Corso di Laurea in Ingegneria Biomedica

Insegnamento di "Informatica Medica"

a.a. 2012-13 - I semestre

Elementi di Calcolo Numerico

Prof. Silvio P. Sabatini

Indice

Capitolo 1

Introduzione al calcolo numerico

La trattazione dei semplici metodi ed algoritmi di calcolo numerico presentata nel seguito in queste note non richiedono al lettore specifiche competenze teoriche sull'argomento. Tuttavia, riteniamo utile in questo capitolo introdurre alcune problematiche legate alle elaborazioni numeriche sui calcolatori. In particolare, consideriamo (i) le approssimazioni dovute all'aritmetica a precisione finita dei calcolatori e alla discretizzazione dei problemi analitici, e (ii) l'analisi degli errori associati a tali approssimazioni.

1.1 Rappresentazione dei numeri in un calcolatore

Parlando di numeri ed operazioni in un calcolatore, sorge l'interrogativo su quante cifre si possono utilizzare per la loro rappresentazione. Noi siamo abituati a non porci tale problema (basta aggiungere cifre!!), ma in un calcolatore il numero di dispostivi hardware che possono contenere e manipolare le cifre è finito. Pertanto, siamo costretti a riservare alla rappresentazione di ogni numero uno "spazio" finito di memoria costituito da una sequenza finita di *bit* (*binary digit*) o *byte* (gruppi di 8 bit).

Ogni calcolatore permette al programmatore di scegliere tra diverse convenzioni di rappresentazione o *tipi di dato*. I tipi di dato possono differire nel numero di bit utilizzati (lunghezza della parola), ma anche nel formato di rappresentazione. Si noti comunque, che qualunque sia la codifica prescelta, la rappresentazione dei numeri reali nel calcolatore è soggetta ad approssimazioni; tali approssimazioni, come vedremo, si propagano nel corso dell'esecuzione delle operazioni, causando errori numerici anche importanti. Il **calcolo numerico** è la disciplina che studia le proprietà dell'esecuzione delle operazioni tramite calcolatore, consentendo di valutare l'entità degli errori numerici introdotti durante l'esecuzione delle operazioni.

Considerato il numero finito di byte disponibili per la codifica dei numeri in un calcolatore, possiamo rappresentare solo quei numeri che, in base alla convenzione di rappresentazione scelta, possono essere contenuti nello spazio previsto per ciascuno di essi; questi numeri sono detti *numeri di macchina*.

1.1.1 Numeri interi

La rappresentazione di un numero intero comporta la rappresentazione del segno del numero oltre al suo valore. Se n bit sono disponibili per rappresentare un intero, la rappresentazione in modulo e segno[1] utilizza il primo bit (la posizione più a sinistra) come bit di segno; per convenzione, 0 indica un numero positivo e 1 un numero negativo. Con questa convenzione, quando sono disponibili n bit come lunghezza fissata per contenere i dati, possono essere rappresentati gli interi compresi fra $-(2^{n-1}-1)$ e $+(2^{n-1}-1)$. Infatti, essendo un bit utilizzato per la rappresentazione del segno, solo $n-1$ bit sono disponibili per rappresentare il modulo del numero (v. Tab. 1.1).

Tipo	**Allocazione in memoria**	**Accuratezza**	**Insieme di rappresentazione**
`signed int`	2	-	$\{-2^{15}-1, 2^{15}-1\}$
`unsigned int`	2	-	$\{0, 2^{16}-1\}$
`float`	4	6 cifre dec.	$\pm\{10^{-38}, 10^{38}\} \cup \{0\}$
`double`	8	15 cifre dec.	$\pm\{10^{-308}, 10^{308}\} \cup \{0\}$

Tabella 1.1: I valori riportati sono solo indicativi e possono variare da macchina a macchina

Compatibilmente con in numero di bit (cifre) disponibili, la rappresentazione di un numero intero di un calcolatore risulta esatta. Anche l'aritmetica sui numeri interi risulta esatta osservando tuttavia che (i) il risultato di una divisione tra interi produce ancora un intero, con conseguente perdita della parte frazionaria del risultato (del resto della divisione) e che (ii) il risultato può essere al di fuori dell'intervallo di rappresentazione. Ad esempio, con numeri a 8 bit, si potrebbe verificare un inconveniente come quello qui evidenziato:

```
  0000 0001  +          1  +
  1111 1111  =        255  =
1 0000 0000             0
```

in cui la prima cifra, che è la più significativa, viene inesorabilmente persa portando a un risultato manifestamente errato. Tale situazione prende il nome di *overflow*.

Per comprendere le proprietà dell'aritmetica finita è utile far ricorso ad una rappresentazione grafica degli interi analoga a quella, basata su una retta, utilizzata per l'aritmetica comune. Sulla retta di Fig. 1.1 sono rappresentati gli interi: un'operazione di somma tra numeri positivi consiste nello spostarsi su tale retta, verso destra a partire dal primo addendo, di un segmento di lunghezza pari al valore del secondo addendo. Lavorando in aritmetica finita, se si supera il limite della rappresentazione si ritorna al valore iniziale, come nell'esempio precedente. Quindi, la rappresentazione

[1] Esistono diverse altre codifiche binarie di numeri interi. Tra queste, una delle più utilizzate per la semplificazione dell'esecuzione di operazioni aritmetiche tra numeri interi, è la *rappresentazione in complemento a 2*.

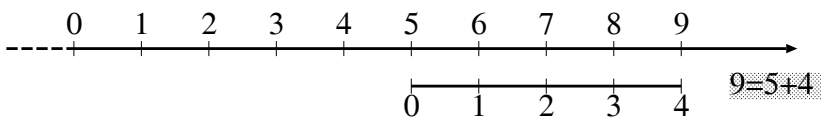


Figura 1.1: La retta dei numeri interi

non sarà più una retta, bensì una circonferenza. La Fig. 1.2 illustra la rappresentazione binaria su 3 bit. In essa, uno spostamento di un segmento di lunghezza 4 a partire da 5 (`101`) porterebbe ad attraversare lo zero, arrivando a 1. Infatti il risultato corretto, 9, non è rappresentabile su 3 bit e l'operazione dà luogo a overflow.

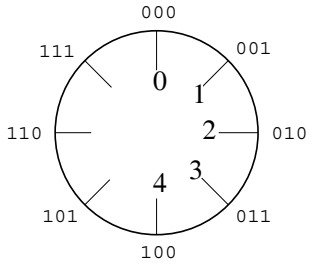


Figura 1.2: Rappresentazione degli interi in aritmetica finita

1.1.2 Numeri reali

In generale, un numero reale può essere rappresentato solo in forma approssimata con arbitraria precisione mediante allineamenti finiti di cifre. Dovendo utilizzare un calcolatore digitale per calcoli scientifici di massima precisione, con valori molto grandi o molto piccoli, il principale problema da affrontare è la limitattezza dello spazio (cioè il numero di bit) a disposizione per rappresentare i numeri. Naturalmente, la strada da seguire non è quella di aumentare arbitrariamente il numero di bit impiegati per rappresentare ciascun numero. Si ricorre invece a un particolare tipo di notazione denominata in *virgola mobile* (*floating point*) che utilizza la notazione esponenziale per la codifica dei numeri reali; per questa rappresentazione non esiste un unico standard, perciò consideriamo una delle tante possibili convenzioni. Secondo tale rappresentazione, ogni numero reale x può essere scritto nella forma:

$$x = mB^e$$

dove m è un numero reale, detto *mantissa*, B è la *base* del sistema di numerazione scelto ($B = 10$ per il sistema decimale, $B = 2$ per il sistema binario), ed e è un *esponente* intero positivo, negativo o nullo. Questa rappresentazione non è unica; infatti abbiamo

$$x = mB^e = m'B^{e-1} = m''B^{e+1} = \ldots, \qquad m' = mB, \;\; m'' = m/B.$$

Si noti che l'insieme dei numeri generabili con questa notazione è un sottoinsieme dei numeri reali, distribuito in modo non uniforme; precisamente, vi saranno valori

estremamente vicini tra loro nell'intorno del numero 0 ed estremamente lontani fra loro nell'intorno del numero massimo esprimibile, positivo o negativo.

La notazione esponenziale consente di scrivere in poco spazio numeri molto grandi o molto piccoli evitando di sprecare un numero elevato di bit con valori nulli atti soltanto ad indicare l'ordine di grandezza del numero. Aumentando così il numero di bit significativi, aumenta la *precisione* del numero. In pratica, nei calcolatori digitali i byte destinati a memorizzare un numero reale sono così suddivisi:

- un bit per il segno della mantissa (e quindi dell'intero numero);
- t bit per la mantissa ($m_i < m < m_s$);
- q bit per l'esponente ($M_i < e < M_s$).

