Introduzione. Rappresentazione di numeri in macchina, condizionamento e stabilità

Stefano Berrone

Dipartimento di Matematica
tel. 011 0907503
stefano.berrone@polito.it
http://calvino.polito.it/~sberrone

Laboratorio di modellazione e progettazione materiali

Il sistema floating point

Fissata la base β di un sistema di numerazione, la rappresentazione **floating point** (virgola mobile) di un numero reale consiste nell'usare una opportuna potenza di β in modo da non avere né parte intera e né zeri dopo la virgola.
Ad esempio, in base $\beta = 10$:

$$123.4567 \Rightarrow 0.1234567 \cdot 10^3$$

$$0.00789 \Rightarrow 0.789 \cdot 10^{-2}$$

$$0.6 \Rightarrow 0.6 \cdot 10^0$$

Osservazione

Si osservi che

$$0.789 = 7 \cdot 10^{-1} + 8 \cdot 10^{-2} + 9 \cdot 10^{-3}$$

Il sistema floating point

Si definisce insieme dei numeri macchina (floating-point) con t cifre di mantissa, base β e range (L, U), l'insieme dei numeri reali definito nel modo seguente:

$$\mathbb{F}(\beta, t, L, U) = \{0\} \cup \left\{ x \in \mathbb{R} \ : \ x = (-1)^s \beta^e \sum_{i=1}^{t} d_i \beta^{-i} \right\}$$

- t, β sono interi positivi con $\beta \geq 2$
- $0 \leq d_i \leq \beta - 1$, $i = 1, .., t$,
- $d_1 \neq 0$ (rappresentazione normalizzata)
- $L \leq e \leq U$, U usualmente positivo e L negativo.

L'esponente e viene chiamato **caratteristica** della rappresentazione floating-point del numero, mentre la quantità $\sum_{i=1}^{t} d_i \beta^{-i}$ si chiama **mantissa** e varia tra β^{-1} e $1 - \beta^{-t}$.

Per ogni numero $x \in \mathbb{F}(\beta, t, L, U)$ si ha

$$x_{min} = \beta^{L-1} \leq |x| \leq \beta^{U}\left(1 - \beta^{-t}\right) = x_{MAX}.$$

Dalla relazione precedente di deduce che non è possibile rappresentare alcun numero (a parte lo zero) minore in valore assoluto di x_{min}.
Per aggirare questa limitazione lo standard IEEE754 prevede una rappresentazione **denormalizzata**.
Quando l'esponente e è pari al valore minimo L la condizione $d_1 \neq 0$ può essere abbandonata e quindi vengono accettate mantisse comprese tra β^{-t} e $\beta^{-1} - \beta^{-t}$.

Approssimazione di un numero reale con un numero di macchina

Sia x un numero reale e sia $\bar{x}$ una sua approssimazione:

$$x = (-1)^s m \cdot \beta^e, \qquad \bar{x} = (-1)^s \bar{m} \cdot \beta^e$$

Errore assoluto:

$$E_a \equiv |\bar{x} - x|$$

Errore relativo:

$$E_r \equiv \frac{|\bar{x} - x|}{|x|}, \ x \neq 0$$

Tecniche di approssimazione

In un elaboratore elettronico non possono essere rappresentati tutti i numeri reali, quindi ogni numero reale $x = \mathrm{sign}(x)\, m\, \beta^e$ la cui caratteristica e cada nel range $[L, U]$ sarà approssimato con un numero $\bar{x} \in \mathbb{F}(\beta, t, L, U)$.
La mantissa $\bar{m}$ di tale approssimazione dovrà necessariamente avere al più t cifre.

Le tecniche di *approssimazione a numeri macchina* previste dallo standard IEEE 754 sono:

1. **Round to nearest** (Arrotondamento) x viene approssimato con il numero rappresentabile *più vicino*. Se x cade esattamente a metà tra due numeri rappresentabili, viene approssimato con quello dei due con la cifra meno significativa della mantissa pari (se $\beta = 2$, quello con bit meno significativo uguale a 0). Garantisce la migliore stabilità numerica.
2. **Round toward zero** (Troncamento): x viene approssimato con il *più grande* numero rappresentabile il cui *valore assoluto* è *minore* di quello del risultato.
3. **Round toward plus Infinity**: x viene approssimato al *più piccolo* numero rappresentabile *maggiore* del risultato.
4. **Round toward minus Infinity**: x viene approssimato al *più grande* numero rappresentabile *minore* del risultato.

