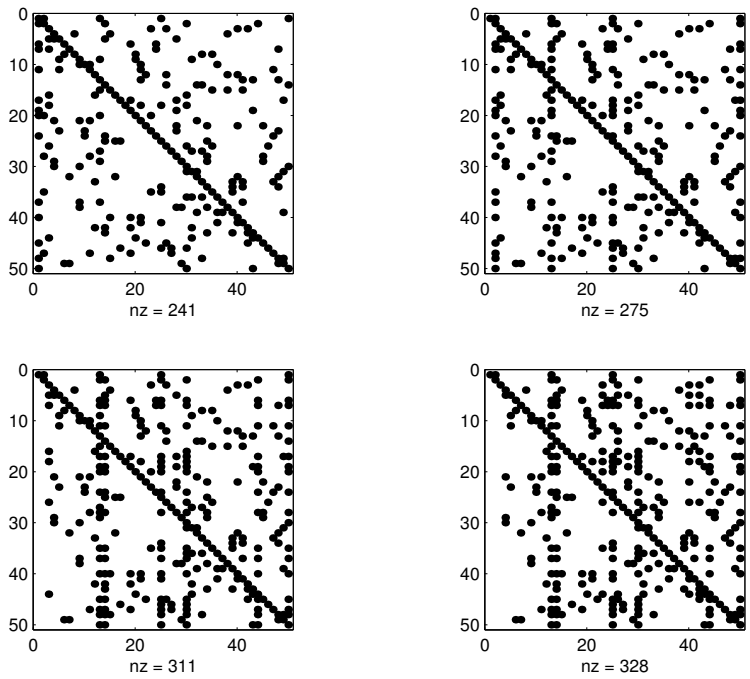
Il costo computazionale del calcolo dell'inversa di una matrice di ordine n con il metodo di Gauss è però inferiore a $4n^3/3$ come risulterebbe dalla (4.9) per $n = r$. Infatti in questo caso si ha $B^{(n)} = L^{-1}$, cioè $B^{(n)}$ è triangolare inferiore: ne segue che al k-esimo passo per la costruzione di $[\,A^{(k+1)} \mid B^{(k+1)}\,]$ occorrono $(n-k)(n+1)$ operazioni moltiplicative. Quindi per gli $n-1$ passi richiesti il costo computazionale è

$$\sum_{k=1}^{n-1} (n-k)(n+1) \simeq \frac{n^3}{2}.$$

Infine la risoluzione dei sistemi $UX = L^{-1}$ ha lo stesso costo computazionale $n^3/2$. In totale il costo computazionale dell'inversione di una matrice di ordine n con il metodo di Gauss è n^3.

4.9 Metodi iterativi per i sistemi lineari.

Per risolvere un sistema lineare $A\boldsymbol{x} = \boldsymbol{b}$, oltre al metodo di Gauss, che appartiene alla classe dei *metodi diretti*, sono disponibili anche i *metodi iterativi*, che sono particolarmente utili quando la matrice è di grandi dimensioni e sparsa. Infatti quando si utilizza il metodo di Gauss può accadere che nelle matrici intermedie vengano generati molti elementi diversi da zero in corrispondenza ad elementi nulli della matrice iniziale (questo fenomeno si chiama *fill-in*). La seguente figura si riferisce ad una matrice di dimensione $n = 50$ che su ogni riga, oltre al termine principale, ha non più di 4 elementi in posizione casuale.



I grafici mostrano lo schema degli elementi non nulli della matrice A iniziale e delle matrici $A^{(2)}$, $A^{(3)}$ e $A^{(4)}$ generate nei primi passi dal metodo di Gauss. Sotto ad ogni matrice è indicato anche il numero degli elementi non nulli. Come si vede, ad ogni passo questo numero cresce, malgrado il fatto che le prime colonne vengano progressivamente svuotate.

Sia A non singolare e si consideri la decomposizione

$$A = M - N, \tag{4.10}$$

dove M è una matrice non singolare. Dalla (4.10), sostituendo nel sistema lineare dato, risulta

$$M\boldsymbol{x} - N\boldsymbol{x} = \boldsymbol{b},$$

cioè

$$\boldsymbol{x} = M^{-1}N\boldsymbol{x} + M^{-1}\boldsymbol{b}.$$

Posto

$$P = M^{-1}N \quad \text{e} \quad \boldsymbol{q} = M^{-1}\boldsymbol{b}, \tag{4.11}$$

si ottiene il seguente sistema equivalente al sistema dato

$$\boldsymbol{x} = P\boldsymbol{x} + \boldsymbol{q}. \tag{4.12}$$

Dato un vettore iniziale $\boldsymbol{x}^{(0)}$, si considera la successione $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots$, così definita

$$\boldsymbol{x}^{(k)} = P\boldsymbol{x}^{(k-1)} + \boldsymbol{q}, \quad k = 1, 2, \ldots. \tag{4.13}$$

La successione $\boldsymbol{x}^{(k)}$ si dice *convergente* al vettore $\boldsymbol{x}^*$ e si indica con

$$\boldsymbol{x}^* = \lim_{k\to\infty} \boldsymbol{x}^{(k)},$$

se al tendere di k all'infinito le componenti di $\boldsymbol{x}^{(k)}$ convergono alle corrispondenti componenti di $\boldsymbol{x}^*$. Allora passando al limite nella (4.13) risulta

$$\boldsymbol{x}^* = P\boldsymbol{x}^* + \boldsymbol{q}, \tag{4.14}$$

cioè $\boldsymbol{x}^*$ è la soluzione del sistema (4.12) e quindi del sistema dato.

La relazione (4.13) individua un *metodo iterativo* in cui, a partire da un vettore iniziale $\boldsymbol{x}^{(0)}$, la soluzione viene approssimata utilizzando una successione $\{\boldsymbol{x}^{(k)}\}$ di vettori. La matrice P si dice *matrice di iterazione del metodo*.

Al variare del vettore iniziale $\boldsymbol{x}^{(0)}$ si ottengono dalla (4.13) diverse successioni $\{\boldsymbol{x}^{(k)}\}$, alcune delle quali possono essere convergenti ed altre no. Un metodo iterativo è detto *convergente* se, qualunque sia il vettore iniziale $\boldsymbol{x}^{(0)}$, la successione $\{\boldsymbol{x}^{(k)}\}$ è convergente.

Esempio. Si consideri il sistema (4.12) in cui

$$P = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 2 \end{bmatrix}, \quad \boldsymbol{q} = \boldsymbol{0}, \quad \text{e quindi} \quad \boldsymbol{x}^* = \boldsymbol{0}.$$

Allora

$$P^k = \begin{bmatrix} 0.5^k & 0 & 0 \\ 0 & 0.5^k & 0 \\ 0 & 0 & 2^k \end{bmatrix}.$$

Se $\boldsymbol{x}^{(0)} = [1, 0, 0]^T$, si ottiene la successione

$$\boldsymbol{x}^{(k)} = [0.5^k, 0, 0]^T, \ k = 1, 2, \ldots,$$

che converge alla soluzione del sistema. Se invece $\boldsymbol{x}^{(0)} = [0, 1, 1]^T$, si ottiene la successione

$$\boldsymbol{x}^{(k)} = [0, 0.5^k, 2^k]^T, \ k = 1, 2, \ldots,$$

che non converge. Questo è un esempio di metodo non convergente. □

Teorema. *Il metodo iterativo (4.13) è convergente se e solo se* $\rho(P) < 1$. □

La condizione espressa da questo teorema è necessaria e sufficiente per la convergenza del metodo (4.13), ma in generale non è di agevole verifica. Conviene allora utilizzare, quando è possibile, una condizione sufficiente di convergenza di più facile verifica, come quella del seguente teorema.

Teorema. *Se esiste una norma matriciale* $\| \cdot \|$ *per cui* $\|P\| < 1$, *il metodo iterativo (4.13) è convergente.*

Dim. Sottraendo membro a membro la (4.13) dalla (4.14) risulta

$$\boldsymbol{x}^* - \boldsymbol{x}^{(k)} = P(\boldsymbol{x}^* - \boldsymbol{x}^{(k-1)}), \quad k = 1, 2, \dots \ .$$

Indicato con

$$\boldsymbol{e}^{(k)} = \boldsymbol{x}^* - \boldsymbol{x}^{(k)},$$

il vettore *errore* alla k-esima iterazione, si ha

$$\boldsymbol{e}^{(k)} = P\boldsymbol{e}^{(k-1)}, \quad k = 1, 2, \dots \tag{4.15}$$

e quindi

$$\boldsymbol{e}^{(k)} = P\boldsymbol{e}^{(k-1)} = P^2\boldsymbol{e}^{(k-2)} = \dots = P^k\boldsymbol{e}^{(0)}.$$

Passando alle norme ne segue che

$$\|\boldsymbol{e}^{(k)}\| = \|P^k\boldsymbol{e}^{(0)}\| \leq \|P\|^k\|\boldsymbol{e}^{(0)}\|.$$

Essendo $\|P\| < 1$, risulta $\lim_{k\to\infty} \|P\|^k = 0$, da cui

$$\lim_{k\to\infty} \|\boldsymbol{e}^{(k)}\| = 0$$

per ogni $\boldsymbol{e}^{(0)}$. Quindi per la continuità della norma è

$$\lim_{k\to\infty} \boldsymbol{e}^{(k)} = \boldsymbol{0}, \tag{4.16}$$

cioè

$$\lim_{k\to\infty} \boldsymbol{x}^{(k)} = \boldsymbol{x}^*. \qquad \square$$

Se il metodo è convergente, la (4.16) vale per ogni $\boldsymbol{x}^{(0)}$, e in particolare deve valere se $\boldsymbol{x}^{(0)}$ è tale che il vettore $\boldsymbol{e}^{(0)} = \boldsymbol{x}^* - \boldsymbol{x}^{(0)}$ è un autovettore di P corrispondente ad un autovalore λ di modulo massimo, cioè $|\lambda| = \rho(P)$. In questo caso risulta

$$P\boldsymbol{e}^{(0)} = \lambda\boldsymbol{e}^{(0)}$$

e quindi

$$\boldsymbol{e}^{(k)} = P^k \boldsymbol{e}^{(0)} = \lambda^k \boldsymbol{e}^{(0)}.$$

Ne segue che

$$\|\boldsymbol{e}^{(k)}\| = [\rho(P)]^k \, \|\boldsymbol{e}^{(0)}\|.$$

Perciò il metodo (4.13) è tanto più veloce quanto minore è $\rho(P)$.

In un metodo iterativo ad ogni iterazione il costo computazionale è principalmente determinato dalla operazione di moltiplicazione della matrice P per un vettore, che richiede n^2 operazioni moltiplicative se la matrice A non ha specifiche proprietà. Se invece A è sparsa, per esempio ha un numero di elementi non nulli dell'ordine di n, la moltiplicazione di P per un vettore richiede molte meno operazioni moltiplicative. In questo caso i metodi iterativi possono risultare vantaggiosi rispetto a quelli diretti.

4.10 Criterio di arresto.

Poiché con un metodo iterativo non è ovviamente possibile calcolare in generale la soluzione con un numero finito di iterazioni, occorre individuare dei criteri per l'arresto del procedimento. Il criterio più comunemente usato, fissata una tolleranza ϵ, che tiene conto anche della precisione utilizzata nei calcoli, è ii seguente:

$$\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\| \leq \epsilon. \tag{4.17}$$

Questa condizione però non garantisce che la soluzione sia stata approssimata con la precisione ϵ. Infatti per la (4.15) è:

$$\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)} = [\boldsymbol{x}^* - \boldsymbol{x}^{(k-1)}] - [\boldsymbol{x}^* - \boldsymbol{x}^{(k)}] = \boldsymbol{e}^{(k-1)} - \boldsymbol{e}^{(k)} = (I - P)\boldsymbol{e}^{(k-1)}.$$

Se esiste una norma $\|\,.\,\|$ per cui $\|P\| < 1$, per la 5. del par. 3.7 è $\rho(P) < 1$. Quindi la matrice $I - P$ è non singolare e si ha

$$\|\boldsymbol{e}^{(k-1)}\| \leq \|(I - P)^{-1}\| \, \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\|,$$

per cui può accadere che $\|\boldsymbol{e}^{(k-1)}\|$ sia elevata anche se la condizione (4.17) è verificata.

In un programma che implementa un metodo iterativo deve essere sempre previsto un controllo sulla effettiva convergenza del metodo, ad esempio verificando che $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\|_\infty$ decresca. Può infatti accadere che un metodo iterativo la cui matrice di iterazione P è tale che $\rho(P) < 1$, per gli effetti indotti dagli errori di arrotondamento, non converga in pratica, e questo accade, in particolare, quando la matrice A è fortemente mal condizionata e $\rho(P)$ è molto vicino ad 1. Inoltre anche se $\rho(P)$ è piccolo è possibile che la (4.17) non sia verificata se ϵ è incompatibile con la precisione con cui si opera. È perciò necessario interrompere l'esecuzione quando il numero delle iterazioni diventa troppo elevato.

4.11 Metodi iterativi di Jacobi e Gauss-Seidel.

Fra i metodi iterativi individuati da una particolare scelta della decomposizione (4.10) sono particolarmente importanti il metodo di Jacobi e il metodo di Gauss-Seidel, per i quali è possibile dare delle condizioni sufficienti di convergenza verificate da molte delle matrici che si ottengono risolvendo problemi differenziali.

Si consideri la decomposizione della matrice A

$$A = D - B - C$$

dove

$$d_{ij} = \begin{cases} a_{ij} & \text{se } i = j \\ 0 & \text{se } i \neq j, \end{cases} \qquad b_{ij} = \begin{cases} -a_{ij} & \text{se } i > j \\ 0 & \text{se } i \leq j, \end{cases} \qquad c_{ij} = \begin{cases} 0 & \text{se } i \geq j \\ -a_{ij} & \text{se } i < j, \end{cases}$$

Scegliendo $M = D, \quad N = B + C$, si ottiene il *metodo di Jacobi*.
Scegliendo $M = D - B, \quad N = C$, si ottiene il *metodo di Gauss-Seidel*.

Per queste decomposizioni risulta $\det M \neq 0$ se e solo se tutti gli elementi principali di A sono non nulli.

Indicando con J la matrice di iterazione del metodo di Jacobi, dalla (4.11) si ha

$$J = D^{-1}(B + C),$$

per cui la (4.13) diviene

$$\boldsymbol{x}^{(k)} = J\boldsymbol{x}^{(k-1)} + D^{-1}\boldsymbol{b}.$$

Nella pratica il metodo di Jacobi viene implementato nel modo seguente:

$$x_i^{(k)} = \frac{1}{a_{ii}}\left[b_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}x_j^{(k-1)}\right], \quad i = 1, 2, \ldots, n. \tag{4.18}$$

In questo metodo quindi le componenti del vettore $\boldsymbol{x}^{(k)}$ sostituiscono simultaneamente al termine dell'iterazione le componenti di $\boldsymbol{x}^{(k-1)}$.

Indicando con G la matrice di iterazione del metodo di Gauss-Seidel, dalla (4.11) si ha

$$G = (D - B)^{-1}C,$$

per cui la (4.13) diviene

$$\boldsymbol{x}^{(k)} = G\boldsymbol{x}^{(k-1)} + (D - B)^{-1}\boldsymbol{b}. \tag{4.19}$$

Per descrivere come il metodo di Gauss-Seidel viene implementato, conviene prima trasformare la (4.19) così

$$(D - B)\boldsymbol{x}^{(k)} = C\boldsymbol{x}^{(k-1)} + \boldsymbol{b},$$

$$D\boldsymbol{x}^{(k)} = B\boldsymbol{x}^{(k)} + C\boldsymbol{x}^{(k-1)} + \boldsymbol{b},$$

$$\boldsymbol{x}^{(k)} = D^{-1}B\boldsymbol{x}^{(k)} + D^{-1}C\boldsymbol{x}^{(k-1)} + D^{-1}\boldsymbol{b}.$$

Il metodo di Gauss-Seidel viene allora implementato nel modo seguente:

$$x_i^{(k)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right], \quad i = 1, 2, \ldots, n. \tag{4.20}$$

La differenza fondamentale con il metodo di Jacobi è che qui per calcolare le componenti del vettore $\boldsymbol{x}^{(k)}$ si utilizzano anche le componenti già calcolate dello stesso vettore. Quindi nella implementazione del metodo di Jacobi è necessario disporre, contemporaneamente, di entrambi i vettori $\boldsymbol{x}^{(k)}$ e $\boldsymbol{x}^{(k-1)}$, mentre per il metodo di Gauss-Seidel è sufficiente disporre di un solo vettore in cui si sostituiscono le componenti via via che si calcolano.

La seguente implementazione schematica in Matlab del metodo di Jacobi parte da un vettore iniziale `x` e costruisce in non più di `maxit` iterazioni il vettore soluzione `y`. Se in uscita `dif` < `tol` vi e' stata convergenza.

```
dif=1; k=0;
while dif>tol & k<maxit
    k=k+1;
    y=b;
    for i=1:n
        for j=[1:i-1,i+1:n] y(i)=y(i)-a(i,j)*x(j); end;
        y(i)=y(i)/a(i,i);
    end;
    dif=max(abs(y-x));
    x=y;
end
```

Per il metodo di Gauss-Seidel invece si ha

```
dif=1; k=0;
while dif>tol & k<maxit
    k=k+1; dif=0;
    for i=1:n
        z=b(i);
        for j=[1:i-1,i+1:n] z=z-a(i,j)*x(j); end;
        z=z/a(i,i);
        dif=max(dif,abs(x(i)-z));
        x(i)=z;
    end;
end
```

In molte applicazioni il metodo di Gauss-Seidel, che utilizza immediatamente i valori calcolati nella iterazione corrente, risulta più veloce del metodo di Jacobi. Però esistono casi in cui risulta non solo che il metodo di Jacobi sia più veloce del metodo di Gauss-Seidel, ma anche che il metodo di Jacobi sia convergente e quello di Gauss-Seidel no.

Esempi. Si esamina la convergenza dei metodi di Jacobi e di Gauss-Seidel applicati al sistema $A\boldsymbol{x} = \boldsymbol{b}$, per diverse matrici $A \in \boldsymbol{R}^{3\times 3}$. Il vettore $\boldsymbol{b}$ è sempre scelto in modo che la soluzione sia $\boldsymbol{x}^* = [1, 1, 1]^T$. Nelle figure sono riportati i grafici delle norme degli errori assoluti $\|\boldsymbol{e}^{(k)}\|_\infty$ delle successioni ottenute a partire dal vettore iniziale $\boldsymbol{x}^{(0)} = \boldsymbol{0}$. Con i quadratini vuoti sono indicati gli errori generati dal metodo di Jacobi, con i quadratini pieni gli errori generati dal metodo di Gauss-Seidel.

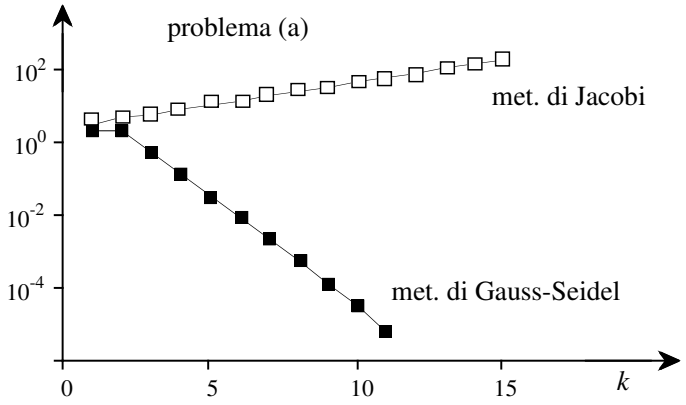
(a) Nel caso

$$A = \begin{bmatrix} 3 & 0 & 4 \\ 7 & 4 & 2 \\ -1 & -1 & -2 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 7 \\ 13 \\ -4 \end{bmatrix}$$

risulta

$$J = \begin{bmatrix} 0 & 0 & -4/3 \\ -7/4 & 0 & -1/2 \\ -1/2 & -1/2 & 0 \end{bmatrix} \quad G = \begin{bmatrix} 0 & 0 & -4/3 \\ 0 & 0 & 11/6 \\ 0 & 0 & -1/4 \end{bmatrix}$$

e $\rho(J) = 1.337510, \quad \rho(G) = 0.25$. Quindi il metodo di Gauss-Seidel è convergente mentre il metodo di Jacobi non lo è.