La rappresentazione di $x \neq 0$ si dice normalizzata quando

$$B^{-1} \leq |m| < 1 \ .$$

Fissata la base B, ogni numero reale $x \neq 0$ è univocamente definito dalla coppia (m, e), per cui è sufficiente memorizzare quest'ultima. Non esistendo la rappresentazione normalizzata del numero $x = 0$, per convenzione si individua tale numero con la coppia $(0, 0)$ (v. Tab. 1.1).

Le operazioni su numeri reali rappresentati con un numero finito di cifre possono dare luogo a fenomeni di *overflow* (numeri con esponente $e > M_s$) o di *underflow* (numeri con esponente $e < M_i$).

1.2 Approssimazioni ed errori

1.2.1 Problemi numerici e algoritmi

Per **problema numerico** intendiamo una descrizione chiara e non ambigua di una relazione funzionale tra i dati (*input*) del problema (che costituiscono le variabili indipendenti) e i risultati desiderati (*output*) (v. Fig. 1.3). I dati x e i risultati y devono essere rappresentabili da vettori (di dimensione finita) di numeri. La relazione funzionale f può essere espressa sia in forma esplicita $y = f(x)$, sia implicita $f(x, y) = 0$.

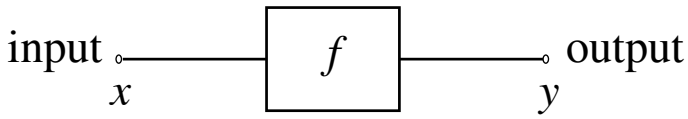


Figura 1.3: Schematica rappresentazione di un problema numerico

Per **algoritmo** di un problema numerico intendiamo una descrizione completa e ben definita di operazioni che permetta di trasformare (in un numero finito di passi) ogni vettore di dati (permissibili) x nel corrispondente output y^*, non necessariamente

uguale a y. Ad ogni problema numerico possiamo associare più algoritmi, ognuno dei quali in genere fornirà risultati con precisione diversa.

Il problema numerico è caratterizzato dalla sola relazione funzionale esistente tra input e output; in un algoritmo invece, ogni singolo passo (operazione) deve essere definito chiaramente ed inequivocabilmente, dai dati sino alla determinazione effettiva dei risultati.

Nei capitoli che seguono parleremo di problemi e di algoritmi; in particolare esamineremo algoritmi per il calcolo della soluzione di problemi numerici spesso ottenuti approssimando (o meglio, *discretizzando*) modelli matematici.

Generalmente, tra la soluzione analitica e quella numerica, si osserva una discrepanza, o *errore*, dovuta al fatto che le tecniche numeriche presuppongono un'approssimazione. Errori di questo tipo sono caratteristici della maggior parte delle tecniche descritte in queste note. Ciò può sembrare a prima vista contrario rispetto a quanto ci si aspetta da una tecnica ingegneristica valida. Gli studenti e i consulenti di ingegneria, per esempio, tentano costantemente di limitare gli errori nel proprio lavoro: quando lo studente sostiene esami o esegue esercizi, viene penalizzato per gli errori commessi. Nella pratica professionale, poi, gli errori possono costare parecchio e possono risultare addirittura catastrofici: il cedimento di una struttura o di una macchina, ad esempio, può comportare la perdita di vite umane.

Sebbene la perfezione sia un traguardo a cui mirare, essa, in pratica, non viene mai raggiunta. I metodi numerici possono introdurre simili discrepanze nell'analisi; anche in questo caso la domanda è quale sia l'errore tollerabile. Nel seguito sono trattati gli argomenti fondamentali relativi all'identificazione, quantificazione e minimizzazione di tali errori. In particolare verranno descritti i due tipi principali di errori: di arrotondamento e di troncamento. Gli *errori di arrotondamento* sono dovuti al fatto che i calcolatori dispongono di un numero finito di cifre per rappresentare i dati. L'*errore di troncamento* è la discrepanza introdotta dal fatto che i metodi numerici utilizzano delle approssimazioni per eseguire calcoli matematici e rappresentare quantità esatte. Per esempio, moltissimi metodi numerici richiederebbero l'esecuzione di una sequenza infinita di passi di calcolo: essi sono perciò "troncati" dopo un numero di passi finito, quando si è raggiunta un'approssimazione sufficiente dei risultati.

1.2.2 Definizione degli errori

Gli *errori numerici* sono dovuti all'uso di approssimazioni nella rappresentazione di quantità esatte e nell'esecuzione di calcoli matematici. Gli *errori di troncamento* dipendono dal fatto che una certa procedura matematica viene approssimata da una serie di operazioni aritmetiche più semplici, mentre gli *errori di arrotondamento* risultano dall'uso di un numero limitato di cifre significative per la rappresentazione dei numeri. In entrambi i casi, la relazione tra il risultato esatto, o vero, e l'approssimazione può essere così formulata:

$$\text{valore vero} = \text{approssimazione} + \text{errore}$$

da cui troviamo che l'errore numerico è uguale alla differenza tra il valore vero e l'approssimazione:

$$e = \text{valore vero} - \text{approssimazione} \tag{1.1}$$

dove e indica l'*errore assoluto*.

Questa definizione presenta uno svantaggio: non tiene in nessun conto l'ordine di grandezza del valore che si sta esaminando. Per esempio, un errore di un centimetro è molto più significativo quando si misura la lunghezza di un chiodo piuttosto che quella di un ponte. Per tener conto dell'ordine di grandezza delle quantità in esame si può normalizzare l'errore rispetto al valore vero:

$$\epsilon = \frac{\text{errore}}{\text{valore vero}} = \frac{e}{\text{valore vero}}$$

L'errore relativo può anche essere moltiplicato per 100 in modo da esprimerlo come frazione di 100 e non di 1:

$$\epsilon_p = \frac{\text{errore}}{\text{valore vero}} 100\%$$

dove ϵ_p viene definito *errore relativo percentuale*.

Nel caso dei metodi numerici, il valore vero è noto solo quando si ha a che fare con funzioni che possono essere espresse analiticamente, come nello studio teorico del comportamento di una particolare tecnica. Nelle applicazioni reali, invece, il risultato esatto non è mai noto a priori: in questi casi, l'alternativa è normalizzare l'errore assoluto usando la migliore stima disponibile del valore vero; normalizzare, cioè, rispetto all'approssimazione stessa

$$\epsilon_a = \frac{\text{valore vero} - \text{approssimazione}}{\text{approssimazione}} \tag{1.2}$$

dove il pedice a significa che l'errore è normalizzato rispetto a un valore approssimato. Nelle applicazioni reali, inoltre, la (1.1) non può essere usata per calcolare il termine relativo all'errore contenuto nella (1.2): una delle difficoltà che si incontrano nell'applicazione dei metodi numerici sta nella determinazione di stime dell'errore in mancanza di nozioni circa il valore vero. Certi metodi numerici, per esempio, arrivano a un risultato procedendo per iterazioni; secondo questo tipo di approccio, l'approssimazione più recente viene calcolata sulla base della precedente. Questo processo viene ripetuto, o iterato, per calcolare approssimazioni sempre migliori. In questi casi, la stima dell'errore che viene assunta è semplicemente la differenza tra il risultato dell'ultima iterazione e quello della precedente. L'errore relativo, quindi, viene determinato secondo la formula

$$\epsilon_a = \frac{\text{approssimazione attuale} - \text{approssimazione precedente}}{\text{approssimazione attuale}} \tag{1.3}$$

Questo e altri approcci per la stima degli errori verranno utilizzati nei prossimi capitoli. I valori che si ottengono dalle espressioni (1.1)-(1.3) possono essere sia positivi che negativi; se l'approssimazione è maggiore del valore vero (o la penultima approssimazione è maggiore dell'ultima), l'errore è negativo; in caso contrario, l'errore

è positivo. Inoltre per le espressioni dell'errore relativo, il denominatore può essere minore di zero e anche questa condizione può portare a un errore negativo. Spesso, quando si eseguono dei calcoli il segno dell'errore può non essere interessante, ma ciò che realmente importa è che il valore assoluto dell'errore sia minore di una tolleranza ϵ_s specificata in partenza. Pertanto, è spesso utile fare riferimento al valore assoluto delle espressioni (1.1)-(1.3); in tali casi, il calcolo viene proseguito finché si ha:

$$|\epsilon_a| < \epsilon_s$$

Se questa condizione è verificata, si assume che il risultato sia corretto entro la tolleranza ϵ_s prefissata.