Errori commessi

Osservazione

Le mantisse $\bar{m}$ *dei numeri macchina sono separate di passi uniformi pari a* β^{-t}. *Infatti due mantisse consecutive hanno la forma*

$$0.d_1 d_2 d_3 \ldots d_{t-1} d_t$$

$$0.d_1 d_2 d_3 \ldots d_{t-1} (d_t + 1)$$

Quindi i numeri floating-point normalizzati sono, per un fissato valore della caratteristica, equispaziati.

Errori commessi

- **Troncamento**: tutte le mantisse $m \in [m_1, m_1 + \beta^{-t})$ vengono approssimate con $\bar{m} = m_1$. L'errore commesso è quindi

$$m - m_1 < \beta^{-t},$$

sempre positivo.

- **Arrotondamento**: tutte le mantisse che cadono in $(m_1 - \frac{1}{2}\beta^{-t}, m_1 + \frac{1}{2}\beta^{-t}]$ vengono approssimate con m_1. In questo modo l'errore sulla mantissa risulta

$$|m - m_1| \leq \frac{1}{2}\beta^{-t}$$

e può essere sia positivo che negativo.

Tra questi metodi l'**arrotondamento** è sicuramente **più oneroso** dal punto di vista di istruzioni del microprocessore, ma provoca un'errore metà degli altri ed è sicuramente **più conveniente** dal punto di vista dell'accuratezza.
Riassumendo gli errori di approssimazione **sulle mantisse** risultano:

$$\begin{aligned} |\bar{m} - m| &< \beta^{-t}, \quad \text{troncamento,} \\ |\bar{m} - m| &\leq \frac{1}{2}\beta^{-t}, \quad \text{arrotondamento} \end{aligned}$$

L'errore assoluto associato all'approssimazione $\bar{x}$ vale:

$$\begin{aligned} |\bar{x} - x| &< \beta^{e-t}, \quad \text{troncamento,} \\ |\bar{x} - x| &\leq \frac{1}{2}\beta^{e-t}, \quad \text{arrotondamento,} \end{aligned}$$

Poiché $m \geq 0.1000\ldots = \beta^{-1}$, si ha

$$|x| = m\beta^e \geq \beta^{-1}\beta^e$$

e quindi per l'errore relativo

$$\begin{aligned} \frac{|\bar{x} - x|}{|x|} \leq \frac{|\bar{x} - x|}{\beta^{e-1}} &< \varepsilon_{\mathrm{m}} \equiv \beta^{1-t}, \quad \text{troncamento,} \\ \frac{|\bar{x} - x|}{|x|} \leq \frac{|\bar{x} - x|}{\beta^{e-1}} &\leq \varepsilon_{\mathrm{m}} \equiv \frac{1}{2}\beta^{1-t}, \quad \text{arrotondamento.} \end{aligned}$$

La quantità $\texttt{eps} \equiv \beta^{1-t}$ verrà chiamata **epsilon di macchina**. Con il simbolo ε_{m} (o ε_{m}) si indicherà invece la **precisione di macchina**. Essa è una costante caratteristica di ogni aritmetica floating-point e rappresenta **la massima precisione relativa di calcolo raggiungibile sul calcolatore** e con il tipo di dati che la implementano.
In pratica due quantità la cui differenza **relativa** è minore della precisione di macchina, sono da considerarsi indistinguibili per il calcolatore.
Non ha senso cercare di determinare approssimazioni con precisione relativa inferiore alla quantità ε_{m}.

Standard IEEE

I personal computer che implementano lo Standard IEEE 754-1985 prevedono:

1. 32 bit per la *singola precisione*, 1+8+23
2. 64 bit per la *doppia precisione*, 1+11+52

Conseguenze:

1. In base 2, $L = -127$, $U = 128$, $\texttt{eps} = 2^{-22}$;
   in base 10, $L \simeq -38$, $U \simeq 38$, $\texttt{eps} \simeq 10^{-7}$
2. In base 2, $L = -1023$, $U = 1024$, $\texttt{eps} \simeq 2^{-52}$;
   in base 10, $L \simeq -308$, $U \simeq 308$, $\texttt{eps} \simeq 10^{-16}$

Overflow e Underflow

1. **Overflow**: errore dovuto al tentativo di rappresentare numeri con $e > U$
2. **Underflow**: errore dovuto al tentativo di rappresentare numeri con $e < L$

Operazioni di macchina effettuate in virgola mobile

Vediamo adesso cosa accade quando si effettuano delle operazioni aritmetiche in macchina.
Il risultato di una operazione aritmetica eseguita tra due numeri macchina non è, in generale, un numero macchina, quindi anche questo risultato dovrà essere approssimato.