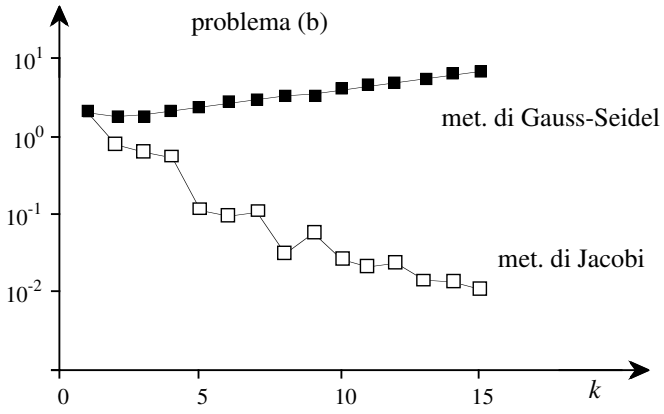
(b) Nel caso

$$A = \begin{bmatrix} -3 & 3 & -6 \\ -4 & 7 & -8 \\ 5 & 7 & -9 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} -6 \\ -5 \\ 3 \end{bmatrix}$$

risulta

$$J = \begin{bmatrix} 0 & 1 & -2 \\ 4/7 & 0 & 8/7 \\ 5/9 & 7/9 & 0 \end{bmatrix} \quad G = \begin{bmatrix} 0 & 1 & -2 \\ 0 & 4/7 & 0 \\ 0 & 1 & -10/9 \end{bmatrix}$$

e $\rho(J) = 0.8133091, \quad \rho(G) = 1.111111$. Quindi il metodo di Jacobi è convergente e il metodo di Gauss-Seidel non lo è.



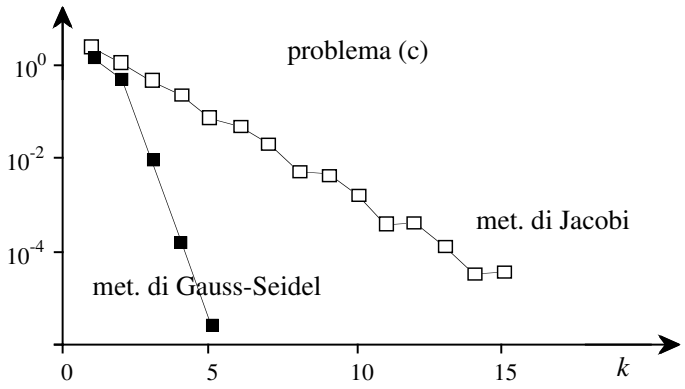
(c) Nel caso

$$A = \begin{bmatrix} 4 & 1 & 1 \\ 2 & -9 & 0 \\ 0 & -8 & -6 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 6 \\ -7 \\ -14 \end{bmatrix}$$

risulta

$$J = \begin{bmatrix} 0 & -1/4 & -1/4 \\ 2/9 & 0 & 0 \\ 0 & -4/3 & 0 \end{bmatrix} \quad G = \begin{bmatrix} 0 & -1/4 & -1/4 \\ 0 & -1/18 & -1/18 \\ 0 & 2/27 & 2/27 \end{bmatrix}$$

e $\rho(J) = 0.4438188, \quad \rho(G) = 0.01851852$. Quindi entrambi i metodi sono convergenti e la successione generata dal metodo di Gauss-Seidel converge più rapidamente di quella generata dal metodo di Jacobi.



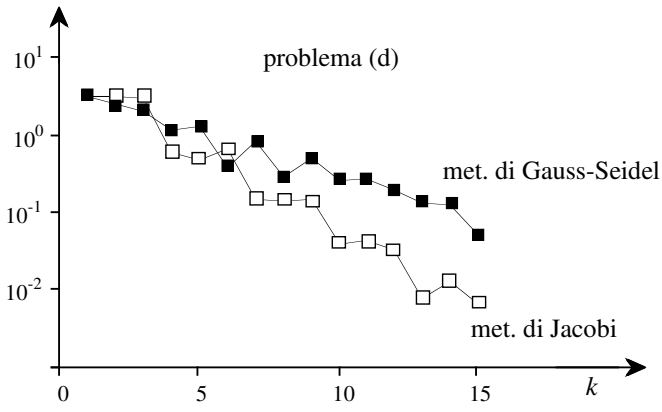
(d) Nel caso

$$A = \begin{bmatrix} 7 & 6 & 9 \\ 4 & 5 & -4 \\ -7 & -3 & 8 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 22 \\ 5 \\ -2 \end{bmatrix}$$

risulta

$$J = \begin{bmatrix} 0 & -6/7 & -9/7 \\ -4/5 & 0 & 4/5 \\ 7/8 & 3/8 & 0 \end{bmatrix} \quad G = \begin{bmatrix} 0 & -6/7 & -9/7 \\ 0 & 24/35 & 64/35 \\ 0 & -69/140 & -123/280 \end{bmatrix}$$

e $\rho(J) = 0.6411328, \quad \rho(G) = 0.7745967$. Quindi entrambi i metodi sono convergenti e la successione generata dal metodo di Jacobi converge più rapidamente di quella generata dal metodo di Gauss-Seidel.



□

Teorema. *Se la matrice A è a predominanza diagonale in senso stretto (per righe o per colonne), il metodo di Jacobi e il metodo di Gauss-Seidel convergono.*

Dim. Sia $A = M - N$ la decomposizione della matrice A corrispondente al metodo di Jacobi (cioè $M = D$ e $N = B + C$) o al metodo di Gauss-Seidel (cioè $M = D - B$ e $N = C$). Nelle ipotesi fatte, gli elementi principali di A sono non nulli, quindi la matrice M risulta non singolare. Un numero complesso λ è autovalore di $M^{-1}N$ se e solo se

$$\det(M^{-1}N - \lambda I) = 0,$$

ed essendo $M^{-1}N - \lambda I = -M^{-1}(\lambda M - N)$, per la regola di Binet λ è autovalore di $M^{-1}N$ se e solo se

$$\det(\lambda M - N) = 0. \tag{4.21}$$

La matrice $H = \lambda M - N$ ha gli elementi

$$h_{ij} = \begin{cases} \lambda a_{ij} & \text{se } i = j \\ a_{ij} & \text{se } i \neq j \end{cases} \qquad \text{per il metodo di Jacobi,}$$

$$h_{ij} = \begin{cases} \lambda a_{ij} & \text{se } i \geq j \\ a_{ij} & \text{se } i < j \end{cases} \qquad \text{per il metodo di Gauss-Seidel.}$$

Se $|\lambda| \geq 1$ si ha

$$|h_{ii}| = |\lambda|\,|a_{ii}| \quad \text{e} \quad |h_{ij}| \leq |\lambda|\,|a_{ij}|, \quad \text{per} \quad i \neq j,$$

e quindi anche la matrice H ha predominanza diagonale in senso stretto. In tal caso, per quanto visto nel par. 3.6, la matrice H è non singolare e quindi un λ, tale che $|\lambda| \geq 1$, non può verificare la (4.21), cioè non può essere autovalore di $M^{-1}N$. Ne segue che gli autovalori di $M^{-1}N$ hanno modulo minore di 1 e che i metodi di Jacobi e di Gauss-Seidel sono convergenti. □

4.12 Metodo delle potenze

Il *metodo delle potenze* è un classico metodo iterativo per approssimare l'autovalore di massimo modulo di una matrice e il corrispondente autovettore. Siano $\lambda_1, \lambda_2, \ldots, \lambda_n$ gli autovalori di A, che supponiamo ordinati in ordine non crescente di modulo, e $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ i corrispondenti autovettori. Il metodo delle potenze è convergente sotto le seguenti ipotesi:

- A ha un solo autovalore di modulo massimo, cioè $|\lambda_1| > |\lambda_2|$,
- gli autovettori $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ sono linearmente indipendenti, cioè costituiscono una base per $\boldsymbol{R}^n$.

Fissato un vettore $\boldsymbol{y} \in \mathbb{R}^n$, tale che $\|\boldsymbol{y}\|_\infty = 1$. Esprimiamo $\boldsymbol{y}$ come combinazione lineare degli autovettori

$$\boldsymbol{y} = \sum_{i=1}^{n} \alpha_i \boldsymbol{x}_i,$$

e supponiamo che $\alpha_1 \neq 0$. Si ha

$$A^k \boldsymbol{y} = \sum_{i=1}^{n} \alpha_i A^k \boldsymbol{x}_i = \sum_{i=1}^{n} \alpha_i \lambda_i^k \boldsymbol{x}_i = \lambda_1^k \left[\alpha_1 \boldsymbol{x}_1 + \sum_{i=2}^{n} \alpha_i \Big(\frac{\lambda_i}{\lambda_1}\Big)^k \boldsymbol{x}_i \right].$$

Per $i \geq 2$ è $|\lambda_i/\lambda_1| < 1$, quindi per k sufficientemente grande le potenze $(\lambda_i/\lambda_1)^k$ si possono trascurare e risulta

$$A^k \boldsymbol{y} \sim \lambda_1^k \alpha_1 \boldsymbol{x}_1, \tag{4.22}$$

dove con la notazione $\boldsymbol{v}^{(k)} \sim \boldsymbol{w}^{(k)}$ per $\boldsymbol{v}^{(k)}, \boldsymbol{w}^{(k)} \in \mathbb{R}^n$ si intende che $\lim_{k\to\infty} v_j^{(k)}/w_j^{(k)} = 1$ per tutte le componenti di indice j che non tendono a zero quando $k \to \infty$ e $\lim_{k\to\infty} v_j^{(k)} = \lim_{k\to\infty} w_j^{(k)} = 0$ altrimenti.

Poniamo $\boldsymbol{t}^{(0)} = \boldsymbol{y}$ e costruiamo la successione

$$\left.\begin{array}{ll} \mathbf{u}^{(k)} & = A\mathbf{t}^{(k-1)}, \\ \mathbf{t}^{(k)} & = \dfrac{\mathbf{u}^{(k)}}{\beta_k} \end{array}\right\}, \quad k = 1, 2, \ldots, \tag{4.23}$$

dove β_k è una componente di massimo modulo di $\boldsymbol{u}^{(k)}$, cioè tale che

$$\beta_k = u_m^{(k)}, \quad \text{con} \quad |u_m^{(k)}| = \max_{j=1,\ldots,n} |u_j^{(k)}| = \|\boldsymbol{u}^{(k)}\|_\infty.$$

Il vettore $\boldsymbol{t}^{(k)}$ ottenuto con la (4.23) è quindi tale che $t_m^{(k)} = 1$. Per k sufficientemente grande risulta

$$\boldsymbol{t}^{(k-1)} = \frac{A^{k-1}\boldsymbol{t}^{(0)}}{\beta_{k-1}\dots\beta_1} \sim \frac{\lambda_1^{k-1}\alpha_1\boldsymbol{x}_1}{\beta_{k-1}\dots\beta_1}, \quad \boldsymbol{u}^{(k)} = \frac{A^k\boldsymbol{t}^{(0)}}{\beta_{k-1}\dots\beta_1} \sim \frac{\lambda_1^k\alpha_1\boldsymbol{x}_1}{\beta_{k-1}\dots\beta_1}$$

Per ogni indice j per cui la j-esima componente di $\boldsymbol{x}_1$ è diversa da 0 si ha

$$\lim_{k\to\infty} \frac{u_j^{(k)}}{t_j^{(k-1)}} = \lambda_1.$$

Si può assumere che da una certa iterazione in poi l'indice m, corrispondente a una componente di massimo modulo di $\boldsymbol{u}^{(k)}$, resti sempre lo stesso. Allora

$$\lim_{k\to\infty} \beta_k = \lim_{k\to\infty} u_m^{(k)} = \lim_{k\to\infty} \frac{u_m^{(k)}}{t_m^{(k-1)}} = \lambda_1.$$

Inoltre la successione $\boldsymbol{t}^{(k)}$ converge all'autovettore $\boldsymbol{x}_1$ normalizzato in norma ∞.

L'errore che si commette approssimando λ_1 con β_k tende a zero come $|\lambda_2/\lambda_1|^k$. Quindi la velocità di convergenza sarà tanto maggiore quanto più $|\lambda_1|$ è grande rispetto a $|\lambda_2|$.

La condizione $\alpha_1 \neq 0$ è, in teoria, necessaria per poter scrivere la (4.22) e quindi per avere la convergenza della successione β_k a λ_1. Infatti, se fosse $\alpha_1 = 0$, $\alpha_2 \neq 0$ e $|\lambda_2| > |\lambda_3|$, risulterebbe $A^k\boldsymbol{y} \sim \lambda_2\alpha_2\boldsymbol{x}_2$. In pratica però, anche se fosse $\alpha_1 = 0$, per la presenza degli errori di arrotondamento, i vettori $\boldsymbol{t}^{(k)}$ effettivamente calcolati sarebbero comunque rappresentabili come combinazioni lineari degli autovettori con una componente non nulla rispetto a $\boldsymbol{x}_1$. Perciò la successione effettivamente calcolata convergerebbe ugualmente a λ_1. Inoltre nel caso che le componenti del vettore $\boldsymbol{y}$ vengano scelte casualmente, la probabilità di ottenere un vettore per cui $\alpha_1 = 0$ è praticamente nulla.

Si può dimostrare che il metodo delle potenze è convergente anche nel caso in cui

$$\lambda_1 = \lambda_2 = \ldots = \lambda_r, \quad \text{purché} \quad |\lambda_r| > |\lambda_{r+1}|,$$

mentre non converge se A ha più autovalori di modulo massimo diversi fra loro, come nel caso che vi siano due autovalori complessi coniugati di modulo massimo.

Fissata una tolleranza ϵ, come condizione di arresto del metodo iterativo (4.23) si può utilizzare la seguente

$$|\beta_{k+1} - \beta_k| < \epsilon. \tag{4.24}$$

Poiché in generale non si può sapere a priori se tutti gli autovalori di modulo massimo coincidono, oltre alla (4.24) si deve imporre anche una condizione di arresto sul numero massimo *maxit* di iterazioni. La seguente implementazione schematica parte da un vettore $\boldsymbol{t}^{(0)}$ con tutte le componenti 1 e fornisce in uscita il vettore `b` delle iterate.

```
t=ones(n,1); dif=1; i=1; b(1)=0;
while dif>tol & i<maxit
    i=i+1;
    t=a*t;
    [m,j]=max(abs(t));
    b(i)=t(j);
    t=t/t(j);
    dif=abs(b(i)-b(i-1));
end
```

Esempio Consideriamo la matrice

$$A = \begin{bmatrix} 15 & -2 & 2 \\ 1 & 10 & -3 \\ -2 & 1 & 0 \end{bmatrix}$$

che, come si è visto nel par. 3.6, ha due autovalori λ_1 e λ_2 in

$$\{\ z \in \mathbf{C} : |z - 15| \leq 3\ \} \cup \{\ z \in \mathbf{C} : |z - 10| \leq 3\ \},$$

e un autovalore λ_3 in

$$\{\ z \in \mathbf{C} : |z| \leq 3\ \}.$$

Si applica il metodo delle potenze a partire dal vettore $\mathbf{y}^{(0)} = [1, 1, 1]^T$. Con il criterio di arresto (4.24) e tolleranza $\epsilon = 10^{-6}$, il metodo si arresta al 41-esimo passo, fornendo le seguenti successioni di valori che approssimano l'autovalore λ_1 e l'autovettore corrispondente:

k	β_k	$\boldsymbol{t}_k^{(k)T}$		
1	15.0000	1.00000	0.53333	−0.06667
2	13.8000	1.00000	0.47343	−0.10628
3	13.8406	1.00000	0.43735	−0.11030
4	13.9047	1.00000	0.41025	−0.11238
⋮	⋮	⋮	⋮	⋮
41	14.1026	1.00000	0.33032	−0.11839

Ripetiamo il calcolo con un'altra matrice

$$B = \begin{bmatrix} 1 & -2 & 2 \\ 1 & -1 & -3 \\ -2 & 1 & 0 \end{bmatrix}$$

Si ottiene la successione β_k

$$\{0,\ -3.0000,\ -2.3333,\ -2.7143,\ 2.6842,\ 3.5490,\ \ldots,\ -2.5758,\ -2.9830,\ -3.4138\}$$

È evidente che non c'è convergenza. La causa della mancata convergenza è che B ha due autovalori complessi e coniugati $\lambda_1 = 0.5866 + 3.0054\,i$ e $\lambda_2 = 0.5866 - 3.0054\,i$ e un terzo autovalore $\lambda_3 = -1.1732$ di modulo minore. □

Per approssimare l'autovalore di modulo minimo di A si può applicare il metodo delle potenze alla matrice inversa A^{-1}. Infatti, sia λ_n l'autovalore di modulo minimo di A, allora $1/\lambda_n$ è l'autovalore di modulo massimo di A^{-1}. Si ha convergenza verso $1/\lambda_n$ se λ_n è l'unico autovalore di A di modulo minimo.

Esempio Applichiamo il metodo delle potenze per approssimare l'autovalore di modulo minimo della matrice A data sopra. Si ottiene la successione β_k

$$\{2.1600,\ 1.9595,\ 1.9530,\ 1.9528,\ 1.9528\}$$

da cui si ricava che l'autovalore di modulo minimo è $\lambda_3 = 1/\beta_5 = 0.5121$. □

Per approssimare un autovalore λ_i di modulo intermedio di A, occorre avere una stima ℓ, anche grossolana, di λ_i. Si applica il metodo delle potenze alla matrice inversa $(A - \ell I)^{-1}$. Se ℓ è più vicino a λ_i che a λ_j per ogni $j \neq i$, allora la matrice $(A - \ell I)^{-1}$ ha l'autovalore $1/(\lambda_i - \ell)$ come unico autovalore di modulo massimo e il metodo delle potenze converge.

Esempio Si stima che $\ell = 8$ sia più vicino all'autovalore intermedio della matrice A data sopra che agli altri due autovalori. Applicando il metodo alla matrice $(A - 8\,I)^{-1}$ si ottiene la successione β_k

$$\{0.2294,\ 0.3097,\ 0.3382,\ 0.4241,\ 0.4072,\ 0.4189,\ 0.4177,\ 0.4191,\ 0.4190\}$$

da cui si ricava che $\lambda_2 = 8 + 1/\beta_9 = 10.3865$. □

Con questa tecnica è possibile accelerare la convergenza anche per l'approssimazione dell'autovalore di modulo massimo.