1.2.3 Errori di arrotondamento, operazioni di macchina

Se x rappresenta un valore esatto e $\bar{x}$ una sua approssimazione, l'errore assoluto e l'errore relativo associati a $\bar{x}$ sono definiti rispettivamente dalle quantità:

$$|x - \bar{x}| \quad \text{e} \quad \left|\frac{x - \bar{x}}{x}\right|, \;\; x \neq 0;$$

inoltre, possiamo scrivere

$$\bar{x} = x(1 + \epsilon), \qquad \epsilon = \frac{\bar{x} - x}{x}.$$

Quando

$$|x - \bar{x}| \leq \frac{1}{2} B^{-k}, \qquad k \geq 1,$$

diciamo che l'approssimazione $\bar{x}$ ha k "decimali" corretti (nella base B), e definiamo *significative* le cifre che precedono il $(k+1)$-esimo decimale (escludendo gli eventuali zeri iniziali). Le cifre significative coincidono con i decimali corretti presenti nella mantissa $\bar{m}$ della rappresentazione (normalizzata) $\bar{x} = \bar{m} B^e$. Se la mantissa $\bar{m}$ ha k decimali corretti possiamo scrivere:

$$|m - \bar{m}| \leq \frac{1}{2} B^{-k};$$

quindi, ricordando che $|m| < 1$, se vale una disuguaglianza del tipo

$$\frac{|x - \bar{x}|}{|x|} \leq \frac{1}{2} B^{-k}$$

possiamo senz'altro affermare che l'approssimazione $\bar{x}$ ha almeno k cifre significative. Infatti, dall'ultima relazione deduciamo

$$|m - \bar{m}| \leq \frac{1}{2} B^{-k} |m| < \frac{1}{2} B^{-k}.$$

Diciamo "almeno k cifre" perché in realtà secondo la nostra definizione potrebbero essere $k + 1$.

Come abbiamo visto nel paragrafo 1.1.2, in un elaboratore non tutti i numeri reali possono venire rappresentati. Nel caso di un sistema floating-point con t cifre (nella base B) per la mantissa, tutti i numeri che nella base scelta ammettono una rappresentazione (normalizzata) con un numero di cifre nella mantissa superiore a t dovranno in qualche modo venire "accorciati", o meglio, arrotondati a t cifre. Le operazioni di arrotondamento usate sono essenzialmente due:

(i) $T_t(x) = B^{-t}\lfloor xB^t \rfloor$ *"trunc"*,
esclude la parte a destra della t-esima cifra;

(ii) $R_t(x) = B^{-t}\lfloor xB^t + \frac{1}{2} \rfloor$ *"round"*,
aggiunge $\frac{1}{2}B^{-t}$ alla mantissa in questione e poi tronca quest'ultima alla t-esima cifra;

dove $\lfloor x \rfloor$ è l'operatore "parte intera" del numero reale x:

$$\lfloor x \rfloor = \max\{n \in Z \quad : \quad n \leq x\}$$

che restituisce il più grande intero più piccolo di x.

Esempio. Supponendo $t = 6$ e $B = 10$, la mantissa

$$m = 0.74749563$$

verrebbe arrotondata a

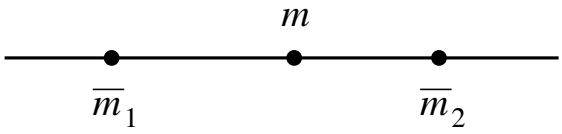
$$\bar{m} = 0.747495$$

adottando la tecnica (i), e a

$$\bar{m} = 0.747496$$

con la (ii).

Le mantisse $\bar{m}$ dei numeri di macchina, $B^{-1} \leq |\bar{m}| < 1$, non hanno più di t cifre, e la distanza tra due mantisse di macchina consecutive è esattamente B^{-t}. Con l'operatore *trunc*, tutte le mantisse m comprese tra due mantisse di macchina consecutive $\bar{m}_1$ e $\bar{m}_2 = \bar{m}_1 + B^{-t}$, positive per esempio,



vengono sostituite da $\bar{m}_1$; in questo caso abbiamo $|m - \bar{m}_1| < B^{-t}$. Con l'operatore *round* invece, tutte le mantisse comprese nell'intervallo $(\bar{m}_1, \bar{m}_1 + \frac{1}{2}B^{-t})$ vengono sostituite con $\bar{m}_1$, mentre le mantisse situate in $[\bar{m}_1 + \frac{1}{2}B^{-t}, \bar{m}_2)$ vengono approssimate con $\bar{m}_2$, così che, denotando con $\bar{m}$ la mantissa di macchina che il criterio di arrotondamento associa a m, risulta $|m - \bar{m}| \leq \frac{1}{2}B^{-t}$.

Dei due metodi, il primo è certamente il peggiore (ma il meno oneroso), non solo perché può provocare un errore maggiore (doppio rispetto al secondo), ma anche perché il segno di $(m - \bar{m})/m$ è costante (positivo).

Sia $x = mB^e$, $B^{-1} \leq |m| < 1$, un numero reale, e sia $\bar{x} = \bar{m}B^e$ il corrispondente numero di macchina ottenuto mediante una delle due tecniche di arrotondamento introdotte. Poiché

$$|m - \bar{m}| \leq \begin{cases} B^{-t} & \text{con la (i)} \\ \frac{1}{2}B^{-t} & \text{con la (ii)}, \end{cases}$$

per gli errori, assoluto e relativo, associati ad $\bar{x}$ abbiamo rispettivamente

$$|x - \bar{x}| \leq \begin{cases} B^{e-t} & \text{(i)} \\ \frac{1}{2}B^{e-t} & \text{(ii)}, \end{cases}$$

e

$$\frac{|x - \bar{x}|}{|x|} \leq \frac{|x - \bar{x}|}{B^{e-1}} \leq \begin{cases} B^{1-t} & \text{(i)} \\ \frac{1}{2}B^{1-t} & \text{(ii)}. \end{cases}$$

La quantità $\varepsilon_m = B^{1-t}$ nel caso (i), e $\varepsilon_m = \frac{1}{2}B^{1-t}$ in (ii), definisce la cosidetta *precisione di macchina*. Essa è una costante caratteristica di ogni aritmetica floating-point (e tecnica di arrotondamento), e rappresenta la massima precisione di calcolo raggiungibile con il calcolatore su cui tale aritmetica è implementata. Non ha pertanto senso cercare di determinare approssimazioni con precisione relativa inferiore alla quantità ε_m. In altre parole la precisione di macchina è legata al massimo numero di cifre significative relative all'ordine di grandezza dell'esponente. Tipicamente, un calcolatore con $B = 2$ e una lunghezza di parola di 32 bit ha ε_m circa pari a $3 \cdot 10^{-8}$. Si noti infine, che ε_m non corrisponde al più piccolo numero rappresentabile nella macchina. Tale numero dipende infatti dal numero di bit che rappresentano l'esponente, mentre ε_m dipende dal numero di bit che rappresentano la mantissa.

Nonostante la tecnica di arrotondamento (ii) sia migliore, per motivi di semplicità, costi e velocità, in molti calcolatori viene implementata la tecnica (i).

Finora abbiamo definito i numeri di macchina e visto come arrotondare un generico numero reale a numero di macchina. Osserviamo tuttavia che i risultati di operazioni aritmetiche tra numeri di macchina generalmente non sono numeri di macchina; pertanto in un calcolatore risulterà impossibile implementare esattamente le operazioni aritmetiche. Dovremo accontentarci di realizzare le cosidette *"operazioni di macchina"*. L'operazione di macchina associa a due numeri di macchina un terzo numero di macchina, ottenuto arrotondando (con le tecniche (i) o (ii)) l'esatto risultato dell'operazione aritmetica in questione.

Se con $\bar{x} = \mathrm{fl}(x)$ indichiamo l'operazione di arrotondamento, in aritmetica floating-point, di x a numero macchina $\bar{x}$ [2], e con $\oplus$, $\ominus$, $\otimes$, $\oslash$ denotiamo le operazioni di

[2]Ricordiamo che $\bar{x} = x(1 + \epsilon)$, $|\epsilon| \leq \varepsilon_m$.

macchina corrispondenti a quelle aritmetiche $+$, $-$, $\times$, $/$, abbiamo

$$\begin{aligned}
\bar{x} \oplus \bar{y} &= \mathrm{fl}(\bar{x} + \bar{y}) = (\bar{x} + \bar{y})(1 + \epsilon_1) \\
\bar{x} \ominus \bar{y} &= \mathrm{fl}(\bar{x} - \bar{y}) = (\bar{x} - \bar{y})(1 + \epsilon_2) \\
\bar{x} \otimes \bar{y} &= \mathrm{fl}(\bar{x} \times \bar{y}) = (\bar{x} \times \bar{y})(1 + \epsilon_3) \\
\bar{x} \oslash \bar{y} &= \mathrm{fl}(\bar{x}/\bar{y}) = (\bar{x}/\bar{y})(1 + \epsilon_4)
\end{aligned}$$

dove $|\epsilon_i| \leq \varepsilon_m$. L'errore relativo introdotto dalle quattro operazioni aritmetiche di macchina, prescindendo dagli eventuali errori presenti nei due operandi $\bar{x}$ e $\bar{y}$, non supera mai la precisione di macchina ε_m.