Esempio Dalle prime due iterazioni del metodo delle potenze applicato direttamente alla matrice A si stima che l'autovalore di modulo massimo debba essere vicino a 14. Si applica allora il metodo alla matrice $(A - 14\,I)^{-1}$. Si ottiene la successione β_k

$$\{9.4000,\ 9.7830,\ 9.7497,\ 9.7508,\ 9.7508\}$$

da cui si ricava, con molte meno delle 41 iterazioni richieste dal metodo applicato alla A, che $\lambda_1 = 14 + 1/\beta_5 = 14.1026$. □

Capitolo 5

Interpolazione e quadratura

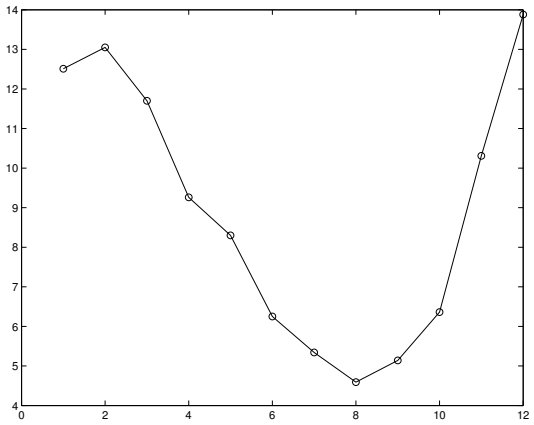
Un problema che frequentemente si presenta in matematica applicata è quello dell'*approssimazione* di funzioni, che consiste nel determinare una funzione $g(x)$, appartenente ad una classe prescelta di funzioni, che meglio approssima una funzione data $f(x)$. Nel caso *discreto*, che esamineremo qui, la funzione $f(x)$ è nota su un insieme di $n+1$ *nodi* $x_0, x_1, \ldots, x_n$, appartenenti ad un intervallo $[a, b]$. Questo è un caso caratteristico dell'analisi di dati sperimentali: talvolta il numero $n+1$ di dati disponibili è piccolo, come nello studio di certi fenomeni fisici o biologici, talvolta è elevato, come nelle analisi statistiche. Si può anche supporre che i valori $f(x_i)$, in particolare quando sono rilevati in laboratorio, siano affetti da errori di misura. La funzione $g(x)$ che si costruisce serve per approssimare valori della $f(x)$ in punti diversi dai nodi o per dedurne proprietà di comportamento.

Al caso discreto può essere ricondotto anche il caso *continuo*, in cui la funzione $f(x)$ è una funzione nota della matematica, in generale non razionale. Lo scopo è quello di produrre una funzione $g(x)$ più semplice, cioè più facilmente "trattabile" (ad esempio più facilmente calcolabile, derivabile o integrabile) della $f(x)$. In questo caso i nodi sono scelti in modo opportuno e i valori $f(x_i)$ sono calcolati con elevata accuratezza, sfruttando le proprietà note della $f(x)$.

Esempio. I seguenti dati si riferiscono alla portata d'acqua di un fiume italiano, misurata mensilmente in m^3/sec.

mese	G	F	M	A	M	G	L	A	S	O	N	D
portata	12.5	13.1	11.7	9.3	8.3	6.3	5.3	4.6	5.1	6.4	10.3	13.9

Disegniamo il grafico di questa funzione, congiungendo con una spezzata i punti assegnati. Supponiamo che le misure siano state prese tutte il primo del mese. Ci chiediamo quale è stata la portata d'acqua il 10 di maggio o il 15 di luglio. La risposta più ovvia è quella di cercare il valore corrispondente sulla spezzata. In pratica in ogni intervallo la funzione incognita che descrive la portata d'acqua viene approssimata con un polinomio lineare, il cui grafico è il segmento congiungente due punti consecutivi. È possibile ottenere delle stime migliori se invece che prendere



in considerazione solo i due dati più vicini si utilizza un maggior numero di dati, ed eventualmente tutti i dati a disposizione, tenendo conto dell'andamento complessivo della funzione? Questa domanda può essere formulata anche così: esistono approssimazioni di una funzione migliori dell'approssimazione lineare?

Noi esamineremo qui il caso dell'approssimazione *polinomiale*: la funzione $g(x)$ con cui si vuole approssimare la funzione $f(x)$ è un polinomio di grado opportuno, i cui coefficienti vengono determinati in modo che l'approssimazione sia la migliore possibile, compatibilmente con i dati che si hanno a disposizione. Per rendere formalmente corretta questa espressione "la migliore possibile", occore definire come si misura la distanza fra la $f(x)$ e la $g(x)$. La $g(x)$ sarà poi determinata in modo da avere la minima distanza possibile dalla $f(x)$.

Nel caso del problema discreto che stiamo considerando, la distanza fra la $f(x)$ e la $g(x)$ viene misurata per mezzo del vettore $\boldsymbol{r}$ degli *scarti*, cioè delle differenze

$$r_i = f(x_i) - g(x_i), \quad \text{per} \quad i = 0, \ldots, n.$$

I coefficienti cercati sono quelli che rendono minima la $\|\boldsymbol{r}\|_2$. Il metodo corrispondente viene detto dei *minimi quadrati*.

5.1 Il problema dell'interpolazione.

Un caso particolarmente importante del problema dei minimi quadrati è quello in cui $\|\boldsymbol{r}\|_2$ è zero. In questo caso il polinomio $g(x)$, che più propriamente viene chiamato *polinomio di interpolazione*, soddisfa la condizione $g(x_i) = f(x_i)$, per $i = 0, \ldots, n$. Vale il seguente teorema.

Teorema. *Se $x_i \neq x_j$ per $i \neq j$, esiste ed è unico il polinomio*

$$p_n(x) = a_0 x^n + a_1 x^{n-1} + \ldots + a_{n-1} x + a_n,$$

di grado al più n, tale che

$$p_n(x_i) = f(x_i), \quad i = 0, 1, \ldots, n. \tag{5.1}$$

$p_n(x)$ è quindi il polinomio di interpolazione della funzione $f(x)$.

Dim. Un metodo per calcolare il vettore $\boldsymbol{a} = [a_0, a_1, \ldots, a_n]^T$ dei coefficienti del polinomio $p_n(x)$ è il seguente. Si considerano il vettore $\boldsymbol{f} = [f(x_0), f(x_1), \ldots, f(x_n)]^T$ e la matrice V, detta matrice di *Vandermonde*, così definita

$$v_{ij} = x_{i-1}^{n-j+1}, \quad i = 1, \ldots n+1, \ j = 1, \ldots, n+1.$$

Imponendo che $p_n(x)$ verifichi le $n+1$ condizioni (5.1), si ottiene il sistema lineare di $n+1$ equazioni in $n+1$ incognite

$$V\boldsymbol{a} = \boldsymbol{f}. \tag{5.2}$$

La matrice di Vandermonde è non singolare, in quanto

$$\det\ V = \prod_{\substack{i,j=0 \\ i>j}}^{n} (x_i - x_j),$$

e i punti x_i sono per ipotesi a due a due distinti. Ne segue che il sistema (5.2) ha una e una sola soluzione e quindi il polinomio $p_n(x)$ esiste ed è unico.

Solo raramente del polinomio di interpolazione sono richiesti i coefficienti, in generale ciò che si vuole è il valore di $p_n(x)$ in uno o più punti. Per questo non è conveniente risolvere il sistema (5.2) che richiederebbe un numero di operazioni moltiplicative dell'ordine di $n^3/3$. Inoltre la matrice di Vandermonde per certe scelte dei nodi x_i può essere malcondizionata.

Esempio. Per $n \geq 4$ il polinomio di interpolazione della funzione che nei punti $x_i = i/n$, $i = 0, \ldots, n$ assume i valori $f(x_i) = x_i^4 + 0.1$ è $p_n(x) = x^4 + 0.1$. Per determinare i coefficienti di tale polinomio si scrive e si risolve il sistema (5.2)). La seguente tabella riporta al crescere di n il numero di condizionamento $\mu_\infty(V)$ della matrice di Vandermonde e il massimo errore assoluto ϵ da cui sono affetti i coefficienti effettivamente calcolati utilizzando il metodo di Gauss.

n	$\mu_\infty(V)$	ϵ
5	$1.04\ 10^3$	$6.40\ 10^{-6}$
6	$8.06\ 10^3$	$4.37\ 10^{-5}$
7	$6.30\ 10^4$	$1.07\ 10^{-4}$
8	$4.96\ 10^5$	$1.64\ 10^{-3}$
9	$3.92\ 10^6$	$3.31\ 10^{-2}$
10	$3.11\ 10^7$	$1.24\ 10^{-1}$

È evidente che già per $n = 10$ i coefficienti ottenuti sono privi di significato. Cambiando i punti x_i il condizionamento della matrice V cambia e quindi è possibile che si ottengano risultati migliori, anche se raramente soddisfacenti. □

Più conveniente dal punto di vista della stabilità e del costo computazionale è scrivere il polinomio di interpolazione (che ricordiamo è unico) nella seguente forma di *Lagrange*

$$p_n(x) = \sum_{j=0}^{n} f(x_j) L_j(x), \quad \text{dove} \quad L_j(x) = \prod_{\substack{k=0 \\ k \neq j}}^{n} \frac{x - x_k}{x_j - x_k}. \tag{5.3}$$

Per controllare che il polinomio così scritto verifichi le (5.1), notiamo che

$$L_j(x_i) = \begin{cases} 1 & \text{se } i = j, \\ \\ 0 & \text{se } i \neq j, \end{cases}$$

e quindi

$$p_n(x_i) = \sum_{j=0}^{n} f(x_j) L_j(x_i) = f(x_i), \quad k = 0, \ldots, n.$$

Il numero di operazioni richieste per calcolare $p_n(x)$ in corrispondenza ad un valore $x \neq x_i$ è dell'ordine di n^2.

Esempio. Utilizzando la (5.3) calcoliamo i valori di $p_n(x)$ dell'esempio del fiume corrispondenti al 10 di maggio, cioè $x = 5.33$ e al 15 di luglio, cioè $x = 7.5$. Si ottengono i valori $p_n(5.33) = 7.6228$ e $p_n(7.5) = 4.9357$. □

Il caso più semplice è quello dell'interpolazione lineare, cioè quando $n = 1$. Il polinomio di grado al più 1 che assume nei punti x_0, x_1, distinti tra loro, rispettivamente i valori $f(x_0)$, $f(x_1)$ è il seguente

$$\begin{aligned} p_1(x) &= f(x_0) \frac{x - x_1}{x_0 - x_1} + f(x_1) \frac{x - x_0}{x_1 - x_0} \\ &= \frac{1}{h} \left(f(x_1)(x - x_0) - f(x_0)(x - x_1), \quad \text{dove} \quad h = x_1 - x_0. \right. \end{aligned} \tag{5.4}$$

Il polinomio $p_1(x)$ ha come grafico il segmento che congiunge i punti $(x_0, f(x_0))$ e $(x_1, f(x_1))$.

Esempio. L'interpolazione lineare è usata spesso per determinare il valore che una funzione tabulata assume in un punto diverso da quelli riportati nelle tavole. Ad esempio, si vuole approssimare il valore di $\tan 1.354$, conoscendo i valori $\tan 1.35 = 4.4552$ e $\tan 1.36 = 4.6734$, tratti dalle tavole delle funzioni trigonometriche, si può utilizzare l'interpolazione lineare sui nodi $x_0 = 1.35$ e $x_1 = 1.36$. Il valore $p_1(1.354) = 4.542$ ottenuto applicando la (5.4) si assume come valore approssimato di $\tan 1.354$. □

5.2 Resto dell'interpolazione.

Si definisce *resto* dell'interpolazione di $f(x)$ con il polinomio $p_n(x)$ la funzione

$$r(x) = f(x) - p_n(x).$$

Il resto misura l'errore che si commette quando si approssima $f(x)$ con $p_n(x)$. Chiaramente $r(x)$ è nullo nei nodi x_i. Vale il seguente teorema.

Teorema. *Posto* $a = \min_{i=0,\dots,n} x_i$ *e* $b = \max_{i=0,\dots,n} x_i$, *se* $f \in C^{n+1}[a,b]$, *esiste un punto* $\xi = \xi(x) \in (a,b)$, *tale che*

$$r(x) = (x - x_0)(x - x_1)\cdots(x - x_n)\,\frac{f^{(n+1)}(\xi)}{(n+1)!}. \tag{5.5}$$

Dim. Per $x = x_i$, $i = 0, \dots, n$, è

$$r(x_i) = \pi_n(x_i) = 0,$$

e quindi la (5.5)) è verificata. Per $x \neq x_i$ sia

$$s(x) = \frac{r(x)}{\pi_n(x)}\,, \tag{5.6}$$

e si consideri la funzione della variabile y

$$z(y) = r(y) - s(x)\pi_n(y).$$

La funzione $z(y)$ si annulla almeno negli $n+2$ punti distinti x_i, $i = 0, \dots, n$ e x. Inoltre, poiché $z(y)$ è derivabile almeno $n+1$ volte, in quanto $f(y) \in C^{n+1}[a,b]$, per il teorema di Rolle la funzione $z'(y)$ si annulla in almeno $n+1$ punti distinti interni ad $[a,b]$, la $z''(y)$ si annulla in almeno n punti distinti interni ad $[a,b]$, …, la $z^{(n+1)}(y)$ si annulla in almeno un punto $\xi \in (a,b)$. Allora si ha:

$$0 = z^{(n+1)}(\xi) = r^{(n+1)}(\xi) - (n+1)!\,s(x) = f^{(n+1)}(\xi) - (n+1)!\,s(x), \tag{5.7}$$

in quanto il polinomio $p_n(x)$ ha la derivata $(n+1)$-esima identicamente nulla. Ricavando $s(x)$ dalla (5.7) e sostituendolo nella (5.6), si ottiene la (5.5) per $x \neq x_i$, $i = 0, \dots, n$. □

Il resto dell'interpolazione lineare è dato da

$$r(x) = (x - x_0)(x - x_1)\,\frac{f''(\xi)}{2}\,, \quad \xi \in (x_0, x_1). \tag{5.8}$$

Posto $M_2 = \max_{x\in(x_0,x_1)} |f''(x)|$ e $h = x_1 - x_0$, si ha

$$\max_{x\in(x_0,x_1)} |(x - x_0)(x - x_1)| = \frac{h^2}{4} \quad \text{e} \quad |r(x)| \leq \frac{M_2 h^2}{8}\,.$$

Nel caso dell'interpolazione lineare della funzione $f(x) = \tan x$, è $f''(x) = \dfrac{2\sin x}{\cos^3 x}$, e per $x \in [1.35, 1.36]$ è $|r(x)| \leq 0.267\ 10^{-2}$. In realtà, confrontando con i valori forniti dal software per la funzione $f(x)$, risulta che $|r(1.354)| \sim 0.23\ 10^{-2}$.

Esempio. Per approssimare la radice quadrata di un numero x, che non sia un quadrato perfetto, si può procedere nel modo seguente: si considerano le radici quadrate dei numeri x_i, $i = 0, \ldots, n$, più vicini a x, che sono quadrati perfetti, e si costruisce il polinomio di interpolazione che in x_i assume il valore $\sqrt{x_i}$. La radice cercata viene approssimata con il valore assunto dal polinomio di interpolazione in x. Approssimiamo con questa tecnica $\sqrt{0.6}$.

Cominciamo con l'approssimazione lineare sui nodi $x_0 = 0.49$ e $x_1 = 0.64$, in cui la radice vale $f(x_0) = 0.7$ e $f(x_1) = 0.8$. L'interpolazione lineare dà il valore $p_1(0.6) = 0.7733$. Per $x \in [0.49, 0.64]$ il resto dell'interpolazione è

$$r(x) = \frac{(x - 0.49)(x - 0.64)}{8\sqrt{\xi^3}}, \quad \xi \in (0.49, 0.64).$$

Si ha perciò

$$|r(0.6)| < \frac{0.44\ 10^{-2}}{8\sqrt{0.49^3}} \approx 1.89\ 10^{-3}.$$

In realtà, confrontando con il valore fornito dal software, risulta $|\sqrt{0.6} - p_2(0.6)| \approx 1.3\ 10^{-3}$. Infatti l'approssimazione ottenuta ha esatte le prime due cifre.

Si aumenta il numero dei punti, considerando $x_0 = 0.49$, $x_1 = 0.64$, $x_2 = 0.81$, in cui la radice vale $f(x_0) = 0.7$, $f(x_1) = 0.8$ e $f(x_2) = 0.9$. Con il polinomio di Lagrange si trova $p_2(0.6) = 0.7744$. Per $x \in [0.49, 0.81]$ il resto dell'interpolazione è

$$r(x) = \frac{(x - 0.49)(x - 0.64)(x - 0.81)}{16\sqrt{\xi^5}}, \quad \xi \in (0.49, 0.81).$$

Si ha perciò

$$|r(0.6)| < \frac{0.93\ 10^{-3}}{16\sqrt{0.49^5}} \approx 3.46\ 10^{-4}.$$

In realtà risulta $|\sqrt{0.6} - p_2(0.6)| \approx 2.64\ 10^{-4}$. Sono esatte le prime 3 cifre di $p_2(0.6)$. Nella figura è riportato il grafico della maggiorazione di $|r(x)|$ ottenuta sostituendo 0.49 al posto di ξ.

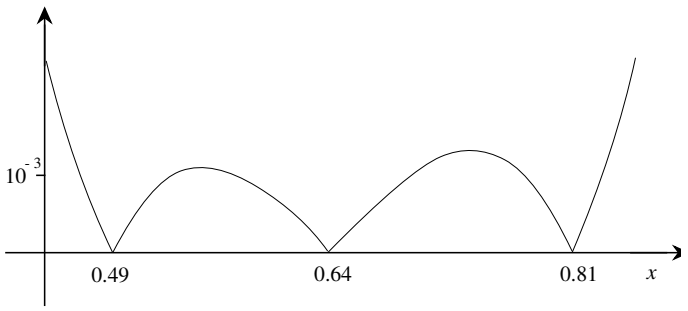
Aumentiamo ancora il numero dei nodi. Assumendo $x_0 = 0.36$, $x_1 = 0.49$, $x_2 = 0.64$, $x_3 = 0.81$, si ottiene $p_3(0.6) = 0.7746$. Per $x \in [0.36, 0.81]$ il resto dell'interpolazione è

$$r(x) = \frac{5(x - 0.36)(x - 0.49)(x - 0.64)(x - 0.81)}{128\sqrt{\xi^7}}, \quad \xi \in (0.36, 0.81).$$

Si ha perciò

$$|r(0.6)| < \frac{0.11\ 10^{-2}}{128\sqrt{0.36^7}} \approx 1.65\ 10^{-4}.$$

In realtà $|\sqrt{0.6} - p_3(0.6)| \approx 6.4\ 10^{-5}$. □



5.3 Studio del resto

Per quanto visto sopra sembrerebbe di poter concludere che all'aumentare del numero dei nodi il polinomio di interpolazione fornisca approssimazioni sempre migliori. Prima di poter fare una simile affermazione, conviene studiare più accuratamente il resto. Posto

$$M_{n+1} = \max_{x\in[a,b]} |f^{(n+1)}(x)|,$$

si ha

$$r_{\max} = \max_{x\in[a,b]} |r(x)| \le \max_{x\in[a,b]} |(x-x_0)(x-x_1)\cdots(x-x_n)| \ \frac{M_{n+1}}{(n+1)!}. \tag{5.9}$$

Per semplicità consideriamo il caso di nodi x_i equidistanti con passo h, cioè $x_i = x_0 + ih$, per $i = 0, \ldots, n$. Ponendo $x = x_0 + th$, si ha

$$r_{\max} \le \max_{t\in[0,n]} |\pi_n(t)| \ h^{n+1} \frac{M_{n+1}}{(n+1)!}, \quad \text{dove} \quad \pi_n(t) = t(t-1)\cdots(t-n).$$

Se $f^{(n+1)}(x)$ non varia molto nell'intervallo $[a, b]$ si può avere un'idea del comportamento del resto studiando il comportamento del polinomio $\pi_n(t)$, i cui zeri sono $0, \ldots, n$.

Per $n = 5$ e $n = 8$ si ottengono grafici delle figure 5.1 e 5.2. È facile vedere che il punto $n/2$ è punto di simmetria per $|\pi_n(t)|$ e che i massimi relativi di $|\pi_n(t)|$ in ogni intervallo $(i, i+1)$ crescono quando ci si allontana dal centro dell'intervallo $[0, n]$ verso gli estremi. Quindi se $f^{(n+1)}$ non varia molto nell'intervallo $[a, b]$, il resto risulta minore nella parte centrale dell'intervallo. Il massimo di $|\pi_n(t)|$, che viene assunto nel primo e nell'ultimo sottointervallo, al crescere di n cresce in maniera esponenziale. Lo studio del comportamento del resto suggerisce perciò di utilizzare il polinomio di interpolazione per approssimare $f(x)$ solo per valori bassi del grado n e per punti x che si trovino nella parte centrale dell'intervallo $[a, b]$.