E' importante osservare che per le operazioni di macchina non valgono le note proprietà (commutativa, associativa, distributiva, ecc.) delle quattro operazioni aritmetiche. In particolare, mentre la proprietà commutativa

$$\bar{x} \oplus \bar{y} = \bar{y} \oplus \bar{x} \ , \qquad \bar{x} \otimes \bar{y} = \bar{y} \otimes \bar{x}$$

risulta ancora valida, le seguenti <u>non</u> lo sono più:

$$\begin{aligned}
&\bar{x} \oplus (\bar{y} \oplus \bar{z}) = (\bar{x} \oplus \bar{y}) \oplus \bar{z} \ , \qquad \bar{x} \otimes (\bar{y} \otimes \bar{z}) = (\bar{x} \otimes \bar{y}) \otimes \bar{z}, \\
&\bar{x} \otimes (\bar{y} \oplus \bar{z}) = (\bar{x} \otimes \bar{y}) \oplus (\bar{x} \otimes \bar{z}), \\
&(\bar{x} \otimes \bar{y}) \oslash \bar{y} = \bar{x}, \qquad (\bar{x} \oslash \bar{y}) \otimes \bar{y} = \bar{x}, \\
&(\bar{x} \otimes \bar{z}) \oslash \bar{y} = (\bar{x} \oslash \bar{y}) \otimes \bar{z}.
\end{aligned}$$

Un'ulteriore relazione anomala è la seguente:

$$\bar{x} \oplus \bar{y} = \bar{x} \qquad \text{quando } \ |\bar{y}| < \frac{\varepsilon_m}{B}|\bar{x}|.$$

Possiamo pertanto concludere che *le espressioni che sono equivalenti in aritmetica esatta non risultano generalmente tali nelle aritmetiche con precisione finita.* Ciononostante due espressioni (non nulle) saranno definite "equivalenti" dal punto di vista del calcolo numerico quando, valutate in un calcolatore, forniscono risultati che differiscono per una tolleranza relativa dell'ordine della precisione di macchina.

Osservazione. Anche se i dati iniziali di un processo di calcolo sono numeri di macchina, o comunque arrotondabili a numeri di macchina, le operazioni intermedie possono dare origine a fenomeni di *overflow* (numeri con esponente $e > M_s$) o di *underflow* (numeri con $e < M_i$); quando ciò accade, i risultati di tali operazioni non sono rappresentabili ed il calcolatore invia una segnalazione di errore.

Esempi.

$$\begin{aligned}
&z = \sqrt{x \otimes y}, \qquad x = m_1 B^{e_1}, \ \ y = m_2 B^{e_2}, \\
&\text{overflow se } \ e_1, e_2 > \frac{M_s}{2}, \\
&\text{underflow se } \ e_1, e_2 < \frac{M_i}{2};
\end{aligned}$$

$z = \sqrt{x \oslash y}$
overflow se $e_1 > 0, e_2 < 0, e_1 - e_2 > M_s$,
underflow se $e_1 < 0, e_2 > 0, e_1 - e_2 < M_i$.

Propagazione degli errori

Vediamo ora come un errore può propagarsi nell'esecuzione di calcoli in successione. Consideriamo l'addizione di due numeri x e y (i loro valori veri) usando i valori approssimati $\bar{x}$ e $\bar{y}$ a cui associamo rispettivamente gli errori assoluti e_x e e_y. Cominciando con $x = \bar{x} + e_x$ e $y = \bar{y} + e_y$, la somma è

$$x + y = (\bar{x} + e_x) + (\bar{y} + e_y) = (\bar{x} + \bar{y}) + (e_x + e_y) \ .$$

Pertanto, per l'addizione, l'errore nel risultato è la somma degli errori degli addendi.

La propagazione dell'errore nella moltiplicazione è più complicata. Il prodotto è

$$xy = (\bar{x} + e_x)(\bar{y} + e_y) = \bar{x}\bar{y} + \bar{x}e_y + \bar{y}e_x + e_x e_y \ . \tag{1.4}$$

Pertanto, se $\bar{x}$ e $\bar{y}$ sono maggiori di 1 in valore assoluto, i termini $\bar{x}e_y$ e $\bar{y}e_x$ indicano che esiste la possibilità di amplificare gli errori iniziali e_x e e_y. Il fenomeno risulta più evidente se esaminiamo l'errore relativo. Riscrivendo la relazione (1.4) si ottiene

$$xy - \bar{x}\bar{y} = \bar{x}e_y + \bar{y}e_x + e_x e_y \ . \tag{1.5}$$

Supponiamo che $x \neq 0$ e $y \neq 0$, allora possiamo dividere la (1.5) per xy e ottenere

$$\begin{aligned} \frac{xy - \bar{x}\bar{y}}{xy} &= \frac{\bar{x}e_y + \bar{y}e_x + e_x e_y}{xy} \\ &= \frac{\bar{x}e_y}{xy} + \frac{\bar{y}e_x}{xy} + \frac{e_x e_y}{xy} \ . \end{aligned}$$

Inoltre, supponiamo che $\bar{x}/x \approx 1$, $\bar{y}/y \approx 1$, e $(e_x/x)(e_y/y) = \epsilon_x \epsilon_y \approx 0$. Allora facendo queste sostituzioni otteniamo la seguente relazione semplificata:

$$\frac{xy - \bar{x}\bar{y}}{xy} \approx \frac{e_y}{y} + \frac{e_x}{x} + 0 = \epsilon_y + \epsilon_x \ .$$

Questa relazione mostra che l'errore relativo sul prodotto xy è approsimativamente la somma degli errori relativi sulle approssimazioni $\bar{y}$ e $\bar{x}$.

Cancellazione numerica

La conseguenza più grave della rappresentazione con precisione finita dei numeri reali è senza dubbio il fenomeno della cancellazione numerica, ovvero la *perdita di cifre significative* dovuta ad operazioni di sottrazione quando il risultato è più piccolo di ciascuno dei due operandi; questo fenomeno si verifica quando i due operandi sono "quasi uguali". Per meglio illustrare quanto accade, supponiamo di avere due numeri floating-point $a = m_1 B^q$ e $b = m_2 B^q$, dove le mantisse m_1 e m_2, pur avendo più di t

cifre, sono rappresentabili solo con t cifre. Supponiamo inoltre che le prime tre cifre, per esempio, di m_1 coincidano con le corrispondenti di m_2. Nella mantissa $\bar{m}$ della differenza (normalizzata) $\bar{a} \ominus \bar{b} = \bar{m}B^{q-3}$, solamente le prime $t-3$ cifre provengono dalle mantisse m_1 e m_2; le restanti 3 (poste uguali a zero) non hanno alcun significato. Consideriamo il seguente esempio numerico:

$$B = 10,\ t = 6 \text{ e tecnica di arrotondamento } (\textit{round})$$
$$m_1 = .147554326, \qquad \bar{m}_1 = .147554$$
$$m_2 = .147251742, \qquad \bar{m}_2 = .147252.$$

La mantissa $\bar{m}$ della differenza di macchina risulta

$$\bar{m} = (\bar{m}_1 \ominus \bar{m}_2) \times 10^3 = .302000\ ,$$

mentre la vera mantissa è

$$m = (m_1 - m_2) \times 10^3 = .302584\ .$$

L'operazione di sottrazione in sè, anche quella di macchina, non introduce alcuna perdita di precisione; come abbiamo visto nel paragrafo precedente, le quattro operazioni aritmetiche di macchina possono provocare un errore relativo che non supera mai la precisione di macchina. La perdita di cifre significative descritta nell'esempio precedente (dove peraltro la sottrazione di macchina non introduce alcun errore) ha la sua origine negli errori presenti nei due operandi; o meglio, l'operazione di sottrazione ha amplificato detti errori. Se i due operandi sono privi di errori, il risultato non presenta alcuna perdita di precisione, e gli zeri finali aggiunti sono esatti.

Stabilità di un algoritmo

Spesso un errore iniziale si propaga in una sequenza di calcoli. Una qualità desiderata di ogni processo di calcolo numerico è che un piccolo errore iniziale produca piccole variazioni nel risultato finale. Un algoritmo che presenta questa proprietà si dice *stabile*; altrimenti, si dice *instabile*. Laddove è possibile saranno, ovviamente, preferiti metodi stabili. In altre parole, se si introduce un errore ad un certo passo (j) come si ripercuote al passo successivo $(j+1)$? Se l'errore cresce il procedimento sarà instabile viceversa sarà stabile. Occorrerà dunque verificare che

$$|\epsilon^{(j+1)}| < |\epsilon^{(j)}|$$

Osserviamo inoltre, che la stabilità può essere condizionata o incondizionata a seconda che dipenda da opportune condizioni sui parametri del problema/algoritmo, oppure no.

Quantitativamente, la propagazione dell'errore può essere descritta mediante la seguente definizione.