Si potrebbe pensare che la presenza del fattore $h^{n+1}/(n+1)!$ nell'espressione del resto faccia comunque diminuire $r_{\max}$ al crescere di n, ma questo non è sempre vero: la convergenza a zero di $r_{\max}$ dipende da come cresce M_{n+1}. Vi sono infatti esempi di funzioni $f(x)$, anche infinitamente derivabili, per le quali la successione $\{p_n(x)\}$

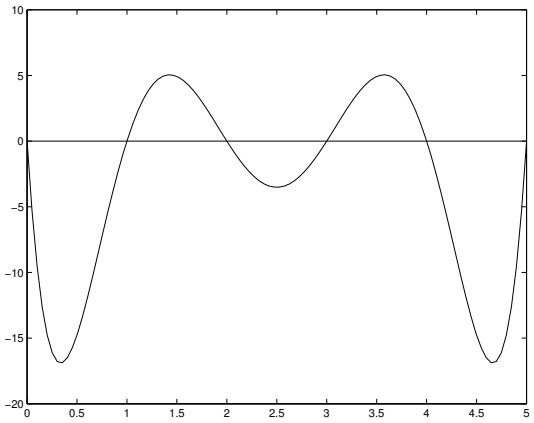


Figura 5.1: Grafico di $\pi_5(t)$ per $t \in [0, 5]$.

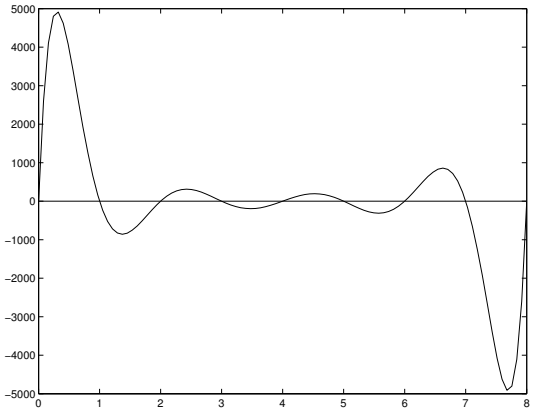


Figura 5.2: Grafico di $\pi_8(t)$ per $t \in [0, 8]$.

dei valori assunti dal polinomio di interpolazione di grado n in un punto $x \in (a, b)$ non converge a $f(x)$ per n che tende all'infinito.

Esempio (la funzione di *Runge*)

$$f(x) = \frac{1}{1 + x^2}\,, \qquad x \in [a, b] = [-5, 5].$$

Sia $p_n(x)$ il polinomio di interpolazione sui nodi $x_i = a+i(b-a)/n$, $i = 0, \ldots, n$. Per $n = 5$ si ottiene il grafico della figura 5.3, in cui il polinomio di interpolazione $p_n(x)$ (linea tratteggiata) è plottato rispetto alla funzione $f(x)$ (linea spessa). Ripetiamo il calcolo con $n = 10$. Si ottiene il grafico della figura 5.4.

Chiaramente il polinomio di grado 10 è una cattiva approssimazione di $f(x)$ quando x è fuori dalla zona centrale dell'intervallo. Anzi, si può dimostrare che al

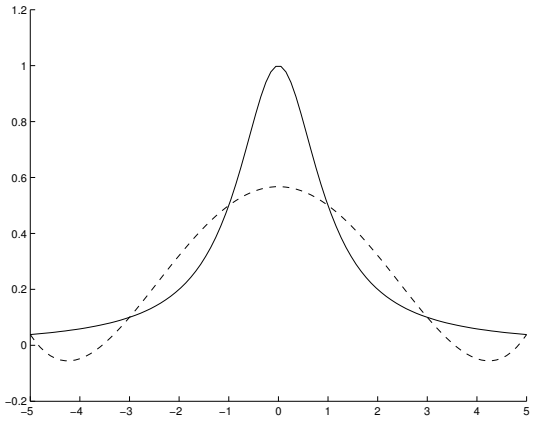


Figura 5.3: Polinomio di interpolazione di grado 5 della funzione di Runge.

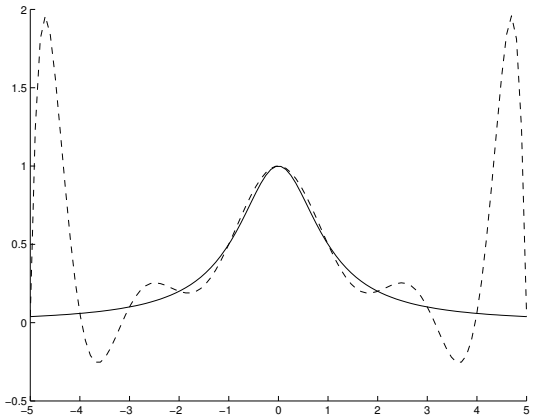


Figura 5.4: Polinomio di interpolazione di grado 10 della funzione di Runge.

crescere di n la successione $\{p_n(x)\}$ non converge puntualmente a $f(x)$ e che i corrispondenti resti diventano in modulo arbitrariamente grandi in punti dell'intervallo $[-5, 5]$. □

Riprendiamo adesso l'esempio del fiume esaminato all'inizio del capitolo e sovrapponiamo ai valori della funzione il grafico del polinomio di interpolazione. Si ottiene la figura 5.5. È evidente che il polinomio di interpolazione non rappresenta affatto una buona approssimazione della funzione discreta al di fuori della zona centrale. Ciò è dovuto principalmente al fatto che il polinomio è di grado troppo elevato e quindi ha troppe oscillazioni.

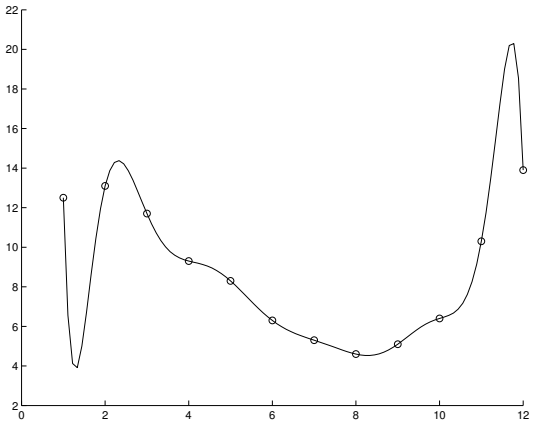


Figura 5.5: Polinomio di interpolazione della funzione dell'esempio del fiume.

5.4 Errori di arrotondamento del polinomio di interpolazione.

Nell'esame dell'accuratezza dell'approssimazione di $f(x)$ con un polinomio di interpolazione $p_n(x)$ occorre tener conto, oltre che dell'errore analitico

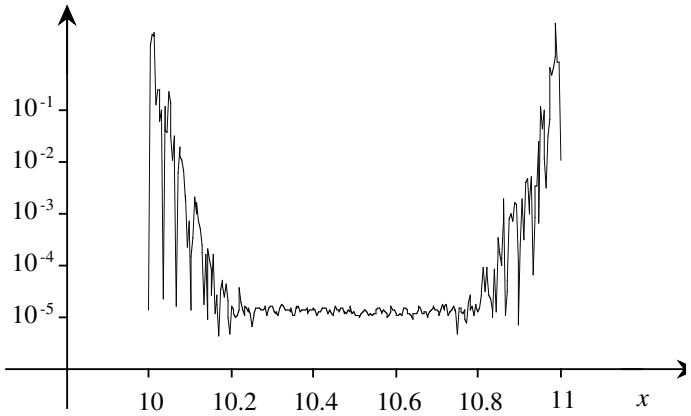
$$\epsilon_{an} = -\frac{r(x)}{f(x)}, \quad f(x) \neq 0,$$

anche dell'errore algoritmico generato nel calcolo di $p_n(x)$, che può diventare preponderante rispetto all'errore analitico e che dipende dalla particolare espressione del polinomio di interpolazione che viene usata.

Come esempio, nella figura è riportato il grafico dei moduli degli errori relativi effettivamente generati nel calcolo del polinomio di interpolazione di Lagrange per la funzione $f(x) = \log x$ nell'intervallo [10,11], assumendo come nodi i punti equidistanti $x_i = 10 + ih$, $h = 1/29$, $i = 0, \ldots, 29$. Nell'intervallo $[10, 11]$ l'errore analitico, per la (5.9)), può essere così maggiorato

$$|\epsilon_{an}| = \tfrac{|r(x)|}{|f(x)|} \leq \frac{|b-a|^{n+1}}{\min\limits_{x\in[10,11]} \log x} \frac{M_{n+1}}{(n+1)!} = \frac{1}{(n+1)\log 10} \max_{x\in[10,11]} \frac{1}{x^{n+1}}$$
$$< \frac{10^{-(n+1)}}{2.3(n+1)},$$

e per $n = 29$ è $|\epsilon_{an}| < 10^{-31}$. Poiché tale quantità è molto minore della precisione di macchina, si può assumere che gli errori effettivamente generati siano sostanzialmente dovuti all'errore algoritmico.



Per il polinomio di Lagrange l'errore algoritmico è prodotto dal calcolo della somma

$$p(x) = \sum_{j=0}^{n} s_j, \quad s_j = L_j(x) f(x_j),$$

quando i termini non sono tutti dello stesso segno e hanno modulo maggiore di quello del risultato, per cui si verifica un errore di cancellazione. Infatti gli errori relativi ϵ_j, presenti nei termini s_j, contribuiscono all'errore relativo della somma con la quantità

$$\epsilon_p = \frac{1}{p(x)} \sum_{j=0}^{n} s_j \epsilon_j.$$

Posto $\epsilon = \max_{j=0,n} |\epsilon_j|$, si ha

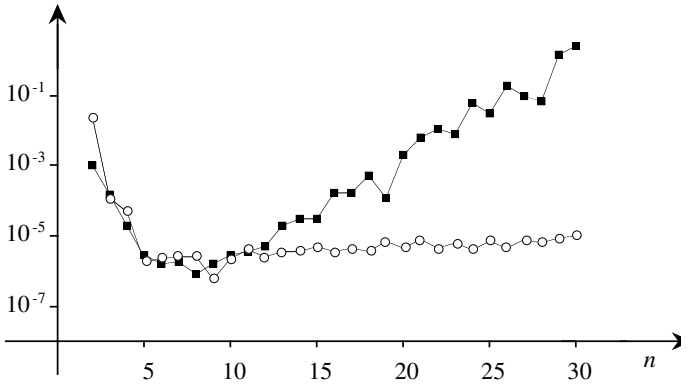
$$|\epsilon_p| < \lambda_n(x)\, \epsilon \max_{x\in[a,b]} \Big| \frac{f(x_j)}{p(x)} \Big|, \quad \text{dove} \quad \lambda_n(x) = \sum_{j=0}^{n} |L_j(x)|.$$

Se $f(x)$ non varia molto nell'intervallo $[a, b]$, il fattore $\lambda_n(x)$ indica quanto influiscono gli errori ϵ_j sul risultato e quindi dà una misura di quanto può essere instabile il calcolo del polinomio di Lagrange nel punto x.

Se i nodi sono equidistanti si può dimostrare che al crescere di n la funzione $\lambda_n(x)$ cresce molto rapidamente quando x non è nella parte centrale dell'intervallo. Nell'esempio della funzione $f(x) = \log x$, se $x = 10.008$ risulta $\lambda_n(x) \approx 10^6$ e infatti l'errore effettivo è circa 0.25, mentre se $x = 10.52$ risulta $\lambda_n(x) \approx 1$ e infatti l'errore effettivo è circa $8.5\ 10^{-6}$.

Nella figura seguente sono riportati i grafici dei moduli degli errori relativi effettivamente generati nel calcolo del polinomio di interpolazione su $n+1$ nodi equidistanti nell'intervallo $[2, 3]$ della funzione $f(x) = \log x$ nel punto $x = 2.008$ (quadratini neri) e nel punto $x = 2.52$ (pallini bianchi), al crescere di n. L'errore inizialmente decresce, mettendo in evidenza la maggiore influenza dell'errore analitico, che decresce

al crescere di n. Poi nel caso del punto 2.008, vicino al primo estremo dell'intervallo, l'errore cresce, con il crescere dell'errore algoritmico, che diventa preponderante rispetto all'errore analitico, mentre nel caso del punto 2.52, nella parte centrale dell'intervallo, l'errore si mantiene più o meno costante. Quindi anche l'analisi dell'er-



rore algoritmico sconsiglia di utilizzare polinomi di interpolazione di grado elevato, specialmente quando x non è nella parte centrale dell'intervallo.

5.5 Il metodo dei minimi quadrati.

Se vogliamo evitare che il polinomio approssimante abbia delle oscillazioni che la funzione data non ha, bisogna ricorrere a polinomi di grado basso anche quando sono noti molti valori della $f(x)$. Occorre cioè trovare con il metodo dei minimi quadrati un polinomio di grado più basso di quello di interpolazione, che in generale non assume gli stessi valori della $f(x)$ nei nodi. Ma questo è ragionevole se i valori $f(x_i)$ sono ottenuti sperimentalmente e quindi sono affetti da errore.

Sia allora $m < n$. Con il metodo dei minimi quadrati si determinano i coefficienti del polinomio

$$p_m(x) = a_0 x^m + a_1 x^{m-1} + \ldots + a_{m-1} x + a_m$$

in modo che risulti minima la norma 2 del vettore degli scarti, cioè

$$\min_{(a_0,a_1,\ldots,a_m)} \|\boldsymbol{r}\|_2, \quad \text{dove} \quad r_i = f(x_i) - p_m(x_i), \quad i = 0, \ldots, n.$$

I punti di minimo di $\|\boldsymbol{r}\|_2$ e di $\|\boldsymbol{r}\|_2^2$ coincidono, quindi si preferisce formulare il problema dei minimi quadrati nel modo seguente

$$\min_{\boldsymbol{a}} \varphi(\boldsymbol{a}), \quad \text{dove } \varphi(\boldsymbol{a}) = \sum_{i=0}^{n} \Big[f(x_i) - p_m(x_i) \Big]^2 = \sum_{i=0}^{n} \Big[f(x_i) - \sum_{j=0}^{m} a_j x_i^{m-j} \Big]^2,$$

avendo posto $\boldsymbol{a} = [a_0, a_1, \ldots, a_m]^T$. Si può dimostrare che se i nodi sono tutti distinti la funzione $\varphi(\boldsymbol{a})$ è convessa. Quindi vi è un unico punto di minimo che si ottiene

imponendo la condizione di stazionarietà

$$\frac{\partial \varphi}{\partial a_k} = -2\sum_{i=0}^{n}\Big[f(x_i) - \sum_{j=0}^{m} a_j x_i^{m-j}\Big]x_i^{m-k} = 0, \quad \text{per} \quad k = 0, \ldots, m.$$

Si risolve quindi il sistema lineare

$$\sum_{j=0}^{m}\Big[\sum_{i=0}^{n} x_i^{2m-j-k}\Big]a_j = \sum_{i=0}^{n} f(x_i)x_i^{m-k}, \quad \text{per} \quad k = 0, \ldots, m. \tag{5.10}$$

Il sistema (5.10), di ordine $m+1$, è detto sistema delle *equazioni normali* e ha una sola soluzione. Il polinomio p_m così ottenuto è chiamato polinomio di *regressione*.

Per $m = 1$ il polinomio di *regressione lineare* che si ottiene è

$$p_1(x) = a_0 x + a_1,$$

dove i coefficienti a_0 e a_1 si ottengono risolvendo il sistema (5.10), che in questo caso particolare ha la forma

$$\begin{cases} \Big[\sum_{i=0}^{n} x_i^2\Big]a_0 + \Big[\sum_{i=0}^{n} x_i\Big]a_1 = \sum_{i=0}^{n} f(x_i)x_i \\ \Big[\sum_{i=0}^{n} x_i\Big]a_0 + (n+1)a_1 = \sum_{i=0}^{n} f(x_i). \end{cases}$$

Esempio. È data la funzione

x	0.5	1.1	1.2	1.4	2.1
$f(x)$	0.9	1.13	1.52	1.67	1.76

Il polinomio di interpolazione è

$$p_4(x) = 30.48 - 119.9x + 161.4x^2 - 88.28x^3 + 16.91x^4,$$

mentre il polinomio di regressione lineare è

$$p_1(x) = 0.5677x + 0.6807.$$

I grafici sono riportati nella figura 5.6. Il grafico del polinomio di interpolazione è con linea tratteggiata, il grafico della retta di regressione è con linea continua. Poiché la funzione discreta è crescente, sembra ragionevole che anche la funzione approssimante debba mantenere questa caratteristica. Questo è vero per la retta di regressione, che passa approssimativamente in mezzo ai valori assegnati della $f(x)$, ma non per il polinomio di interpolazione. □

Esempio. In figura 5.7 è riportato il grafico del polinomio di regressione di grado 5 della funzione discreta dell'esempio del fiume. Si confronti questo grafico con quello della figura 5.5. Il polinomio di regressione non passa esattamente per i punti della funzione discreta, ma l'approssimazione complessiva è sicuramente preferibile.

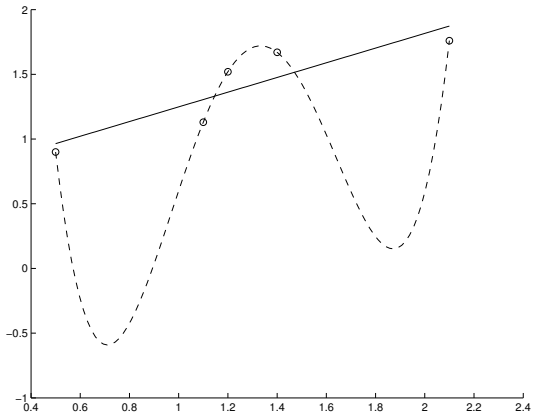


Figura 5.6: Polinomio di interpolazione e polinomio di regressione lineare.

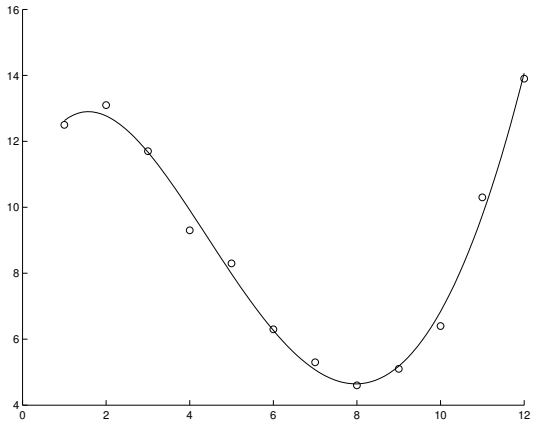


Figura 5.7: Polinomio di regressione della funzione dell'esempio del fiume.

5.6 Formule di quadratura.

Sia $f(x)$ una funzione reale definita su un intervallo $[a, b]$. Il problema che si vuole studiare è quello della approssimazione dell'integrale

$$S = \int_a^b f(x)\ dx \tag{5.11}$$

Nel caso in cui $f(x)$ sia una funzione continua, il teorema fondamentale del calcolo integrale assicura l'esistenza su $[a, b]$ di una funzione $F(x)$, detta *primitiva* della $f(x)$ tale che

$$S = F(b) - F(a).$$

Non sempre però la $F(x)$ è esprimibile in termini di funzioni elementari e, anche quando questo è possibile, il calcolo di $F(a)$ e $F(b)$ può essere costoso. Per questo è importante disporre di tecniche numeriche per il calcolo approssimato della (5.11).

Il metodo più semplice per approssimare un integrale è basato proprio sulla definizione di integrale di Riemann. Con una griglia di punti x_i, $i = 0, \ldots, N$ si suddivide l'intervallo di integrazione $[a, b]$ in N sottointervalli $[x_i, x_{i+1}]$. Per la proprietà di additività dell'integrale si ha

$$S = \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x)\, dx. \tag{5.12}$$

Si approssima poi la $f(x)$ su $[x_i, x_{i+1}]$ con il polinomio lineare di interpolazione nei nodi x_i e x_{i+1}, cioè si approssima l'integrale su $[x_i, x_{i+1}]$ con l'area del trapezio avente come basi i valori $f(x_i)$ e $f(x_{i+1})$ e come altezza $x_{i+1} - x_i$ (si veda la figura 5.8). In pratica il valore dell'integrale è approssimato dalla somma

$$T_N = \frac{1}{2} \sum_{i=0}^{N-1} (x_{i+1} - x_i)\, [f(x_i) + f(x_{i+1})].$$

La formula risulta più semplice se i punti x_i sono scelti equidistanti di passo h, cioè

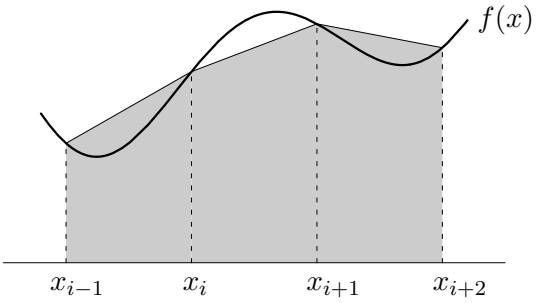


Figura 5.8: Formula dei trapezi.

$x_i = a + ih$, per $i = 0, \ldots, N$, dove $h = (b - a)/N$. In questo caso si ha

$$T_N = \frac{h}{2} \sum_{i=0}^{N-1} [f(x_i) + f(x_{i+1})] = \frac{b-a}{2N} \left[f(a) + 2 \sum_{i=1}^{N-1} f(x_i) + f(b) \right]. \tag{5.13}$$

La formula (5.13) è la classica *formula di quadratura dei trapezi*. Come appare anche dalla figura 5.8, quando si approssima S con T_N si commette un errore dato dall'area compresa fra la $f(x)$ e la spezzata di vertici $f(x_i)$. Questo errore è detto *resto* della formula dei trapezi e si ha

$$R_N^{(T)} = S - T_N = \sum_{i=0}^{N-1} r_i,$$

dove r_i è il resto su ciascun sottointervallo.

Esempio. Per valutare l'accuratezza della formula dei trapezi, approssimiamo l'integrale

$$S = \int_0^\pi \mathrm{e}^x \sin x \; dx.$$

usando la (5.13) con $N = 10$. Il valore che si ottiene è 11.8725. Il valore esatto dell'integrale è $(\mathrm{e}^\pi + 1)/2 = 12.0703$, come è possibile verificare notando che $\mathrm{e}^x(\sin x - \cos x)/2$ è una primitiva di $f(x)$. L'approssimazione ottenuta non è molto buona. Si riapplica la formula con $N = 20$. Adesso si ottiene 12.0207. Quindi il resto si è ridotto.

Per vedere se effettivamente quando si aumenta N si hanno approssimazioni sempre migliori dell'integrale, cioè se per $N \to \infty$ il resto R_N converge a zero, consideriamo il resto sul singolo intervallo $[x_i, x_{i+1}]$

$$r_i = \int_{x_i}^{x_{i+1}} f(x) \; dx - h \, \frac{f(x_i) + f(x_{i+1})}{2} = \int_{x_i}^{x_{i+1}} (f(x) - p_i(x)) \; dx,$$

dove $p_i(x)$ è il polinomio lineare di interpolazione della $f(x)$ nei nodi x_i e x_{i+1}. Per la (5.8) se $f \in C^2[a, b]$ si ha

$$r_i = \int_{x_i}^{x_{i+1}} r(x) \; dx = \frac{1}{2} \int_{x_i}^{x_{i+1}} (x - x_i)(x - x_{i+1}) f''(\xi(x)) \; dx.$$

Poiché la funzione $(x - x_i)(x - x_{i+1})$ ha segno costante in (x_i, x_{i+1}) e la $f''(x)$ è continua, si può applicare il teorema della media integrale e si ha

$$r_i = \frac{1}{2} \, f''(\zeta_i) \int_{x_i}^{x_{i+1}} (x - x_i)(x - x_{i+1}) \; dx = -\frac{h^3}{12} \, f''(\zeta_i),$$

dove ζ_i è un opportuno punto in (x_i, x_{i+1}).

Applichiamo questa relazione a tutto l'intervallo $[a, b]$. Si ha

$$R_N = \sum_{i=0}^{N-1} r_i = -\frac{h^3}{12} \sum_{i=0}^{N-1} f''(\zeta_i) = -\frac{h^3}{12} \, N f_2,$$

dove f_2 è la media degli N valori $f''(\zeta_i)$. Poiché f_2 è compreso fra il più piccolo e il più grande dei valori della funzione $f''(x)$, che è continua per ipotesi, esiste un punto $\xi \in (a, b)$ tale che $f''(\xi) = f_2$. Si ottiene in tal modo l'espressione del resto della formula dei trapezi

$$R_N^{(T)} = -\frac{1}{12} \left(\frac{b - a}{N} \right)^3 N f''(\xi) = -\frac{(b - a)^3}{12 N^2} f''(\xi). \tag{5.14}$$

Esempio. Approssimiamo l'integrale dell'esempio precedente con la formula dei trapezi, facendo crescere N in modo regolare, raddoppiandolo ogni volta.

N	resto	trapezi
2	4.5141	7.55630
4	1.2147	10.8557
8	$3.0863\ 10^{-1}$	11.76172
16	$7.7458\ 10^{-2}$	11.99289
32	$1.9383\ 10^{-2}$	12.05096
64	$4.8470\ 10^{-3}$	12.06550
128	$1.2118\ 10^{-3}$	12.06913
256	$3.0296\ 10^{-4}$	12.07004
512	$7.5740\ 10^{-5}$	12.07027
1024	$1.8935\ 10^{-5}$	12.07033

Il valore ottenuto con $N = 1024$ è una buona approssimazione dell'integrale esatto, ma il calcolo ha richiesto 1025 valori della funzione $f(x)$. Se ogni valutazione di $f(x)$ ha un elevato costo computazionale, è opportuno usare un metodo che converga più rapidamente al valore esatto dell'integrale quando $N \to \infty$.

La formula di *Cavalieri-Simpson*, che studiamo ora, in generale dà un'approssimazione dell'integrale buona come quella che si può ottenere con i trapezi, ma con un numero più basso di valutazioni della $f(x)$. La tecnica con cui si ricava la formula di Cavalieri-Simpson è simile a quella usata per la formula dei trapezi, solo che adesso nella (5.12) l'integrale su ciascun sottointervallo $[x_i, x_{i+1}]$ viene approssimato con l'area della figura che si ottiene sostituendo la $f(x)$ con una parabola. Più precisamente si considera il polinomio $p_i(x)$ di 2^o grado che interpola la $f(x)$ nei tre nodi x_i, $(x_i + x_{i+1})/2$ e x_{i+1} (il secondo nodo è il punto di mezzo degli altri due). L'integrale di $f(x)$ su $[x_i, x_{i+1}]$ viene così approssimato con l'integrale di $p_i(x)$, che a conti fatti dà

$$\int_{x_i}^{x_{i+1}} p_i(x)\, dx = \frac{x_{i+1} - x_i}{6} \left[f(x_i) + 4f\Big(\frac{x_i + x_{i+1}}{2}\Big) + f(x_{i+1}\right].$$

L'integrale su $[a, b]$ viene allora approssimato dalla somma

$$C_N = \frac{1}{6} \sum_{i=0}^{N-1} (x_{i+1} - x_i) \left[f(x_i) + 4f\Big(\frac{x_i + x_{i+1}}{2}\Big) + f(x_{i+1})\right].$$

Se scegliamo i punti x_i equidistanti in $[a, b]$ si ottiene la formula di Cavalieri-Simpson

$$C_N = \frac{b-a}{6N} \left[f(a) + 2 \sum_{k=1}^{N-1} f(x_i) + 4 \sum_{i=0}^{N-1} f\Big(\frac{x_i + x_{i+1}}{2}\Big) + f(b)\right]. \tag{5.15}$$

Si può dimostrare che se $f \in C^4[a,b]$ il resto della formula di Cavalieri-Simpson è

$$R_N^{(C)} = -\frac{(b-a)^5}{2880\ N^4}\ f^{(4)}(\xi), \tag{5.16}$$

dove ξ è un opportuno punto di $[a,b]$.

Esempio. Applichiamo la formula di Cavalieri-Simpson con N crescente alla funzione $f(x)$ dell'esempio visto sopra.

N	resto	Cavalieri-Simpson
2	$1.1490\ 10^{-1}$	11.95545
4	$6.6053\ 10^{-3}$	12.06374
8	$4.0231\ 10^{-4}$	12.06994
16	$2.4975\ 10^{-5}$	12.07032
32	$1.5582\ 10^{-6}$	12.05034

Dalla tabella risulta evidente che la formula di Cavalieri-Simpson converge molto più rapidamente di quella dei trapezi. Un'approssimazione migliore di quella ottenuta con i trapezi con 1025 valutazioni della $f(x)$ viene qui ottenuta con $N = 32$, cioè con 65 valutazioni della $f(x)$.

Appendice A

Numeri complessi

L'estensione del campo $\boldsymbol{R}$ dei numeri reali al campo $\boldsymbol{C}$ dei numeri complessi si rese necessaria nel 16^o secolo, quando furono scoperte le formule risolutive delle equazioni di 3^o e 4^o grado. Naturalmente fin dall'antichità era noto che certe equazioni di 2^o grado non hanno soluzioni reali, ma questo era in accordo con l'interpretazione geometrica che se ne dava. Se infatti si considera il calcolo delle soluzioni dell'equazione

$$ax^2 + bx + c = 0$$

come quello della ricerca dei punti di intersezione della parabola

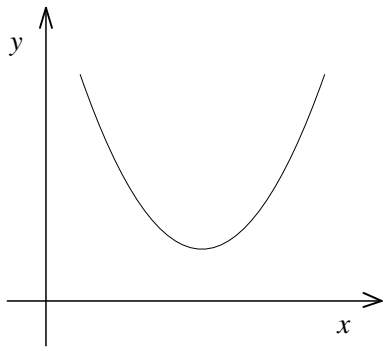
$$y = ax^2 + bx + c$$

con l'asse delle x

$$y = 0,$$

è ragionevole pensare che l'equazione non abbia soluzioni quando la parabola non interseca l'asse delle x, cioè quando

$$\Delta = b^2 - 4ac < 0.$$



In questo caso perciò non si avverte alcuna necessità di ampliare il campo in cui si opera per ottenere le soluzioni di una equazione di 2^o grado anche quando il discriminante Δ è negativo. Sembra anzi che una tale estensione rappresenti solo un'elegante costruzione dei matematici, di scarsa utilità pratica.

La questione però cambiò completamete aspetto con la formula risolutiva dell'equazione di 3^o grado: in questo caso infatti è necessario passare attraverso il calcolo di radici quadrate di numeri negativi anche per trovare le soluzioni reali dell'equazione. Ci vollero più di 200 anni perché i matematici completassero l'estensione del campo $\boldsymbol{R}$ al campo $\boldsymbol{C}$, ne studiassero tutte le conseguenze e finalmente ne dessero la rappresentazione grafica, che semplifica di molto la comprensione.

Al giorno d'oggi i numeri complessi sono indispensabili non solo in matematica, ma anche nella applicazioni pratiche, come ad esempio per rappresentare le correnti alternate.

A.1 Definizione di numero complesso

Per definire i numeri complessi dobbiamo innanzi tutto introdurre il simbolo $\boldsymbol{i}$, detto *unità immaginaria* e definito dalla relazione

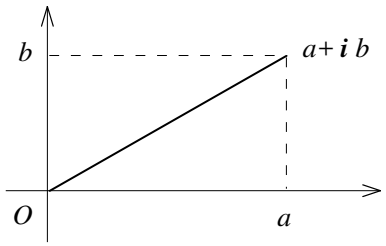
$$\boldsymbol{i}^2 = -1.$$

È chiaro che $\boldsymbol{i}$ non rappresenta alcun numero reale. Si definisce ora *numero complesso* un numero della forma

$$a + \boldsymbol{i}\, b, \quad \text{dove} \quad a,\, b \in \boldsymbol{R}.$$

a e b vengono detti rispettivamente *parte reale* e *coefficiente dell'immaginario* del numero complesso.

L'interpretazione grafica che Gauss diede dei numeri complessi ci renderà più immediate tutte le definizioni che daremo successivamente. Un numero complesso viene rappresentato in un piano di coordinate ortogonali con un punto, la cui ascissa è uguale alla parte reale e la cui ordinata è uguale al coefficiente dell'immaginario.



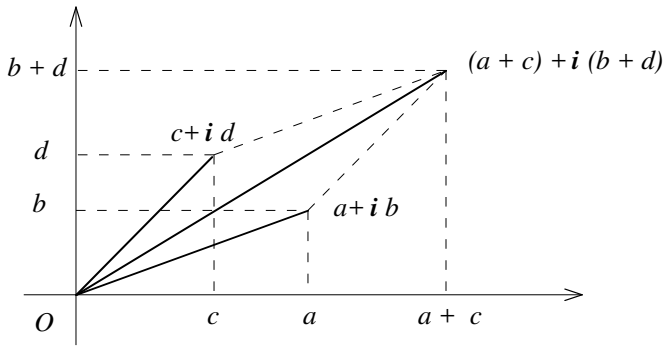
Se identifichiamo un numero complesso con coefficiente dell'immaginario nullo $a + \boldsymbol{i}\ 0$ con il numero reale a, si può dire che l'insieme $\boldsymbol{R}$ dei numeri reali è un sottoinsieme dell'insieme dei numeri complessi $\boldsymbol{C}$.

A.2 Operazioni fra numeri complessi

Si definiscono su $\boldsymbol{C}$ delle relazioni di uguaglianza e delle operazioni aritmetiche che verificano le stesse proprietà che valgono su $\boldsymbol{R}$. Queste relazioni e definizioni risultano naturali se si applica il calcolo simbolico alle quantità a, b, $\boldsymbol{i}$, sostituendo, dove compare, $\boldsymbol{i}^2$ con -1

$$\begin{aligned}
&a + \boldsymbol{i}\, b = c + \boldsymbol{i}\, d \quad \text{se e solo se} \quad a = c \quad \text{e} \quad b = d,\\
&(a + \boldsymbol{i}\, b) + (c + \boldsymbol{i}\, d) = (a + c) + \boldsymbol{i}\, (b + d),\\
&(a + \boldsymbol{i}\, b) \cdot (c + \boldsymbol{i}\, d) = (ac - bd) + \boldsymbol{i}\, (bc + ad).
\end{aligned}$$

Graficamente l'addizione di due numeri complessi corrisponde alla cosiddetta *regola del parallelogramma.*



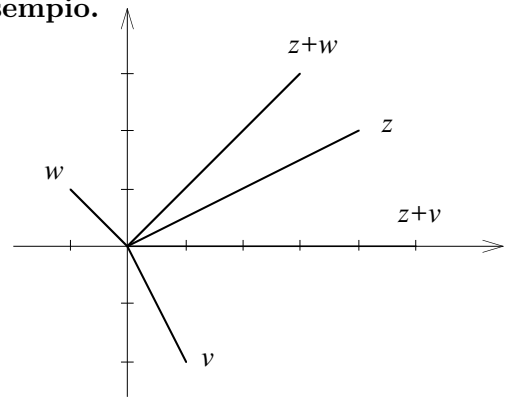
Anche le operazioni inverse di sottrazione e divisione non presentano difficoltà. Si ha infatti

$$(a + \boldsymbol{i}\, b) - (c + \boldsymbol{i}\, d) = (a - c) + \boldsymbol{i}\, (b - d),$$

e se $c^2 + d^2 \neq 0$

$$\frac{a + \boldsymbol{i}\, b}{c + \boldsymbol{i}\, d} = \frac{(a + \boldsymbol{i}\, b) \cdot (c - \boldsymbol{i}\, d)}{(c + \boldsymbol{i}\, d) \cdot (c - \boldsymbol{i}\, d)} = \frac{(ac + bd) + \boldsymbol{i}\, (bc - ad)}{c^2 + d^2} = \Big(\frac{ac + bd}{c^2 + d^2}\Big) + \boldsymbol{i}\, \Big(\frac{bc - ad}{c^2 + d^2}\Big).$$

Esempio.



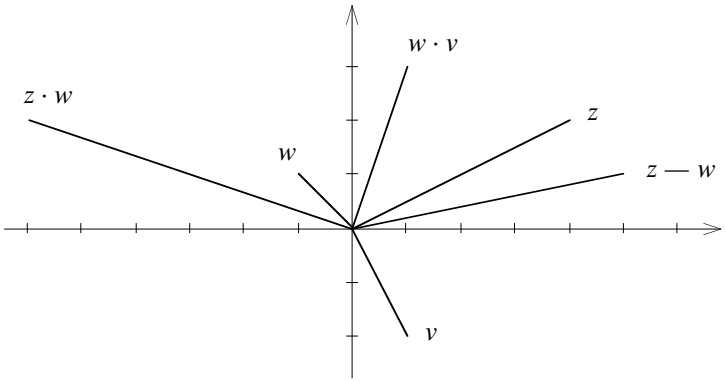
Consideriamo i numeri complessi

$$z = 4 + \boldsymbol{i}\, 2,$$
$$w = -1 + \boldsymbol{i},$$
$$v = 1 - \boldsymbol{i}\, 2.$$

Risulta

$$z + w = 3 + \boldsymbol{i}\, 3,$$
$$z + v = 5 + \boldsymbol{i}\, 0 = 5,$$

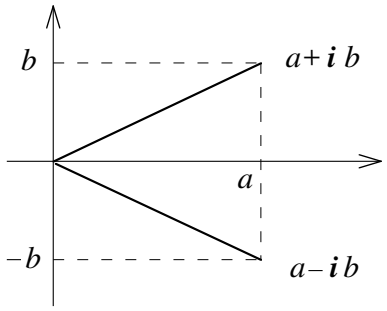
$$z - w = 5 + \boldsymbol{i},$$
$$z \cdot w = (4 + \boldsymbol{i}\, 2) \cdot (-1 + \boldsymbol{i}) = -6 + \boldsymbol{i}\, 2,$$
$$w \cdot v = (-1 + \boldsymbol{i}) \cdot (1 - \boldsymbol{i}\, 2) = 1 + \boldsymbol{i}\, 3.$$



Se $z = a + \boldsymbol{i}\, b$ è un numero complesso, il numero

$$\overline{z} = a - \boldsymbol{i}\, b$$

viene detto il *coniugato* di z. Graficamente $\overline{z}$ è rappresentato da un punto ribaltato attorno all'asse reale rispetto al punto che rappresenta z.



La somma e il prodotto di due numeri complessi coniugati sono due numeri reali. Risulta infatti

$$\begin{aligned} z + \overline{z} &= (a + \boldsymbol{i}\, b) + (a - \boldsymbol{i}\, b) = 2a, \\ z \cdot \overline{z} &= (a + \boldsymbol{i}\, b) \cdot (a - \boldsymbol{i}\, b) = a^2 + b^2. \end{aligned} \tag{A.1}$$

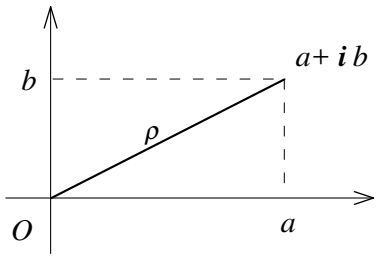
Per esempio, se $z = 2 + \boldsymbol{i}\, 3$, è

$$\overline{z} = 2 - \boldsymbol{i}\, 3, \quad z + \overline{z} = 4, \quad z \cdot \overline{z} = 13.$$

Il numero reale

$$\rho = \sqrt{a^2 + b^2}$$

viene detto *modulo* di z e graficamente corrisponde alla distanza del punto che rappresenta z dall'origine O del piano complesso.