Definizione. Supponiamo che $E(n)$ rappresenti la crescita dell'errore dopo n passi. Se $|E(n)| \approx n\epsilon$, la crescita dell'errore si dice *lineare*. Se $|E(n)| \approx K^n\epsilon$, la crescita dell'errore si dice *esponenziale*. Se $K > 1$, l'errore esponenziale cresce indefinitamente per $n \to \infty$, e se $0 < K < 1$, l'errore esponenziale decresce a zero per $n \to \infty$.

1.2.4 Errori di troncamento

Gli errori di troncamento derivano dall'uso di un procedimento di approssimazione in sostituzione di operazioni matematiche esatte. Per comprendere le caratteristiche di tali errori, prendiamo in esame una formula matematica molto spesso utilizzata nei metodi numerici per esprimere una funzione in maniera approssimata: la serie di Taylor.

Serie di Taylor

Una serie di Taylor a infiniti termini può essere utilizzata per calcolare il valore di una funzione in corrispondenza di un dato valore della variabile indipendente x. Analogamente, la serie di Taylor permette di valutare il valore di una funzione nel punto x_{i+1} in termini del valore della funzione e delle sue derivate in un vicino punto x_i.

Invece di presentare immediatamente la serie di Taylor nella sua completezza, la costruiremo termine per termine allo scopo di comprenderne meglio il comportamento. Considerando solo il primo termine della serie, si ha

$$f(x_{i+1}) \simeq f(x_i) \tag{1.6}$$

Questa relazione, che viene detta *approssimazione di ordine zero*, indica che il valore di f nel nuovo punto x_{i+1} è lo stesso che aveva nel punto precedente x_i. Questo risultato ha senso, anche intuitivamente, in quanto, se x_i e x_{i+1} sono vicini, è probabile che i corrispondenti valori di f non differiscano di molto.

Ovviamente, se la funzione che viene approssimata è una costante, allora la (1.6) fornisce una stima esatta. Se, però, la funzione varia lungo tutto l'intervallo di interesse, sono necessari altri termini della serie di Taylor per ottenere una stima migliore. Per esempio, la *approssimazione del primo ordine* si ottiene aggiungendo il secondo termine alla serie:

$$f(x_{i+1}) \simeq f(x_i) + f'(x_i)(x_{i+1} - x_i) \tag{1.7}$$

Il termine del primo ordine è costituito da una pendenza $f'(x_i)$ moltiplicata per la distanza tra x_i e x_{i+1}; ora, quindi, l'espressione rappresenta una linea retta di pendenza variabile ed è in grado di approssimare un aumento o una diminuzione della funzione tra x_i e x_{i+1}.

Anche se la (1.7) è in grado di tener conto di una variazione, l'approssimazione che essa dà è valida solo se questa variazione è *lineare*, cioè se può essere rappresentata da una retta. Aggiungiamo alla serie, quindi, un altro termine, ottenendo l'approssimazione del *secondo ordine* che ci permetta, almeno in parte, di tener conto della eventuale curvatura della funzione:

$$f(x_{i+1}) \simeq f(x_i) + f'(x_i)(x_{i+1} - x_i) + \frac{f''(x_i)}{2!}(x_{i+1} - x_i)^2 \tag{1.8}$$

Continuando ad aggiungere i termini, si arriva ad ottenere lo sviluppo completo in serie di Taylor.

$$\begin{aligned} f(x_{i+1}) &= f(x_i) + f'(x_i)(x_{i+1} - x_i) + \frac{f''(x_i)}{2!}(x_{i+1} - x_i)^2 + \frac{f'''(x_i)}{3!}(x_{i+1} - x_i)^3 + \dots \\ &+ \frac{f^{(n)}(x_i)}{n!}(x_{i+1} - x_i)^n + R_n \end{aligned} \tag{1.9}$$

Siccome la (1.9) è una serie infinita, il segno di approssimazione usato dalla (1.6) alla (1.8) può essere sostituito da un segno di uguale. L'espressione contiene un termine *resto* R_n che tiene conto di tutti i termini da $n+1$ all'infinito:

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x_{i+1} - x_i)^{n+1} \tag{1.10}$$

dove n indica che il resto viene calcolato per l'approssimazione di ordine n-esimo e ξ è un valore compreso tra x_i e x_{i+1}.

Tale resto di R_n è il valore dell'errore di troncamento che si ha se la serie viene troncata all'n-esimo ordine.

Conviene spesso semplificare l'espressione della serie di Taylor introducendo il passo $h = x_{i+1} - x_i$ ed esprimendo la (1.9) come

$$f(x_{i+1}) = f(x_i) + f'(x_i)h + \frac{f''(x_i)}{2!}h^2 + \frac{f'''(x_i)}{3!}h^3 + \dots + \frac{f^{(n)}(x_i)}{n!}h^n + R_n \tag{1.11}$$

dove

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!}h^{n+1} \tag{1.12}$$

In generale, lo sviluppo in serie di Taylor di ordine n-esimo risulta esatto per un polinomio di ordine n. Nel caso di altre funzioni continue e derivabili, come la sinusoide o la funzione esponenziale, un numero finito di termini probabilmente non è in grado di dare una stima esatta. Ciascun nuovo termine contribuirà a migliorare, per quanto poco, l'approssimazione: la serie dà un risultato esatto solo se viene sommato un numero infinito di termini.

Nonostante quanto appena detto, lo sviluppo in serie di Taylor è utile in pratica in quanto, nella maggior parte dei casi, prendendo in considerazione solo pochi termini si ottiene un'approssimazione abbastanza vicina al valore reale per gli scopi pratici. La determinazione del numero di termini necessari per ottenere un'approssimazione "abbastanza buona" si basa sulla valutazione del resto dello sviluppo la cui forma generale, ricordiamo, è data dalla (1.12).

Questa relazione comporta principalmente due svantaggi: in primo luogo, ξ non è noto, esattamente ma si sa soltanto che si trova da qualche parte tra x_i e x_{i+1}. Inoltre,

per utilizzare la (1.12) bisogna determinare la derivata $(n+1)$-esima di $f(x)$ e, per far ciò, bisogna conoscere $f(x)$. Se conoscessimo $f(x)$, però, non avremmo alcun bisogno di sviluppare la funzione in serie di Taylor!

Nonostante questo dilemma, la (1.12) è ancora utile in quanto ci permette di approfondire la nostra conoscenza sugli errori di troncamento; ciò è possibile in quanto, in effetti, siamo in grado di controllare il termine h^{n+1} nell'equazione. In altre parole, possiamo decidere a che distanza da x_i vogliamo valutare $f(x)$ e quanti termini includere nello sviluppo. Di conseguenza, la (1.12) viene solitamente espressa come

$$R_n = O(h^{n+1})$$

nella quale la notazione $O(h^{n+1})$ significa che l'errore di troncamento è dell'ordine di h^{n+1}. L'errore, cioè, è proporzionale al passo h elevato alla $n+1$. Anche se questa approssimazione non dice nulla riguardo all'ordine di grandezza delle derivate che moltiplicano h^{n+1}, essa è estremamente utile per apprezzare l'errore relativo di metodi numerici basati sullo sviluppo in serie di Taylor. Per esempio, se l'errore è $O(h)$, un dimezzamento dell'ampiezza del passo dimezzerà anche l'errore, mentre, se l'errore è $O(h^2)$, un passo grande la metà dà un quarto dell'errore di partenza. Tutto ciò è vero solo se la derivata in ξ è quasi costante. In generale, possiamo contare sul fatto che aumentando il numero di termini della serie di Taylor l'errore di troncamento diminuisce. Inoltre, se h è sufficientemente piccolo, il primo termine più pochi altri costituiscono solitamente l'origine di una percentuale sproporzionatamente grande dell'errore; pertanto, sono sufficienti solo alcuni termini per ottenere una stima adeguata.

1.2.5 Errore numerico globale

L'errore numerico globale è la somma degli errori di troncamento e di arrotondamento. Spesso, l'unico modo di ridurre gli errori di arrotondamento consiste nell'aumentare il numero di cifre significative trattabili dal calcolatore. Inoltre, gli errori di arrotondamento tendono ad aumentare al crescere del numero di operazioni richieste da un'elaborazione. Ad esempio, le stime della derivata migliorano al diminuire del passo; dato che un passo più piccolo comporta un numero maggiore di operazioni, si ha che l'errore di troncamento decresce al crescere del numero di operazioni. In conclusione, abbiamo di fronte il seguente dilemma: l'espediente che permette di diminuire una componente dell'errore globale porta ad un aumento dell'altra componente. In generale, si tende a diminuire il passo per minimizzare gli errori di troncamento, per poi scoprire che, così facendo, l'errore di arrotondamento tende a dominare la soluzione e l'errore globale cresce: succede così che il rimedio è peggiore del male (Fig. 1.4).