Se $b = 0$ e il numero complesso $z = a + \boldsymbol{i}\, b$ si identifica con il numero reale a, il modulo di z coincide con il valore assoluto di a; per questo motivo si indica di solito con $|z|$ il modulo di z anche quando questo non è reale. Dalla (A.1) risulta che

$$|z|^2 = z \cdot \overline{z}.$$

A.3 Equazione di 2^o grado

Verifichiamo adesso che ogni equazione di 2^o grado a coefficienti reali

$$ax^2 + bx + c = 0, \quad a \neq 0, \tag{A.2}$$

ha soluzione in $\boldsymbol{C}$. Dividendo per a e portando l'ultimo termine al secondo membro si ha

$$x^2 + \frac{b}{a}\, x = -\frac{c}{a}\,,$$

da cui

$$x^2 + \frac{b}{a}\, x + \frac{b^2}{4a^2} = \frac{b^2}{4a^2} - \frac{c}{a}\,,$$

e quindi

$$\left(x + \frac{b}{2a}\right)^2 = \frac{b^2 - 4ac}{4a^2}\,.$$

Ponendo $y = x + \dfrac{b}{2a}$, si ottiene l'equazione

$$y^2 = \frac{\Delta}{4a^2}, \quad \Delta = b^2 - 4ac,$$

che, se $\Delta \geq 0$, ha, come sappiamo, le due soluzioni reali

$$y_1 = \frac{\sqrt{\Delta}}{2a}, \quad y_2 = -\frac{\sqrt{\Delta}}{2a}.$$

Se $\Delta < 0$ l'equazione non ha soluzioni reali, ma può essere scritta nella forma

$$y^2 = (-1)\,\frac{-\Delta}{4a^2}, \tag{A.3}$$

e i due numeri complessi

$$y_1 = \boldsymbol{i}\,\frac{\sqrt{-\Delta}}{2a}, \quad y_2 = -\boldsymbol{i}\,\frac{\sqrt{-\Delta}}{2a},$$

sono tali che i loro quadrati soddisfano la (A.3). Le soluzioni di (A.2) risultano quindi

$$x_{1,2} = \begin{cases} -\dfrac{b}{2a} \pm \dfrac{\sqrt{\Delta}}{2a}, & \text{se } \Delta \geq 0, \\ -\dfrac{b}{2a} \pm \boldsymbol{i}\,\dfrac{\sqrt{-\Delta}}{2a}, & \text{se } \Delta < 0. \end{cases}$$

Per esempio, l'equazione

$$2x^2 - 2x + 5 = 0$$

ha le due soluzioni complesse $x_1 = \dfrac{1}{2} + \boldsymbol{i}\,\dfrac{3}{2}$ e $x_2 = \dfrac{1}{2} - \boldsymbol{i}\,\dfrac{3}{2}$, fra loro coniugate.

A.4 Rappresentazione trigonometrica

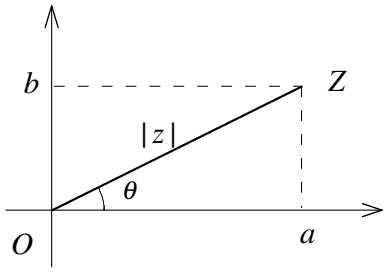
Dei numeri complessi si può dare anche un'altra rappresentazione, detta *trigonometrica*. Dalla figura, in cui il numero complesso $z = a + \boldsymbol{i}\ b$ è rappresentato con il punto Z e l'angolo compreso fra il semiasse reale positivo e il segmento OZ è chiamato θ, risulta che

$$a = |z|\,\cos\theta, \quad b = |z|\,\sin\theta,$$

e quindi

$$z = |z|\,(\cos\theta + \boldsymbol{i}\ \sin\theta).$$

L'angolo θ viene detto *argomento* di z.



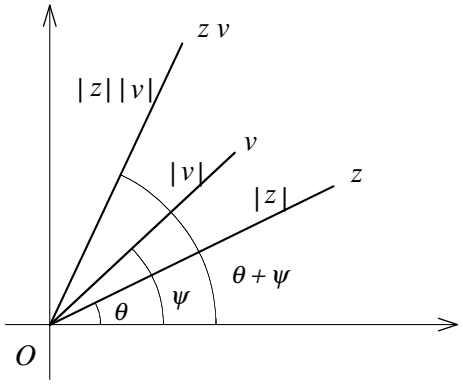
La rappresentazione trigonometrica si usa specialmente quando si vogliono eseguire moltiplicazioni o elevamenti a potenza di numeri complessi. Posto infatti

$$z = |z|\,(\cos\theta + \boldsymbol{i}\ \sin\theta) \quad \text{e} \quad v = |v|\,(\cos\psi + \boldsymbol{i}\ \sin\psi),$$

risulta

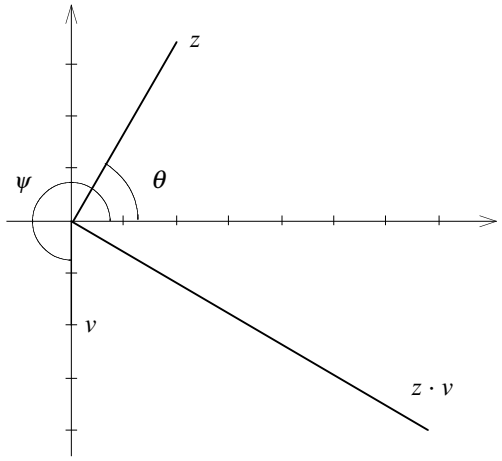
$$\begin{aligned} z \cdot v &= |z|\ |v| \Big[\cos\theta\cos\psi - \sin\theta\sin\psi + \boldsymbol{i}\ (\cos\theta\sin\psi + \sin\theta\cos\psi) \Big] \\ &= |z|\ |v| \Big[\cos(\theta+\psi) + \boldsymbol{i}\ \sin(\theta+\psi) \Big]. \end{aligned}$$

Perciò il prodotto di due numeri complessi ha il modulo uguale al prodotto dei moduli dei fattori e argomento uguale alla somma degli argomenti dei fattori.



Esempio. Se $z = 2 + \boldsymbol{i}\ 2\sqrt{3}$ e $v = -2\boldsymbol{i}$, si ha $|z| = 4$, $\theta = \arg(z) = \dfrac{\pi}{3}$, $|v| = 2$, $\psi = \arg(v) = \dfrac{3}{2}\,\pi$ e risulta

$$\begin{aligned} z \cdot v &= 8\Big(\cos\ \frac{11}{6}\,\pi + \boldsymbol{i}\ \sin\ \frac{11}{6}\,\pi \Big) \\ &= 8\Big(\cos\ \frac{\pi}{6} - \boldsymbol{i}\ \sin\ \frac{\pi}{6} \Big). \end{aligned}$$



A.5 Formula di de Moivre

Applicando ripetutamente la formula del prodotto di numeri complessi, si ottiene la seguente formula di *de Moivre* per la potenza n-esima, con n intero positivo

$$z^n = |z|^n\,(\cos n\theta + \boldsymbol{i}\ \sin n\theta), \tag{A.4}$$

cioè la potenza n-esima di z ha modulo uguale alla potenza n-esima del modulo di z e argomento multiplo dell'argomento di z. Per esempio, se

$$z = 2 + \boldsymbol{i}\, 2\sqrt{3} = 4\Big(\cos\frac{\pi}{3} + \boldsymbol{i}\, \sin\frac{\pi}{3}\Big),$$

si ha

$$\begin{aligned} z^2 &= 4^2\Big(\cos\frac{2\pi}{3} + \boldsymbol{i}\, \sin\frac{2\pi}{3}\Big) = 8(-1 + \boldsymbol{i}\,\sqrt{3}),\\ z^3 &= 4^3(\cos\pi + \boldsymbol{i}\, \sin\pi) = -64,\\ z^4 &= 4^4\Big(\cos\frac{4\pi}{3} + \boldsymbol{i}\, \sin\frac{4\pi}{3}\Big) = -128(1 + \boldsymbol{i}\,\sqrt{3}), \end{aligned}$$

e così via.

È facile verificare che la (A.4) vale anche quando n è un intero negativo o nullo. In particolare per $n = 0$ si ha

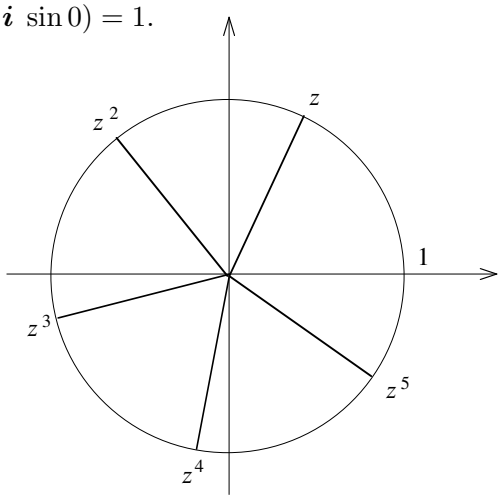
$$z^0 = |z|^0\,(\cos 0 + \boldsymbol{i}\, \sin 0) = 1.$$

Se z è un numero complesso di modulo uguale ad 1, cioè è della forma

$$z = \cos\theta + \boldsymbol{i}\, \sin\theta,$$

le sue potenze n-esime hanno ancora modulo 1 e sono della forma

$$z^n = \cos n\theta + \boldsymbol{i}\, \sin n\theta.$$



A.6 La radice n-esima

L'operazione inversa della potenza n-esima, cioè la radice n-esima, merita di essere studiata con più cura. Dato il numero

$$w = |w|\,(\cos\psi + \boldsymbol{i}\, \sin\psi),$$

si cerca un numero complesso

$$z = |z|\,(\cos\theta + \boldsymbol{i}\, \sin\theta),$$

tale che $z^n = w$. Per la (A.4) deve essere

$$z^n = |z|^n\,(\cos n\theta + \boldsymbol{i}\, \sin n\theta) = |w|(\cos\psi + \boldsymbol{i}\, \sin\psi), \tag{A.5}$$

cioè

$$|z|^n = |w|, \quad \cos n\theta = \cos\psi \quad \text{e} \quad \sin n\theta = \sin\psi,$$

da cui

$$|z| = |w|^{1/n} \quad \text{e} \quad n\theta = \psi + 2k\pi, \quad \text{per} \quad k = 0, 1, 2, \ldots.$$

Allora, posto

$$\theta_k = \frac{\psi}{n} + \frac{2k\pi}{n}, \quad \text{per} \quad k = 0, 1, 2, \ldots \tag{A.6}$$

i numeri

$$z_k = |w|^{1/n} \, (\cos\theta_k + \boldsymbol{i} \, \sin\theta_k), \quad \text{per} \quad k = 0, 1, 2, \ldots$$

sono radici n-esime di w. Però dei θ_k scritti nella (A.6), solo i primi n valori producono soluzioni diverse dell'equazione (A.5). Si ha infatti

$$\theta_n = \frac{\psi}{n} + \frac{2n\pi}{n} = \theta_0 + 2\pi,$$

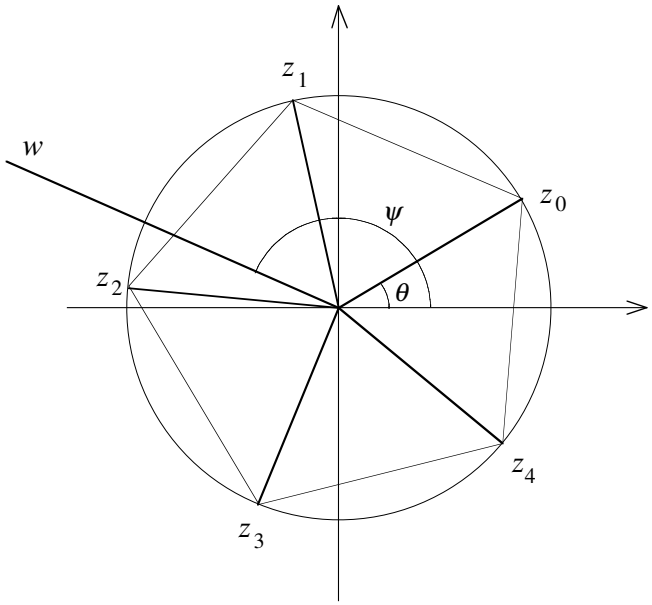
per cui

$$\begin{aligned} z_n &= |w|^{1/n} \, (\cos\theta_n + \boldsymbol{i} \, \sin\theta_n) = |w|^{1/n} \, (\cos(\theta_0 + 2\pi) + \boldsymbol{i} \, \sin(\theta_0 + 2\pi)) \\ &= |w|^{1/n} \, (\cos\theta_0 + \boldsymbol{i} \, \sin\theta_0) = z_0. \end{aligned}$$

Quindi l'equazione $z^n = w$ ha le n soluzioni

$$z_k = |w|^{1/n} \, (\cos\theta_k + \boldsymbol{i} \, \sin\theta_k), \quad \text{per} \quad k = 0, 1, \ldots, n-1.$$

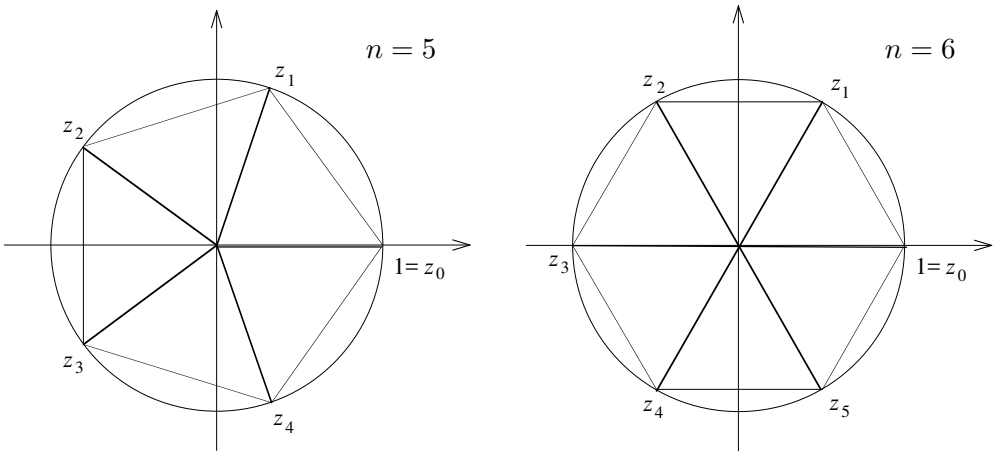
Graficamente risulta, per esempio per $n = 5$



Cioè le radici n-esime di z, di modulo $|w|^{1/n}$, stanno ai vertici di un poligono regolare di n lati inscritto nella circonferenza di centro l'origine del piano complesso e raggio $|w|^{1/n}$, e avente il primo vertice in corrispondenza di z_0. Di particolare interesse sono le radici n-esime dell'unità, cioè del numero complesso $w = 1$. Nella notazione trigonometrica w ha modulo 1 e argomento $\psi = 0$, per cui le sue radici n-esime risultano

$$z_k = \cos\theta_k + \boldsymbol{i}\ \sin\theta_k, \quad \theta_k = \frac{2k\pi}{n}, \quad k = 0, 1, \ldots, n-1.$$

Graficamente si ha



Per n dispari solo la soluzione $z_0 = 1$ è reale, le altre $n-1$ soluzioni non sono reali. Per n pari invece vi sono due soluzioni reali, $z_0 = 1$ e $z_{n/2} = -1$ e le altre $n-2$ non sono reali. Le soluzioni non reali sono coniugate a coppie: $\overline{z}_1 = z_{n-1}$, $\overline{z}_2 = z_{n-2}$, e così via.

A.7 Esercizi

1. Rappresentare nel piano complesso i punti

$$z_1 = 2 + \boldsymbol{i}\ 2, \quad z_2 = 1 - \boldsymbol{i}, \quad z_3 = -2, \quad z_4 = \boldsymbol{i},$$
$$\overline{z}_1, \quad \overline{z}_2, \quad \overline{z}_3, \quad \overline{z}_4,$$
$$z_5 = z_1 + z_2, \quad z_6 = z_3 + z_4, \quad z_7 = z_1 + \overline{z}_4,$$
$$z_8 = z_1 \cdot z_2, \quad z_9 = z_3 \cdot z_4, \quad z_{10} = z_3 \cdot \overline{z}_4, \quad z_{11} = (z_1 \cdot \overline{z}_3)^2.$$

2. Verificare che il quoziente di due numeri complessi ha modulo uguale al quoziente dei moduli e argomento uguale alla differenza degli argomenti.

3. Scrivere in forma trigonometrica i numeri complessi

$$z_1 = \frac{\sqrt{2}}{3} + \boldsymbol{i}\ \frac{\sqrt{2}}{\sqrt{3}}, \quad z_2 = \frac{\sqrt{3}}{2} - \boldsymbol{i}\ \frac{1}{2},$$

e calcolare

$$z_1 \cdot z_2, \quad \frac{z_1}{z_2}, \quad z_1^2, \quad z_2^{-3}.$$

4. Verificare che la definizione di uguaglianza fra numeri complessi verifica le proprietà simmetrica, riflessiva e transitiva.

5. Verificare che le operazioni di addizione e moltiplicazione di numeri complessi, definite in (A.2) verificano le proprietà commutativa, associativa e distributiva, e che vale la legge di annullamento del prodotto.

6. Verificare che il modulo della somma di due numeri complessi è minore o uguale alla somma dei moduli dei due numeri.

7. Verificare che il modulo del prodotto di due numeri complessi è uguale al prodotto dei moduli dei due numeri.

8. Verificare che se $z = \cos\theta + \boldsymbol{i}\ \sin\theta$, allora $\dfrac{1}{z} = \overline{z}$.

9. Dedurre dalla formula di de Moivre le relazioni trigonometriche che legano $\sin k\theta$ e $\cos k\theta$ con $\sin\theta$ e $\cos\theta$ per $k = 2, 3, 4, \ldots$

(Traccia: posto $z = a + \boldsymbol{i}\, b$, con $a = \cos\theta$, $b = \sin\theta$, è

$$z^2 = (a + \boldsymbol{i}\, b)^2 = (a^2 - b^2) + \boldsymbol{i}\, 2ab = \cos^2\theta - \sin^2\theta + \boldsymbol{i}\, 2\sin\theta\cos\theta;$$

d'altra parte per la formula di de Moivre è

$$z^2 = (\cos\theta + \boldsymbol{i}\ \sin\theta)^2 = \cos 2\theta + \boldsymbol{i}\ \sin 2\theta.\ )$$

10. Calcolare le radici quadrate del numero

$$z = \frac{1}{2} + \boldsymbol{i}\,\frac{\sqrt{3}}{2}.$$

11. Calcolare le radici cubiche di $\boldsymbol{i}$.

12. Calcolare le radici seste di 1.

13. Calcolare le radici seste di -1.

14. Calcolare le radici quarte di $1 - \boldsymbol{i}$.

15. Risolvere l'equazione $z^2 + (4 + \boldsymbol{i})z + 5 - \boldsymbol{i}\ = 0$.

16. Verificare che $\overline{z}^{\,k} = \overline{(z^k)}$. Sfruttando questa proprietà dimostrare che se z è soluzione dell'equazione algebrica di grado n

$$a_n z^n + a_{n-1} z^{n-1} + \ldots + a_0 = 0,$$

con i coefficienti $a_0, a_1, \ldots, a_n$ reali, allora $\overline{z}$ è anch'esso soluzione dell'equazione. Verificate che questa proprietà può non valere se i coefficienti non sono numeri reali, come nel caso dell'equazione dell'esercizio 15.

Appendice B

Introduzione a Matlab

Matlab è un sistema interattivo progettato per l'esecuzione di calcoli numerici. Ha il pregio di essere al tempo stesso molto potente e semplice da usare. Il nome Matlab viene da "Matrix laboratory" e indica che inizialmente si trattava di un sistema specifico per il calcolo matriciale. Ancora adesso i dati di Matlab sono fondamentalmente delle matrici: anche gli scalari sono in realtà delle matrici di un solo elemento. Naturalmente con il tempo altri tipi di dati si sono aggiunti, per cui oggi si può usare Matlab per risolvere problemi anche non strettamente numerici.

Questa breve introduzione ha solo lo scopo di fornire allo studente alcune informazioni di base e non intende assolutamente sostituire la documentazione originale di Matlab, in particolare la guida in linea, che viene richiamata con il comando `help` e che fornisce tutte le informazioni necessarie per sfruttare al meglio la grande varietà di strumenti che Matlab mette a disposizione.

B.1 Comandi generali.

All'inizio di una sessione di Matlab sul desktop compaiono diverse finestre. Esaminiamo le più importanti.

- La finestra dei comandi (*Command window*), in cui vengono dati i comandi, vengono scritte le espressioni, vengono richiamate le funzioni. Il *prompt*, cioè il segno `>>`, indica che il sistema è in attesa di ricevere un comando.
  - Su una stessa riga possono essere dati più comandi, separati da punto e virgola.
  - Uno stesso comando può estendersi su più righe, scrivendo ... (tre punti) al termine di ogni riga (esclusa l'ultima) come continuazione.
  - Invece di scrivere una nuova linea si può richiamare una qualunque linea precedente usando il tasto ↑. Se si deve correggere qualche carattere già battuto ci si può muovere sulla riga con i tasti ← e →.
  - Se il comando che viene dato non è corretto il sistema lo segnala. Richiamandolo si può correggere.

  - Può capitare che per errore il sistema si blocchi. In tal caso è possibile ripristinare la situazione normale premendo insieme i tasti Ctrl e Break.
  - le righe di commento devono iniziare con il carattere %.

- La finestra della cronologia dei comandi (*Command history*) riporta tutte le righe che sono state inserite nella finestra dei comandi, indicando il giorno e l'ora di inizio di ogni sessione. Le righe di questa finestra possono essere selezionate, copiate nella finestra dei comandi e rieseguite.

- La finestra dello spazio di lavoro (*Workspace*) contiene i nomi e le specifiche di tutte le variabili che sono state definite nella sessione corrente. Selezionando il nome di una variabile se ne può vedere il contenuto.

- La finestra della directory corrente (*Current directory*), in cui sono elencati tutti i file che vengono salvati, per esempio gli `m-file`, le figure, le copie di testo e i file di variabili.

- La finestra di lancio (*Launch pad*) fornisce l'accesso alla documentazione e ai programmi di dimostrazione (*demo*).

- Una o più finestre grafiche possono essere aperte quando si usano i comandi di Matlab per la grafica. Le diverse finestre sono individuate da un numero progressivo.

Esaminiamo adesso alcuni comandi di carattere generale. Prima di tutto il comando `help` che è necessario utilizzare per avere informazioni dettagliate su tutti gli altri comandi. Se si scrive solo `help` si ottiene una lista di argomenti fondamentali, fra i quali scegliere quello che ci interessa. Per esempio, se si vuole la lista dei comandi di carattere generale si scrive

```
>> help general
```

Si ottiene l'elenco completo e accanto ad ogni comando ne è indicata la funzione in modo conciso. Se si vuole una descrizione più dettagliata di un comando, per esempio il comando `diary`, si utilizza ancora `help`

```
>> help diary
```

Questo comando consente di salvare su disco una copia in formato testo della sessione in corso dal momento in cui viene dato il comando

```
>> diary
```

fino a quando il comando viene revocato con

```
>> diary off
```

Se si vuole salvare in un file su disco i valori correnti delle variabili si utilizza il comando `save`

```
>> save variabili
```

Si possono recuperare le variabili con i valori che avevano al momento del salvataggio con

```
>> load variabili
```

La lista delle variabili correnti si ottiene con il comando

```
>> whos
```

La lista degli m-file si ottiene con il comando

```
>> what
```

B.2 Espressioni matematiche.

Il comando più semplice consiste in espressioni matematiche formate da costanti, operatori aritmetici ed eventualmente le parentesi tonde. Per esempio

```
>> ((2+5*10)/13)^2
ans =
   16
```

Il sistema assegna automaticamente il risultato a una variabile chiamata `ans`. Naturalmente è possibile assegnare il risultato a una variabile che viene memorizzata nello spazio di lavoro

```
>> a=((2+5*10)/13)^2
a =
   16
```

Le variabili a cui è stato assegnato un valore possono a loro volta comparire in espressioni matematiche

```
>> b=a+1
b =
   17
```

Se un'assegnazione termina con il punto e virgola, il risultato non è visualizzato

```
>> b=a+1;
```

Nelle espressioni matematiche possono comparire anche costanti e funzioni elementari predefinite in Matlab, per esempio

```
>> c=pi+log(b)
c =
   5.9748
```

Un elenco delle più importanti costanti e funzioni predefinite in Matlab è riportato nel paragrafo B.8. Gli elenchi completi possono essere ottenuti con

```
>> help elfun
>> help specfun
>> help elmat
>> help matfun
```

Le costanti che compaiono nelle espressioni possono essere scritte con formati diversi: formato intero (per esempio `125`) e formato reale (per esempio `10.47` o `1.36e-2`). Per i risultati il formato può essere specificato usando i comandi `format short`, `format short e`, `format long`, `format long e`, per esempio

```
>> format long
>> c
c =
   5.97480599764601
>> format long e
>> 1/c
ans =
   1.673694510573207\ 10^{-1}$
>> format long; pi
ans =
   3.14159265358979
>> format long e; pi
ans =
    3.141592653589793e+000
>> format short; pi
ans =
   3.1416
```

Il minimo numero positivo e il massimo numero rappresentabili sono

```
>> format long; realmin
ans =
  2.225073858507201e-308
>> realmax
ans =
  1.797693134862316e+308
```

La precisione di macchina è

```
>> eps
ans =
  2.220446049250313e-016
>> 2^-52
ans =
  2.220446049250313e-016
```

Vi sono due funzioni predefinite per la conversione di interi da base 10 a base 2 e viceversa. La prima `dec2bin` trasforma un intero x scritto in base 10 in una stringa di caratteri binari che danno le cifre del numero in base 2. La seconda `bin2dec` fa la conversione opposta.

```
>> x=523; y=dec2bin(x)
y =
  1000001011
>> y='101011100111000101'; x=bin2dec(y)
x =
       178629
```

Un'espressione matematica può contenere anche numeri complessi. L'unità immaginaria è `i`. Ad esempio

```
>> x=1+2i
x =
   1.0000 + 2.0000i
>> y=-3+2*i
y =
  -3.0000 + 2.0000i
```

Risultati complessi possono presentarsi anche quando si usano funzioni di solito usate con argomenti reali

```
>> s=sqrt(-5)
s =
       0 + 2.2361i
>> r=(-1)^0.25
r =
   0.7071 + 0.7071i
>> t=log(-10)
t =
   2.3026 + 3.1416i
```

Le funzioni `conj`, `abs`, `real`, `imag` danno rispettivamente il coniugato, il modulo, la parte reale e la parte immaginaria di un numero complesso

```
>> z=x^2
z =
  -3.0000 + 4.0000i
>> imag(z)
ans =
     4
>> x*conj(x)
ans =
     5
```

B.3 Vettori e matrici

I vettori riga si assegnano come liste ordinate di numeri o espressioni, separati da virgole o spazi, e racchiuse da parentesi quadre, per esempio

```
>> v=[1 4 -5]
v =
     1     4    -5
```

La singola componente di un vettore viene indicata con le parentesi tonde, per esempio

```
>> v(3)
ans =
     -5
```

e può essere modificata semplicemente riassegnandola

```
>> v(3)=6
v =
     1     4     6
```

Il vettore vuoto si assegna così

```
>> w=[]
w =
     []
```

Uno strumento potente che Matlab mette a disposizione è la *colon notation* che rappresenta una scorciatoia per rappresentare vettori riga, per esempio

```
>> 2:5
ans =
     2     3     4     5
```

Con la notazione `a:b:c` si genera un vettore che ha il primo elemento uguale ad `a` e i successivi equispaziati con passo `b` fino al massimo valore che non supera `c`, per esempio

```
>> v=[1:2:10]
v =
     1     3     5     7     9
```

Se il valore indicato come massimo per la colon notation è minore di quello indicato come minimo, il vettore risultante è vuoto. Con la colon notation si possono costruire sottovettori

```
>> v(2:4)
ans =
     3     5     7
>> v(5:-3:1)
ans =
     9     3
```

Un vettore di interi può essere usato per individuare gli indici di un altro vettore. Questo è un altro modo per costruire sottovettori

```
>> w=[1,4,2,1];
>> v(w)
ans =
     1     7     3     1
```

o semplicemente

```
>> v([1,4,2,1])
ans =
     1     7     3     1
```

Con il comando `linspace` si generano vettori con un assegnato numero di componenti equispaziate fra due estremi (compresi). Per esempio un vettore di 10 componenti equispaziate fra 1 e 2 è dato da

```
>> z=linspace(1,2,10)
z =
Columns 1 through 7
  1.0000    1.1111    1.2222    1.3333    1.4444    1.5556    1.6667
Columns 8 through 10
  1.7778    1.8889    2.0000
```

Il numero di elementi del vettore può non essere assegnato. In tal caso il valore di default è 100. Per sapere qual è il numero di componenti di un vettore si usa la funzione predefinita `length`

```
>> length(v)
ans =
   5
```

Oltre a questa vi sono altre funzioni predefinite che agiscono sulle componenti di un vettore: per esempio, `min` (trova il minimo), `max` (trova il massimo), `sum` (calcola la somma), `prod` (calcola il prodotto), `mean` (calcola la media), `norm` (calcola la norma euclidea). Si può sommare o moltiplicare uno scalare per un vettore o si possono sommare due vettori di uguale dimensione

```
>> v1=3*v
  v1 =
     3     9    15    21    27
>> v2=3+v
  v2 =
     4     6     8    10    12
>> w=[1:3:13]
w =
     1     4     7    10    13
>> v3=v1-w
  v3 =
     2     5     8    11    14
```

Il prodotto scalare di vettori viene calcolato con la funzione `dot`

```
>> x=[1 -3 6 2]; y=[3 -1 -2 4];
>> dot(x,y)
ans =
     2
```

Ai vettori possono essere applicate tutte le funzioni predefinite che hanno un solo argomento. Il risultato è il vettore ottenuto applicando le funzioni ad ogni elemento

```
>> format short
>> rz=sqrt(z)
rz =
  Columns 1 through 7
    1.0000    1.0541    1.1055    1.1547    1.2019    1.2472    1.2910
  Columns 8 through 10
    1.3333    1.3744    1.4142
```

Matlab fornisce anche gli operatori *puntuali* `.*` `./` `.^` che estendono le normali operazioni aritmetiche ai singoli elementi di vettori (si noti che queste operazioni puntuali non sono normalmente definite in algebra lineare)

```
>> v1=[3 7 -2]; v2=[2 5 2];
>> w=v1.*v2
w =
     6    35    -4
>> w=v1./v2
w =
    1.5000    1.4000   -1.0000
>> v=[1:5]; w=v.^2
w =
     1     4     9    16    25
```

Il trasposto di un vettore riga è un vettore colonna. L'operatore di trasposizione è l'apice, per esempio

```
>> z=w'
z =
     1
     4
     9
    16
    25
```

Le matrici si assegnano come liste ordinate di numeri o espressioni in cui le virgole o gli spazi separano gli elementi di una stessa riga e i punti e virgola o l'andata a capo separano le righe, per esempio

```
>> a=[1 2 4 -5; 2 -3 6 2; 0 -3 1 5]
a =
     1     2     4    -5
     2    -3     6     2
     0    -3     1     5
```

Le sottomatrici si costruiscono indicando prima il sottoinsieme ordinato degli indici di riga e, separato da una virgola, il sottoinsieme ordinato degli indici di colonna, per esempio

```
>> b=a([1,3],[2 4])
b =
     2    -5
    -3     5
```

Quando tutte le righe o tutte le colonne vengono selezionate, l'insieme può essere indicato semplicemente con la colon notation

```
>> c=a([1 2],:)
c =
     1     2     4    -5
     2    -3     6     2
```

Anche le permutazioni di righe o colonne di una matrice si possono ottenere con la colon notation

```
>> d=a(3:-1:1,:)
d =
     0    -3     1     5
     2    -3     6     2
     1     2     4    -5
```

Con l'apice si ottiene la matrice trasposta, cioè la matrice che ha come righe le colonne della matrice data

```
>> b=a'
b =
     1     2     0
     2    -3    -3
     4     6     1
    -5     2     5
```

Con la colon notation si può costruire il vettore che contiene gli elementi di una matrice (ordinati per colonna)

```
>> va=a(:);  va'
ans =
  Columns 1 through 9
     1     2     0     2    -3    -3     4     6     1
  Columns 10 through 12
    -5     2     5
```

Per cambiare forma a una matrice si usa la funzione `reshape`: gli elementi della matrice data (ordinati per colonna) vengono riallocati nella nuova matrice, di cui vengono specificati il numero di righe e colonne. È necessario che il numero di elementi delle due matrici coincida.

```
>> ar=reshape(a,4,3)
ar =
     1    -3     1
     2    -3    -5
     0     4     2
     2     6     5
```

Si possono costruire matrici particolari indicando le dimensioni (se si indica una sola dimensione la matrice risulta quadrata), per esempio la matrice nulla

```
>> zeros(1,2)
ans =
     0     0
```

la matrice formata da tutte le componenti uguali a 1

```
>> ones(2)
ans =
     1     1
     1     1
```

la matrice identica

```
>> eye(3)
ans =
     1     0     0
     0     1     0
     0     0     1
>> eye(3,2)
ans =
     1     0
     0     1
     0     0
```

la matrice casuale

```
>> rand(3)
ans =
    0.9218    0.4057    0.4103
    0.7382    0.9355    0.8936
    0.1763    0.9169    0.0579
```

la matrice casuale i cui elementi vengono generati a partire da un seme indicato esplicitamente (in questo modo è possibile generare ripetutamente la stessa matrice casuale)

```
>> rand('state',0); m=rand(3,4)
m =
    0.9501    0.4860    0.4565    0.4447
    0.2311    0.8913    0.0185    0.6154
    0.6068    0.7621    0.8214    0.7919
```

Il vettore degli elementi principali di una matrice si ottiene con il comando `diag`

```
>> m=rand(3)
m =
    0.9501    0.4860    0.4565
    0.2311    0.8913    0.0185
    0.6068    0.7621    0.8214
>> v=diag(m)
  v =
    0.9501
    0.8913
    0.8214
```

mentre lo stesso comando, applicato ad un vettore, produce la matrice diagonale i cui elementi principali sono quelli del vettore

```
>> mm=diag(v)
mm =
```

```
    0.9501         0         0
         0    0.8913         0
         0         0    0.8214
```

Gli operatori puntuali `.*` `./` `.^` possono essere applicati anche alle matrici

```
>> a=[1 2 3; 4 5 6; 7 8 9]
a =
     1     2     3
     4     5     6
     7     8     9
>> aa=a.^2
aa =
     1     4     9
    16    25    36
    49    64    81
```

Come si è visto in tutti gli esempi precedenti, non è necessario dichiarare, prima di usarle, quali variabili sono scalari, vettori o matrici e allocare la memoria indicandone la dimensione. La memoria necessaria viene allocata quando richiesto. Questo significa che quando viene eseguita l'istruzione `a(i,j)= ...` per ogni nuovo valore degli indici `i` e `j` viene allocata nuova memoria ad `a`. Se la matrice `a` è molto grossa questo modo di procedere è dispendioso e conviene allocare la memoria richiesta da `a` tutta all'inizio, per esempio inizializzandola a 0

```
>> n=100; a=zeros(n);
```

Lo spazio allocato ad un vettore o ad una matrice può essere disallocato con il comando `clear`

```
>> clear('a')
```

La somma di due matrici si calcola con l'operatore `+`, per esempio

```
>> d=c'+eye(4,2)
d =
     2     2
     2    -2
     4     6
    -5     2
```

Le due matrici devono avere le stesse dimensioni, a meno che una delle due sia uno scalare. La moltiplicazione (righe per colonne) si esegue con l'operatore `*`

```
>> a=[ 1  2  3
       4  5  6
       7  8  9 ];
>> b=[ 1  2 -1
```

```
      -4  5  2
       2  8 -3 ];
>> c=a*b
c =
    -1    36    -6
    -4    81   -12
    -7   126   -18
```

L'inversa di una matrice viene calcolata con la funzione predefinita `inv`

```
>> n=3; a=rand(n)
a =
    0.3795    0.7095    0.1897
    0.8318    0.4289    0.1934
    0.5028    0.3046    0.6822
>> inv(a)
ans =
   -0.9828    1.7926   -0.2349
    1.9775   -0.6877   -0.3549
   -0.1586   -1.0142    1.7975
```

Il rango si calcola con la funzione `rank`. Consideriamo per esempio la matrice formata scrivendo per colonne i primi 16 interi nell'ordine naturale. Qual è il suo rango?

```
>> a=reshape([1:16],4,4)
a =
     1     5     9    13
     2     6    10    14
     3     7    11    15
     4     8    12    16
>> rank(a)
ans =
     2
```

La funzione `size` fornisce le dimensioni di una matrice, la funzione `det` calcola il determinante

```
>> d=diag([1:4])
d =
     1     0     0     0
     0     2     0     0
     0     0     3     0
     0     0     0     4
>> det(d)
ans =
    24
```

Per risolvere i sistemi lineari si usa l'operatore \ (*backslash*)

```
>> a =[ 3     2     1
        2     3     1
        1     2     3  ];
>> b=[39; 34; 26];
>> x=a\b
x =
   9.2500
   4.2500
   2.7500
```

Se la matrice del sistema è singolare o il suo determinante è molto vicino a zero, Matlab dà un messaggio di errore

```
>> a=[ 1     2     3
       4     5     6
       7     8     9  ];
>> b=[6; 15; 24];
>> x=a\b
Warning: Matrix is close to singular or badly scaled.
         Results may be inaccurate. RCOND = 1.541976e-018.
x =
     0
     3
     0
```

In realtà la matrice `a` è singolare, ma il sistema è consistente e la soluzione trovata è accettabile. Per controllo si calcola il prodotto di `a` per `x`

```
>> a*x
ans =
     6
    15
    24
```

Oltre a quella trovata vi sono infinite altre soluzioni: ad esempio $[1, 1, 1]^T$, $[1, 5, -3]^T$, $[1.5, 0, 1.5]^T$, come è facile verificare.