Ciò di cui abbiamo bisogno è un criterio che ci permetta di scegliere un passo abbastanza ampio, tale da diminuire il numero di operazioni necessarie e da minimizzare gli errori di arrotondamento, ma piccolo abbastanza da evitare gli errori di troncamento. Se l'errore globale ha l'andamento visualizzato in Fig. 1.4, l'ideale sarebbe conoscere la posizione del punto nel quale gli errori di arrotondamento cominciano ad annullare i benefici di una riduzione del passo.

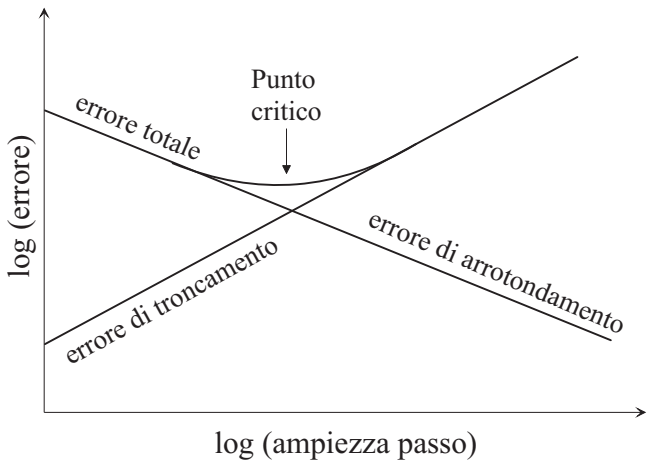


Figura 1.4: Rappresentazione grafica del compromesso tra l'errore di arrotondamento e l'errore di troncamento che talvolta si presenta utilizzando un dato metodo numerico. Viene indicato il punto di minimo, nel quale gli arrotondamenti cominciano ad annullare il beneficio ottenuto con la riduzione del passo.

Nei casi comuni, però, situazioni come quella descritta sono abbastanza rare, dato che in generale i calcolatori dispongono di un numero di cifre significative tale da annullare l'importanza degli errori di arrotondamento. Ciò nonostante, tali situazioni si presentano effettivamente, suggerendo una specie di "principio di indeterminazione numerica" che pone un limite invalicabile all'accuratezza ottenibile utilizzando certi metodi numerici su calcolatore.

A causa di questi limiti, la nostra capacità di valutare gli errori risulta limitata; come conseguenza, la valutazione degli errori nei metodi numerici diventa una specie di arte che dipende in parte dai suggerimenti ottenibili da prove pratiche e in parte dall'intuito e dall'esperienza dell'utilizzatore.

Bibliografia

G. Monegato, *Fondamenti di calcolo numerico*, Libreria Editrice Universitaria Levrotto&Bella, Torino, 1990

S.C. Chapra, R.P. Canale, *Metodi numerici per l'ingegneria*, McGraw-Hill Italia, 1988

W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, *Numerical recipes in C*, Cambridge University Press (2^a ed.), 1992

J.H. Mathews, *Numerical methods*, Prentice Hall (2^a ed.), 1992

O. Caligaris, P. Oliva, *Analis Matematica I*, ECIG Genova (2^a ed.), 1986

S. Ceri, D. Mandrioli, L. Sbattella, *Informatica - Arte e mestiere*, McGraw-Hill Italia, 1999

P. Tosoratti, *Introduzione all'informatica*, Casa Editrice Ambrosiana (2^a ed.), 1998

R. Bevilacqua, D.B.M. Capovani, O. Menchi, *Metodi numerici*, Zanichelli, 1992

Capitolo 2

Integrazione e derivazione di una funzione

2.1 Integrazione

Tema principale di questo capitolo è la valutazione numerica di integrali definiti

$$I(f) = \int_a^b f(x)dx \ .$$

Spesso è impossibile determinare $I(f)$ per via analitica; ma anche quando tale via risulta percorribile, l'espressione finale è sovente così complessa rispetto alla funzione integranda f da suggerire l'uso di approcci più semplici. Inoltre l'eventuale soluzione analitica potrebbe coinvolgere funzioni elementari e non, che devono poi venire valutate (e quindi approssimate). Se invece la funzione $f(x)$ è nota solo per punti, oppure è valutabile per ogni valore dell'argomento x mediante una *routine*, l'approccio analitico non può neppure essere preso in considerazione. Pertanto, supponendo di conoscere o di poter valutare la funzione integranda $f(x)$ nei punti $\{x_i\} \subseteq [a, b]$, prefissati oppure da noi scelti, esaminiamo alcuni procedimenti per il calcolo approssimato del suo integrale definito tra gli estremi a e b.

2.1.1 Integrale come somma

Intuitivamente il valore dell'integrale definito di una funzione può essere considerato come l'area sottesa dalla funzione stessa rispetto all'asse coordinato. E' possibile formulare un metodo approssimato che sfrutta direttamente questa proprietà.

Richiamiamo alcuni concetti preliminari sull'integrazione di funzioni.

Sia $f(x)$ definita in $[a, b]$ e ivi limitata. Chiamiamo partizione di $[a, b]$ un insieme $P = \{x_0, x_1, \ldots, x_{n-1}\}$ di punti di $[a, b]$ tali che

$$a = x_0 < x_1 < \cdots < x_{n-1} = b \ .$$

Se definiamo

$$m_i = \inf\{f(x), x \in [x_i, x_{i+1}]\} \ , \qquad M_i = \sup\{f(x), x \in [x_i, x_{i+1}]\}$$

allora

$$s(f,P)=\sum_{i=0}^{n-1} m_i(x_{i+1}-x_i) \leq \int_a^b f(x)dx \leq \sum_{i=0}^{n-1} M_i(x_{i+1}-x_i) = S(f,P)$$

dove $s(f,P)$ e $S(f,P)$ sono dette, rispettivamente, *somma inferiore* e *somma superiore* di f rispetto alla partizione P[1]. Mediante tali somme il valore dell'integrale è approssimato per difetto o per eccesso. In particolare, se viene scelta la partizione

$$P_n = \left\{ a + i\frac{(b-a)}{n-1}, i = 0,1,\ldots,n-1 \right\}$$

si ha che

$$\lim_{n\to\infty} s(f,P_n) = \sup\{s(f,P_n), n \in N\} = \int_a^b f(x)dx = \inf\{S(f,P_n), n \in N\} = \lim_{n\to\infty} S(f,P_n)$$

Pertanto, possiamo direttamente utilizzare le definizioni di $s(f,P)$ e $S(f,P)$ per una valutazione approssimata dell'integrale I, inoltre per $n \to \infty$ l'approssimazione tende al valore esatto assicurandoci la convergenza del procedimento (v. Fig. 2.1).

2.1.2 Metodo di calcolo dell'integrale per somme di rettangoli

Consideriamo n sottointervalli di $[a,b]$ di ampiezza $h = \frac{(b-a)}{n-1}$, dove h è detto *passo di integrazione*. In queste ipotesi $[x_i, x_{i+1}] = [x_i, x_i+h]$. Possiamo applicare direttamente le definizioni di somme inferiori e superiori e approssimare così il valore dell'integrale I come media aritmetica delle sue approssimazioni attraverso le somme inferiori e superiori:

```
h=(b-a)/((float) (n-1));
x=a;
y=f(a);
sum=0.;
SUM=0.;
for(i=0;i<n;i++)
 {
   yprec=y;
   x+=h;
   y=f(x);
   sum+=h*min(y,yprec);
   SUM+=h*max(y,yprec);
 }
I_inf=sum;
```