B.4 Comandi condizionali

Molti dei comandi condizionali standard sono disponibili in Matlab. Per esempio, si vogliono calcolare le radici di un'equazione di secondo grado soltanto nel caso che siano reali

```
>> a=1; b=5.25; c=3;
>> delta=b^2-4*a*c;
>> if delta>0 r=sqrt(delta);
     x1=(-b+r)/(2*a),
     x2=(-b-r)/(2*a),
   end
x1 =
   -0.6525
x2 =
   -4.5975
```

Le istruzioni che devono essere eseguite quando la condizione è vera terminano con il comando `end`. Fra `if` e `end` le istruzioni sono separate da virgola se si vuole che il risultato intermedio venga scritto, e da punto e virgola in caso contrario. I seguenti esempi sfruttano la formula di Gregory per il calcolo di π

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots$$

Cominciamo con il calcolare la somma dei primi 100 termini, usando il comando `for` che fa eseguire ripetutamente le istruzioni comprese nel suo ciclo (che termina con `end`)

```
>> s=0;
>> p=1;
>> for k=1:2:199
     s=s+p/k;
     p=-p;
   end;
>> 4*s
ans =
    3.1316
```

È evidente che la serie converge molto lentamente. Si sommano allora i termini della serie fino a quando la differenza fra due somme consecutive è minore di 10^{-4}. In questo caso si usa il comando `while` che consente di eseguire le istruzioni comprese in un ciclo in modo condizionato, cioè fino a quando la condizione specificata risulta vera

```
>> s=0; p=1; k=1;
   while  1/k>10^-4
     s=s+p/k;
     p=-p;
     k=k+2;
   end;
>> 4*s
```

```
ans =
    3.1414
>> k
k =
       10001
```

Per ottenere il valore 3.1414 è stato necessario sommare 5000 termini.

Il comando `find` verifica se una data condizione logica è soddisfatta dagli elementi di un vettore o una matrice (nel caso di una matrice, questa viene considerata come vettorizzata per colonne). Per esempio, si vuole vedere quali componenti di un vettore casuale hanno radice quadrata compresa fra 0.1 e 0.5

```
>> rand('state',0);
>> x=rand(1,20);
>> k=find(sqrt(x)>0.1 & sqrt(x)<0.5)
k =
     2     8    15
>> x(k)
ans =
    0.2311    0.0185    0.1763
```

Matlab dispone di numerose funzioni logiche il cui nome comincia con `is` e che danno risultato 1 (per "vero") o 0 (per "falso"). Ad esempio, la funzione `isprime` che dice se l'argomento è un numero primo, o la funzione `isreal` che dice se l'argomento è un numero reale

```
>> n=503; isprime(n)
ans =
     1
>> a=sqrt(-1); isreal(a)
ans =
     0
```

L'elenco completo delle funzioni `is` si ottiene con

```
>> doc is
```

B.5 m-file

Un `m-file` è formato da un insieme di comandi elencati in un file di testo ed è individuato da un nome, seguito dall'estensione `.m`. Per costruire un file si deve selezionare la voce `file` nel menu di Matlab, la voce `new` (se il file è nuovo) oppure `open` (se esiste già). Viene così aperta un'apposita finestra in cui si costruisce il file.

Ci sono due tipi di `m-file`: gli script e le funzioni. Lo script è il tipo più semplice, non ha parametri di ingresso o uscita e, quando viene richiamato, opera su variabili globali, cioè accede a tutte le variabili dello spazio di lavoro e può modificarle. Per

esempio lo script `quadra.m` costruisce e scrive il vettore dei quadrati dei primi 10 interi

```
numeri=[1:10];
quadrati=numeri.^2
```

Richiamando uno script per mezzo del suo nome (senza l'estensione) si ottiene l'esecuzione dei comandi come se essi fossero stati scritti nella finestra dei comandi

```
>> quadra
ans =
     1     4     9    16    25    36    49    64    81   100
```

La funzione ha parametri di ingresso e uscita. Inizia con il comando `function` seguito dalla lista dei parametri in uscita, dal nome della funzione (che deve coincidere con il nome dell'`m-file`) e dalla lista dei parametri in ingresso. Le variabili che non sono parametri di uscita sono locali alla funzione, cioè stanno in uno spazio di lavoro appositamente creato, che viene cancellato dopo l'esecuzione. Questo vale anche per i parametri di ingresso, a cui al momento della chiamata vengono assegnati valori che stanno nello spazio di lavoro corrente.

La seguente funzione calcola l'elemento di massimo modulo di una matrice e ne trova gli indici

```
function [maxmat,im,jm]=massimo(mat)
[m,n]=size(mat);
temp=0;
for i=1:m
   for j=1:n
      temp1=abs(mat(i,j));
      if temp1>temp temp=temp1;
         im=i; jm=j;
      end
   end
end
maxmat=temp;
```

Si richiama la funzione passando in ingresso una matrice di numeri casuali

```
>> a=rand(4)
a =
    0.3529    0.2028    0.1988    0.9318
    0.8132    0.1987    0.0153    0.4660
    0.0099    0.6038    0.7468    0.4186
    0.1389    0.2722    0.4451    0.8462
>> massimo(a)
ans =
```

```
    0.9318
>> [m,imax,jmax]=massimo(a)
m =
    0.9318
imax =
     1
jmax =
     4
```

Quando i parametri in uscita sono più di uno, la sola chiamata della funzione restituisce solo il primo. Per ottenerli tutti è necessario assegnare la funzione a un vettore.

Se la funzione che vogliamo definire è esprimibile come una sola espressione, si può usare il comando `inline` direttamente nella finestra dei comandi

```
>> f = inline('sin(2*pi*(x+y)^2)')
f =
     Inline function:
     f(x,y) = sin(2*pi*(x+y)^2)
```

Come parametro di una funzione può essere passato anche il nome di un'altra funzione, includendolo fra apici, per esempio

```
>> ris=risolvi('fun',a,b)
```

Nella lista dei parametri di ingresso della funzione `risolvi` il primo parametro, diciamo `f`, riceve il nome della funzione `fun`. Nel corpo della funzione `risolvi` il calcolo dei valori di `f` deve essere eseguito tramite il comando `feval`. Per esempio, la seguente funzione `bisez` implementa il metodo di bisezione per approssimare la soluzione α dell'equazione $f(x) = 0$, con $\alpha \in [a, b]$. Nel corso dell'elaborazione si usano i valori $f(a)$ e $f(b)$, che vengono calcolati con `feval`

```
function r=bisez(f,a,b,tol)
 i=0;
 while b-a>tol
    c=(a+b)/2;
    i=i+1;
    if feval(f,a)*feval(f,c)<0 b=c; else a=c; end;
 end;
 r=c;
```

Utilizziamo la `bisez` per approssimare la soluzione dell'equazione $\cos x = 0$ nell'intervallo $[1, 2]$

```
>> r=bisez('cos',1,2,10^-3)
r =
    1.5713
```

B.6 Grafici

Matlab consente di ottenere grafici di funzioni con il comando `plot`. Il grafico compare nella finestra grafica corrente. La prima volta che viene dato il comando `plot` viene aperta una finestra e ogni nuovo comando `plot` sulla stessa finestra cancella il grafico precedente, a meno che non si usi il comando

```
>> hold on
```

che viene revocato con

```
>> hold off
```

Una nuova finestra grafica viene aperta con

```
>> figure
```

e gli viene assegnato un numero progressivo. Una finestra aperta precedentemente viene chiusa con il comando `close` seguito dal numero della finestra che si vuole chiudere, per esempio

```
>> close 3
```

Per far diventare corrente una finestra aperta precedentemente, basta "cliccarla".

I grafici bidimensionali sono ottenuti congiungendo con una spezzata punti del piano le cui coordinate sono le componenti di due vettori: il primo contiene le ascisse e il secondo contiene le ordinate.

```
>> x=linspace(0,10,100);
>> y=cos(x);
>> plot(x,y)
```

Si ottiene così il grafico del coseno sull'intervallo $[0, 10]$ (figura B.1). Il grafico appare sullo schermo di colore blu. Per ottenere un grafico di colore diverso si deve specificare un parametro. La lista completa dei parametri da usare con la `plot` si ottiene con

```
>> help plot
```

Si scopre così che si possono ottenere grafici con linee tratteggiate oppure indicando i punti con opportuni simboli. Ad esempio con

```
>> x=linspace(0,10,20);
>> y=cos(x);
>> plot(x,y,'o-.')
```

si ottiene il grafico della figura B.2. Se si disegnano grafici di funzioni diverse in tempi successivi, ogni nuovo grafico sostituisce il precedente. Si possono sovrapporre più grafici disegnandoli contemporaneamente. Ad esempio con

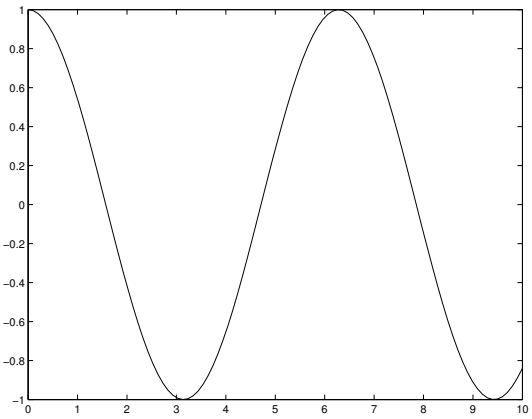


Figura B.1: Grafico di $\cos x$.

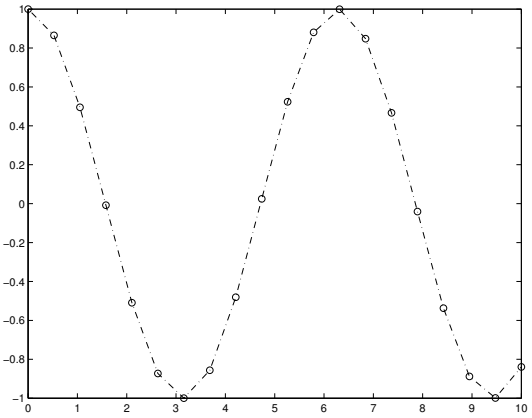


Figura B.2: Grafico di $\cos x$ tratteggiato e con pallini.

```
>> x=linspace(0,10,100);
>> y=cos(x);
>> z=sin(x);
>> plot(x,y,'k-',x,z,'k--')
```

si ottiene la sovrapposizione dei grafici del coseno e del seno sull'intervallo $[0, 10]$ (figura B.3). I parametri specificano che il grafico del coseno è di colore nero e linea continua, il grafico del seno è di colore nero e linea tratteggiata. Per default Matlab non disegna gli assi dei grafici. Un sistema facile per ottenere l'asse x è quello di sovrapporre al nostro grafico il grafico della retta di equazione $y = 0$.

```
>> plot(x,y,'k-',[0,10],[0,0],'k-')
```

In questo modo si ha la sovrapposizione del grafico del coseno con la retta passante per i punti (0,0) e (10,0), in pratica l'asse x.

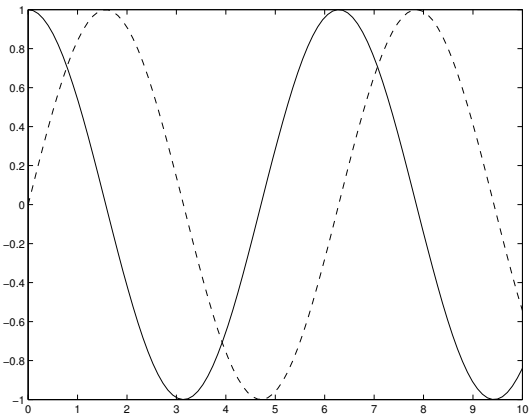


Figura B.3: Grafici sovrapposti di $\sin x$ e $\cos x$.

B.7 Risoluzione di equazioni.

Matlab mette a disposizione delle funzioni predefinite per il calcolo delle soluzioni di equazioni.

(a) La funzione `fzero` calcola una soluzione di $f(x) = 0$, dove $f(x)$ è una qualunque funzione di una variabile. Come parametri di ingresso si devono fornire la funzione e un'approssimazione della soluzione cercata. Ad esempio l'equazione

$$f(x) = 2.2 + \cos^3 x - \sqrt{x} = 0,$$

ha diverse soluzioni comprese fra 2 e 7. Vogliamo approssimare quella più vicina a 2

```
>> f=inline('2.2+cos(x).^3-sqrt(x)');
>> y=fzero(f,2)
y =
    2.5713
```

Se una scelta non felice del punto iniziale porta ad individuare un intervallo che non contiene soluzioni reali si ha un messaggio di errore, come il seguente

```
>> y=fzero(f,1)
Exiting fzero: aborting search for an interval containing a sign
change because complex function value encountered during search
(Function value at -0.28 is 3.0877-0.52915i)
Check function or try again with a different starting value.
```

(b) La funzione `roots` calcola tutte le soluzioni (reali e non) di un'equazione algebrica di grado n della forma

$$f(x) = a_1 x^n + a_2 x^{n-1} + \ldots + a_n x + a_{n+1} = 0.$$

Basta fornire il vettore dei coefficienti. Ad esempio, troviamo le soluzioni dell'equazione

$$f(x) = 3x^5 + 5x^4 - x^3 + 4x^2 + 1 = 0.$$

```
>> a=[3 5 -1 4 0 1];
>> rad=roots(a)
rad =
  -2.1324
   0.3535 + 0.6455i
   0.3535 - 0.6455i
  -0.1207 + 0.5235i
  -0.1207 - 0.5235i
```

L'equazione ha una sola soluzione reale. Le quattro soluzioni non reali sono coniugate a due a due.

B.8 Sommario dei comandi

Il seguente elenco di comandi, di costanti e funzioni predefinite comprende solo una minima parte di quelli disponibili in Matlab. Per ulteriori informazioni riferirsi sempre alla guida in linea o al manuale di riferimento.

Comandi di gestione	
`help`	guida in linea
`lookfor`	lista di tutte le voci della guida in linea che contengono una data parola chiave
`doc`	stampa la documentazione html
`whos`	elenca tutte le variabili correnti
`what`	elenca gli `m-file`
`save`	salva in un file su disco i valori delle variabili dello spazio di lavoro
`load`	recupera da file su disco i valori di variabili precedentemente salvate
`clear`	cancella i valori delle variabili dello spazio di lavoro
`diary`	salva su disco una versione in formato testo della sessione di Matlab in corso
`clc`	cancella la finestra dei comandi (ma non la storia passata)
`type`	lista dei comandi di un `m-file`
`disp`	scrive un testo o una matrice
`format`	definisce il formato di uscita
`tic`	accende il cronometro
`toc`	spenge il cronometro e dà il tempo trascorso in secondi

Caratteri speciali	
%	commenti
=	assegnazione
.	punto decimale
+	addizione
−	sottrazione
*	moltiplicazione
/	divisione
^	elevamento a potenza
.*	moltiplicazione vettorizzata
./	divisione vettorizzata
.^	elevamento a potenza vettorizzato
( )	indica la precedenza delle operazioni aritmetiche o racchiude gli argomenti di una funzione
[ ]	racchiude gli elementi di un vettore o di una matrice
,	separa gli elementi di un vettore o gli argomenti di una funzione
...	caratteri di continuazione di un comando su più righe
;	separa le righe di una matrice, al termine di un comando sopprime la stampa
:	colon notation
>	maggiore di
>=	maggiore o uguale a
<	minore di
<=	minore o uguale a
==	uguale a (operatore di relazione)
=	non uguale a
&	and
|	or
~	not
'	trasposto
\	operatore di *backslash*, per risolvere i sistemi lineari

Costanti predefinite	
`pi`	π
`i`, `j`	unità immaginaria
`eps`	precisione di macchina
`realmax`	massimo numero di macchina
`realmin`	minimo numero di macchina positivo
`inf`	speciale costante di macchina che rappresenta ∞
`NaN`	speciale configurazione di macchina che segnale una situazione anomala

Funzioni scalari predefinite	
`abs`	valore assoluto o modulo
`sqrt`	radice quadrata
`gcd`	massimo comun divisore
`lcm`	minimo comune multiplo
`rem`	resto della divisione fra interi
`round`	arrotonda all'intero più vicino
`floor`	arrotonda all'intero immediatamente inferiore
`ceil`	arrotonda all'intero immediatamente superiore
`factorial`	fattoriale di un intero
`sign`	segno
`exp`	esponenziale
`log`	logaritmo naturale
`log10`	logaritmo in base 10
`log2`	logaritmo in base 2
`sin`, `asin`	seno, arcoseno
`cos`, `acos`	coseno, arcocoseno
`tan`, `atan`	tangente, arcotangente
`conj`	coniugato di un numero complesso
`real`	parte reale di un numero complesso
`imag`	parte immaginaria di un numero complesso

Matrici speciali	
`zeros`	matrice nulla
`ones`	matrice di elementi uguali a 1
`eye`	matrice identica
`diag`	estrae la diagonale di una matrice o costruisce una matrice diagonale
`linspace`	vettore di elementi equispaziati
`logspace`	vettore di elementi spaziati in scala logaritmica
`rand`	matrice di elementi random
`hilb`	matrice di Hilbert
`invhilb`	inversa della matrice di Hilbert
`vander`	matrice di Vandermonde
`pascal`	matrice di Pascal

Funzioni predefinite di vettori e matrici	
`length`	dimensione di un vettore
`find`	trova gli indici degli elementi di un vettore che soddisfano alla condizione indicata
`sort`	mette in ordine crescente gli elementi di un vettore (*)
`max`	massimo elemento di un vettore (*)
`min`	minimo elemento di un vettore (*)
`sum`	somma degli elementi di un vettore (*)
`prod`	prodotto degli elementi di un vettore (*)
`mean`	media aritmetica degli elementi di un vettore (*)
`dot`	prodotto scalare di vettori
`size`	dimensioni di un vettore o di una matrice
`trace`	traccia di una matrice
`det`	determinante di una matrice
`rank`	rango di una matrice
`norm`	norma 2 di vettore o di una matrice
`cond`	numero di condizionamento in norma 2 di una matrice
`inv`	inversa di una matrice
`poly`	applicata ad un vettore dà i coefficienti del polinomio avente come radici gli elementi del vettore, applicata ad una matrice dà i coefficienti del polinomio caratteristico della matrice
`eig`	autovalori e autovettori di una matrice
`svd`	decomposizione ai valori singolari di una matrice
`kron`	prodotto tensore di matrici
`reshape`	rialloca gli elementi di una matrice
(*) la funzione può essere applicata anche ad una matrice: in tal caso il risultato è un vettore la cui i-esima componente è il valore della funzione applicata all'i-esima colonna della matrice.	

Comandi di programmazione	
`if`	comando di esecuzione condizionale
`else`, `elseif`	usati con `if`
`end`	terminazione di `if`, `for` e `while`
`for`	ripetizione per un numero prefissato di volte
`while`	ripetizione condizionale
`break`	interruzione di un ciclo di `for` o di `while`
`function`	definisce una funzione
`feval`	calcola una funzione all'interno di una `function`
`return`	uscita da una `function`
`isprime`	controlla se l'argomento è un numero primo
`isreal`	controlla se l'argomento è un numero reale

Metodi numerici	
`roots`	radici di un polinomio con assegnati coefficienti
`polyval`	calcola il valore in un punto di un polinomio con assegnati coefficienti
`polyfit`	calcola i coefficienti del polinomio di grado assegnato approssimante ai minimi quadrati
`polyder`	calcola i coefficienti della derivata di un polinomio
`fzero`	approssima uno zero di una funzione
`quad`	calcola l'integrale con la formula adattiva di Cavalieri-Simpson
`quadl`	calcola l'integrale con la formula adattiva di Lobatto
`ode23`	risolve un'equazione differenziale ordinaria con un metodo di Runge-Kutta del secondo ordine e controllo adattivo del passo
`ode45`	risolve un'equazione differenziale ordinaria con un metodo di Runge-Kutta del quarto ordine e controllo adattivo del passo

Funzioni di grafica	
`plot`	disegna la spezzata passante per i punti assegnati
`title`	titolo del grafico
`xlabel`	etichetta dell'asse x
`ylabel`	etichetta dell'asse y
`figure`	crea una nuova figura
`close`	chiude una figura
`hold`	mantiene aperto il grafico corrente
`axis`	controlla la scala e l'aspetto degli assi
`meshgrid`	costruisce una matrice per grafici tridimensionali
`mesh`	disegna una superficie
`surf`	disegna una superficie ombreggiata
`contour`	disegna le linee di livello di una superficie
`contourf`	disegna e riempie le linee di livello di una superficie
`spy`	indica gli elementi non nulli di una matrice