[1]Si suppone di conoscere i valori della $f(x)$ solo nei punti della partizione

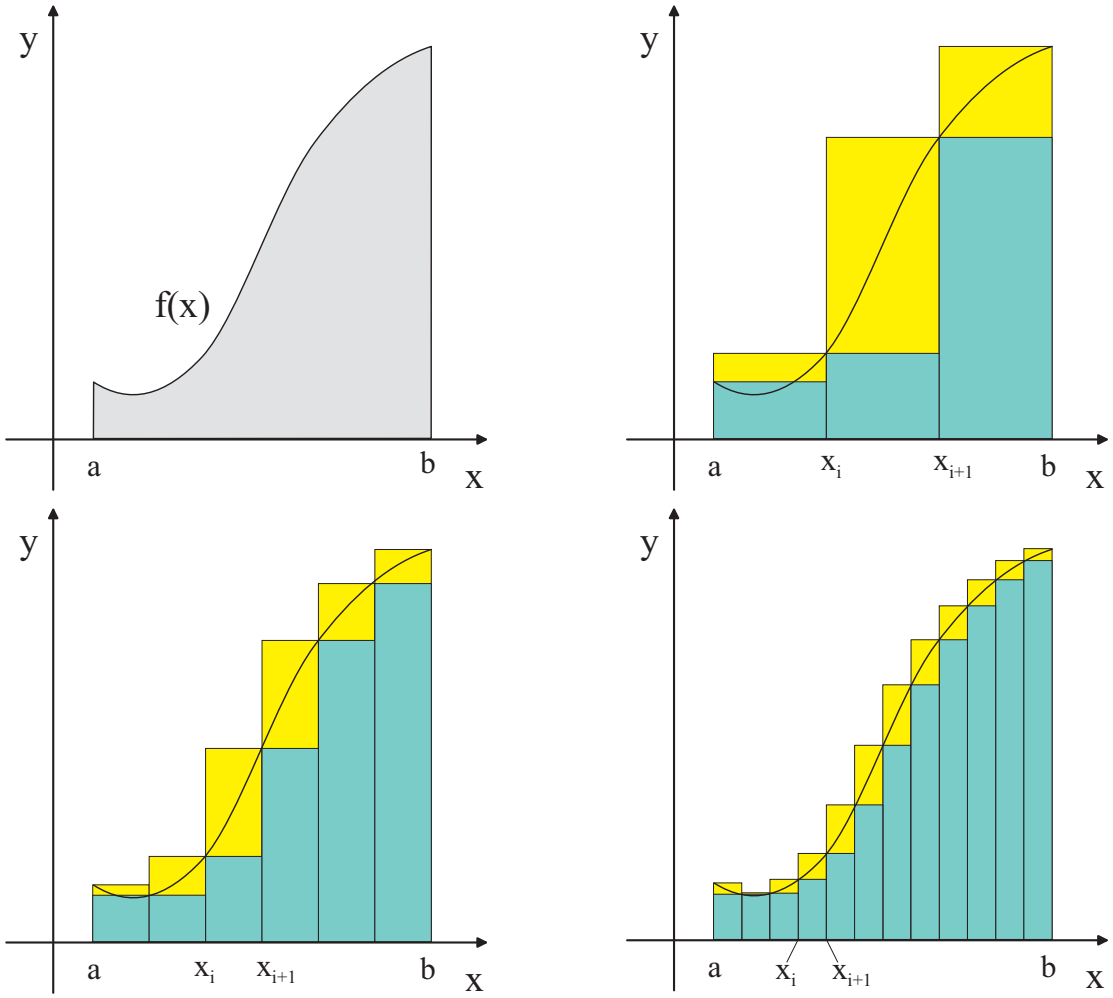


Figura 2.1: Rappresentazione grafica del calcolo approssimato di un integrale definito mediante somme superiori e somme inferiori, per un numero crescente dei punti della partizione

```
I_sup=SUM;
I=0.5*(I_inf+I_sup);
```

dove `sum`$\rightarrow s(f,P_n)$ e `SUM`$\rightarrow S(f,P_n)$ e dove

$$\min(p,q)=\begin{cases} p & \text{se } p<q \\ q & \text{altrimenti} \end{cases} \qquad \max(p,q)=\begin{cases} p & \text{se } p\geq q \\ q & \text{altrimenti} \end{cases}$$

Considerato che, al crescere di n, sia le somme inferiori sia quelle superiori convergono al valore esatto dell'integrale, le prime attraverso approssimazioni per difetto, le seconde attraverso approssimazioni per eccesso, è anche possibile costruire una formula approssimata (*formula di Eulero*) che evita di valutare la condizione sul valore massimo e minimo della funzione rispetto agli estremi del sottointervallo $\{x_i, x_i+h\}$

$$I_h(f)=\sum_{i=0}^{n-1} f(x_i)(x_{i+1}-x_i)=h\sum_{i=0}^{n-1} f(x_i) \tag{2.1}$$

Il codice per il calcolo dell'integrale può quindi essere semplificato nel modo seguente:

```
h=(b-a)/((float) (n-1));
x=a;
I=0.;
for(i=0;i<n;i++)
 {
   y=f(x);
   I+=h*y;
   x+=h;
 }
```

2.1.3 Metodi interpolatori

I metodi di integrazione numerica sin qui presentati sfruttano direttamente la definizione di integrabilità di una funzione nell'intervallo $[a,b]$. Più in generale, il calcolo numerico fornisce numerose **formule**, dette **di quadratura**, del tipo

$$Q_n(f)=\sum_{i=0}^{n-1} w_i f(x_i) \tag{2.2}$$

tali che

$$I(f)=\int_a^b f(x)dx \approx Q_n(f)\ .$$

I numeri (reali) $\{x_i\}\subseteq[a,b]$ e $\{w_i\}$ vengono chiamati rispettivamente *nodi* e *pesi* della formula di quadratura. L'idea alla base dei metodi interpolatori è quella di scegliere i pesi della formula di quadratura sulla base di un'interpolazione dei valori della f con

funzioni elementari. In particolare, la funzione integranda $f(x)$ viene approssimata con un polinomio di $L_n(f;x)$, di grado $k \leq n-1$, unico, che interpola la funzione nei nodi $\{x_i\}$. I coefficienti (pesi) nella formula di quadratura sono ricavati calcolando analiticamente il valore dell'integrale del polinomio interpolatore che passa per $k+1$ punti della $f(x)$. Le formule costruite in questo modo vengono chiamate interpolatorie e risultano "esatte" ogniqualvolta $f(x)$ è un polinomio di grado minore o uguale a k.

Affinchè la formula di quadratura (2.2) definisca una "buona" discretizzazione dell'integrale è necessario che

$$\lim_{n\to\infty} \sum_{i=0}^{n-1} w_i f(x_i) = \int_a^b f(x)dx$$

in questo caso diciamo che la formula è convergente. Più regolare è la funzione $f(x)$ più rapida è la convergenza della quadratura al valore esatto dell'integrale.

Si può osservare che la formula di Eulero, detta anche formula di quadratura rettangolare, presentata nel paragrafo precedente, si ottiene quando il grado del polinomio $k = 0$; le formule derivanti da polinomi interpolatori di grado $k > 0$ consentono, in generale, di ottenere, a parità di tempo di calcolo, precisioni migliori; in particolare, largamente utilizzate sono la *formula dei trapezi* o di Bezout, ottenuta per $k = 1$ e quella di *Simpson*, ottenuta per $k = 2$.

Formula di Eulero (k=0) Detto f_0 il valore di $f(x)$ nel punto x_0, la formula di Eulero deriva dall'integrazione della costante f_0 nell'intervallo $[x_0, x_0 + h]$

$$\int_{x_0}^{x_0+h} f(x)dx \approx hf_0 \tag{2.3}$$

Estendendo la (2.3) a tutto l'intervallo $[a, b]$ si ottiene:

$$\int_{x_0}^{x_{n-1}} f(x)dx \approx Q_n(f) = \ h(f_0 + f_1 + \cdots + f_{n-1}) \tag{2.4}$$

Formula dei trapezi (k=1) Detti f_0 e f_1 i valori di $f(x)$ nei punti x_0 e $x_0 + h$, la formula dei trapezi deriva dall'integrazione della retta che passa per i punti f_0 e f_1

$$\int_{x_0}^{x_0+h} f(x)dx \approx \frac{h}{2}(f_0 + f_1) \tag{2.5}$$

Estendendo la (2.5) a tutto l'intervallo $[a, b]$:

$$\int_{x_0}^{x_{n-1}} f(x)dx \approx Q_n(f) = h\left(\frac{1}{2}f_0 + f_1 + f_2 + \cdots + f_{n-2} + \frac{1}{2}f_{n-1}\right) \tag{2.6}$$

Formula di Simpson (k=2) Detti f_0, f_1 e f_2 i valori di $f(x)$ nei punti x_0, x_0+h e x_0+2h, la formula di Simpson deriva dall'integrazione della parabola che passa per i punti f_0, f_1, f_2:

$$\int_{x_0}^{x_0+2h} f(x)dx \approx \frac{h}{3}(f_0+4f_1+f_2) \tag{2.7}$$

Estendendo la formula all'intero intervallo $[a,b]$ si ottiene:

$$\int_{x_0}^{x_{n-1}} f(x)dx \approx Q_n(f) = \frac{h}{3}\left(f_0+4f_1+2f_2+4f_3+\cdots+2f_{n-3}+4f_{n-2}+f_{n-1}\right) \tag{2.8}$$

2.1.4 Stima dell'errore

Supponiamo di aver scelto una formula di quadratura

$$Q_n(f) = \sum_{i=0}^{n} w_i f(x_i)$$

per approssimare l'integrale

$$I(f) = \int_a^b f(x)dx\ .$$

Affinchè la stima $Q_n(f)$ sia credibile occorre poter associare ad essa un'indicazione della sua precisione, valutando lo scarto

$$R_n(f) = I(f) - Q_n(f)$$

Per le formule di tipo interpolatorio si può dare una rappresentazione integrale dell'errore:

$$R_n(f) = \int_a^b E_n(f;x)dx \qquad \text{dove} \qquad E_n(f;x) = f(x) - L_n(f;x)$$

altre sono reperibili nella letteratura specializzata. Tuttavia, queste espressioni dell'errore hanno un'importanza soprattutto teorica. Solitamente, la stima dell'errore $R_n(f)$ viene prodotta nel modo seguente: si prende una seconda formula $Q_m(f)$ dello stesso tipo di $Q_n(f)$, ma più "precisa", ovvero con un numero maggiore di nodi $(m>n)$, e si pone

$$|R_n(f)| \approx |Q_n(f) - Q_m(f)|\ .$$

Generalmente si sceglie $m=n+1$ quando la funzione $f(x)$ è ritenuta sufficientemente "regolare", e $m=2n$ quando non lo è. In pratica, allo scopo di eseguire il calcolo dell'integrale $I(f)$ con precisione ε preassegnata si può iterativamente valutare l'integrale mediante una formula di quadratura con valori decrescenti del passo di integrazione h. Ad esempio, dimezzando iterativamente h (e quindi raddoppiando i nodi, vedi Fig. 2.2) e arrestando il procedimento di calcolo quando risulta

$$|Q_n(f) - Q_{2n}(f)| < \varepsilon$$

Si osserva che per quanto detto sulla precisione delle operazioni di macchina, non potremo considerare passi di intregrazione $h < \varepsilon_m$.

E' inoltre immediato verificare che i metodi di integrazione numerica qui presentati risultano stabili e ben condizionati.

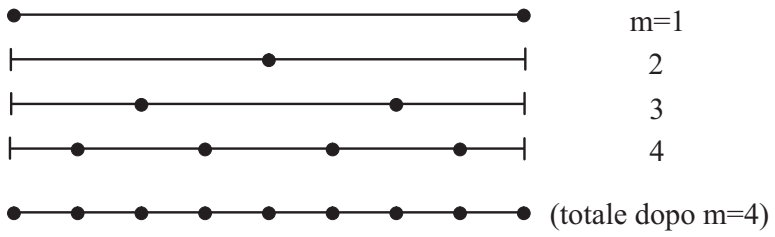


Figura 2.2: Rappresentazione schematica dei punti dell'intervallo ove occorre valutare la funzione considerando suddivisioni successive del passo di integrazione. Si osservi che ad ogni suddivisione rimangono validi i punti calcolati precedentemente e devono essere calcolati solo i nuovi punti

2.2 Derivazione

2.2.1 Differenziali numerici

In parole molto povere si può dire che una funzione $f : (a, b) \to R$ è continua in un punto $x_0 \in (a, b)$ se, quando x si discosta di poco da x_0, $f(x)$ è poco distante da $f(x_0)$. Se si vuole valutare come varia la quantità

$$\frac{f(x) - f(x_0)}{x - x_0} \tag{2.9}$$

(coefficiente angolare della corda che passa per i punti $(x_0, f(x_0))$ e $(x, f(x))$ vicino al punto x_0 è naturale considerare il suo limite per $x \to x_0$. In particolare la quantità (2.9) è detta rapporto incrementale relativo alla funzione f nel punto x_0 e diciamo che f è derivabile in x_0 se

$$\lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0} \qquad \text{esiste finito.}$$

In tal caso chiamiamo il suo valore *derivata* di f in x_0 e scriviamo

$$f'(x_0) \qquad \text{oppure} \qquad \frac{df}{dx}(x_0) \ .$$

La derivata di f fornisce, vicino a x_0, una valutazione del modo in cui $f(x) - f(x_0)$ varia al variare di $x - x_0$. In altre parole, vicino a x_0, la quantità $f(x) - f(x_0)$ è approssimabile, in qualche senso, mediante la quantità $f'(x_0)(x - x_0)$ che è ovviamente di ben più facile trattazione. Tale quantità rappresenta geometricamente una retta

tangente alla funzione f nel punto x_0 (che rappresenta l'approssimazione del primo ordine della funzione in x_0). Allo stesso modo si possono definire le derivate di ordine superiore.

Il calcolo della derivata puntuale di una funzione può quindi essere approssimato dal suo rapporto incrementale. Per valutare l'errore che si commette con tale approssimazione vediamo prima l'errore su un singolo passo $h = (x - x_0)$ e poi estenderemo a l'errore sull'intero intervallo. Sia $f \in C^2$

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2}h^2 f''(\xi) \qquad \xi \in [x_0, x_0 + h]$$

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{1}{2}hf''(\xi)$$

Poiché l'errore sul singolo passo h è una $O(h)$ e il numero di sottointervalli è $n = (b - a)/h$, l'errore complessivo sarà $O(1)$, ciò vuol dire che se $h \to 0$ l'errore risulta finito. Siamo pertanto in una situazione pessimistica in cui l'errore di troncamento sull'intero intervallo non viene ridotto riducendo il passo.

E' facile verificare che si giunge a conclusioni simili considerando il rapporto incrementale sinistro.

Supponiamo che $f \in C^3$ e scriviamo le approssimazioni di Taylor per $f(x_0 + h)$ e $f(x_0 - h)$:

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f'''(\xi)h^3 \qquad \xi \in [x_0, x_0 + h]$$

$$f(x_0 - h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f'''(\eta)h^3 \qquad \eta \in [x_0 - h, x_0]$$

dalla differenza delle due espressioni si ottiene:

$$f(x_0 + h) - f(x_0 - h) = 2f'(x_0)h + \frac{1}{6}\left[f'''(\xi) + f'''(\eta)\right]h^3$$

Sapendo che per qualunque funzione continua in un intervallo la media di due suoi valori è uguale al valore della funzione in un punto intermedio.

$$f(x_0 + h) - f(x_0 - h) = 2f'(x_0)h + \frac{1}{3}f'''(\zeta)h^3 \qquad \zeta \in [x_0 - h, x_0 + h]$$

da cui si ottiene

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} - \frac{1}{6}f'''(\zeta)h^2$$

che rappresenta il rapporto incrementale centrato.

Approssimando la derivata con questo rapporto incrementale l'errore sul singolo passo va con h^2 e l'errore complessivo, sull'intero intervallo, sarà $O(h)$ (tenderà a zero con h). L'errore che si commette approssimando la derivata con il rapporto incrementale centrato va a zero più rapidamente di quello associato ad un rapporto incrementale monolaterale (destro o sinistro), assicurando complessivamente un errore

più piccolo. Diminuendo il passo siamo sicuri che l'errore diminuisce, cosa che non era certa nel caso di differenze non centrate.

L'espressione approssimata per una derivata seconda si può ricavare con procedimento analogo sommando gli sviluppi di Taylor al quarto ordine della funzione f e ottenendo

$$f(x_0+h)+f(x_0-h)=2f(x_0)+f''(x_0)h^2+\frac{1}{24}\left[f^{iv}(\xi)+f^{iv}(\eta)\right]h^4$$

Questa espressione mi permette di calcolare una forma approssimata per la derivata seconda della funzione e di valutare l'errore ad essa associato:

$$f''(x_0)=\frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2}-\frac{1}{12}f^{iv}(\zeta)h^2$$

Anche in questo caso l'errore che commetto va a zero con h^2, e l'errore globale andrà a zero con ordine h.

Un risultato analogo si ottiene ricavando l'espressione approssimata della derivata seconda come rapporto incrementale delle derivate prime:

$$f''(x_0)=\frac{f'(x_0)_+-f'(x_0)_-}{h}=\frac{[f(x_0+h)-f(x_0)]-[f(x_0)-f(x_0-h)]}{h^2}$$

2.2.2 Derivazione numerica

L'approssimazione della derivata di una funzione in un punto mediante il suo rapporto incrementale suggerisce direttamente l'algoritmo:

1. scegliere un valore piccolo per il passo h
2. valutare $f(x+h)$
3. valutare $f(x)$ nel caso non fosse già disponibile tale valore
4. applicare la relazione
$$f'(x)\sim\frac{f(x+h)-f(x)}{h} \tag{2.10}$$

Un'applicazione indiscriminata di tale procedimento rischia tuttavia di portare a risultati non accurati. In particolare, un aspetto critico è rappresentato dalla scelta di h.

In generale l'approssimazione della derivata mediante differenziali numerici contiene due sorgenti di errore: (1) errori di troncamento e (2) errori di arrotondamento:

$$e=e_r+e_t$$

per ulteriori dettagli si veda, ad esempio, Press *et al*, 1992.

Bibliografia

W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, *Numerical recipes in C*, Cambridge University Press (2^a ed.), 1992

G. Monegato, *Fondamenti di calcolo numerico*, Libreria Editrice Universitaria Levrotto&Bella, Torino, 1990

O. Caligaris, P. Oliva, *Analisi Matematica I*, ECIG Genova (2^a ed.), 1986

B. Fadini, C. Savy, *Fondamenti di Informatica - fondamenti di programmazione*, Liguori editore, 1991