Definizione

Chiameremo **operazione di macchina** *il risultato dell'operazione eseguita sui numeri macchina seguita da un'approssimazione.*

Indichiamo con $\mathrm{fl}(x)$ l'operazione di approssimazione di x a numero di macchina in aritmetica floating-point e con $\oplus$, $\ominus$, $\otimes$, $\oslash$ le operazioni macchina corrispondenti a $+$, $-$, $\times$, $/$.
Quindi, ad esempio,

$$a \oplus b := \mathrm{fl}(\mathrm{fl}(a) + \mathrm{fl}(b))$$

Sia δ l'errore relativo della rappresentazione di x:

$$\delta = \frac{\mathrm{fl}(x) - x}{x} \qquad \Rightarrow \qquad \mathrm{fl}(x) = x(1+\delta)$$

Ricordiamo che si ha, dalla definizione di ε_{m}, $|\delta| \leq \varepsilon_{\mathrm{m}}$
Si ha allora

$$\mathrm{fl}(x) = x(1+\delta), \quad |\delta| \leq \varepsilon_{\mathrm{m}}.$$

Questo è il modo in cui viene generalmente descritto il legame tra un numero (non di macchina) x e la sua rappresentazione $\mathrm{fl}(x)$.

Per le operazioni di macchina si ha

$$
\begin{aligned}
a \oplus b &= \mathrm{fl}(\mathrm{fl}(a) + \mathrm{fl}(b)) = (\mathrm{fl}(a) + \mathrm{fl}(b))(1+\delta_1), \quad |\delta_1| \leq \varepsilon_{\mathrm{m}} \\
a \ominus b &= \mathrm{fl}(\mathrm{fl}(a) - \mathrm{fl}(b)) = (\mathrm{fl}(a) - \mathrm{fl}(b))(1+\delta_2), \quad |\delta_2| \leq \varepsilon_{\mathrm{m}} \\
a \otimes b &= \mathrm{fl}(\mathrm{fl}(a) \times \mathrm{fl}(b)) = (\mathrm{fl}(a) \times \mathrm{fl}(b))(1+\delta_3), \quad |\delta_3| \leq \varepsilon_{\mathrm{m}} \\
a \oslash b &= \mathrm{fl}(\mathrm{fl}(a) \,/\, \mathrm{fl}(b)) = (\mathrm{fl}(a) \,/\, \mathrm{fl}(b))(1+\delta_4), \quad |\delta_4| \leq \varepsilon_{\mathrm{m}}
\end{aligned}
$$

Osservazione

Queste relazioni mostrano che, prescindendo dagli eventuali errori di macchina presenti nelle rappresentazioni degli operandi, l'errore relativo commesso eseguendo operazioni di macchina non è mai superiore alla precisione di macchina.

Osservazione

Non tutte le proprietà delle operazioni aritmetiche si conservano per le operazioni di macchina. La proprietà commutativa per somma e prodotto si conserva anche per le operazioni di macchina:

$$a \oplus b = b \oplus a, \qquad a \otimes b = b \otimes a,$$

Ma non valgono più le seguenti proprietà:

$$\begin{aligned}
a \oplus (b \oplus c) &\neq (a \oplus b) \oplus c, \\
a \otimes (b \otimes c) &\neq (a \otimes b) \otimes c, \\
a \otimes (b \oplus c) &\neq (a \otimes b) \oplus (a \otimes c), \\
(a \otimes b) \oslash b &\neq a, \\
(a \oslash b) \otimes b &\neq a, \\
(a \otimes b) \oslash c &\neq (a \oslash c) \otimes b.
\end{aligned}$$

Osservazione

Inoltre per le operazioni di macchina può accadere che valga la relazione

$$a \oplus b = \mathrm{fl}(a), \qquad 0 < |\mathrm{fl}(b)| \ll |\mathrm{fl}(a)|.$$

Due espressioni equivalenti in aritmetica infinta possono non esserlo in aritmetica finita.

Definizione

Diremo **equivalenti** *due espressioni che in aritmetica finita forniscano risultati la cui distanza relativa differisce di una quantità dell'ordine della precisione di macchina.*

Cancellazione numerica

Il fenomeno della **cancellazione numerica** è la conseguenza più grave dell'aritmetica finita.
Essa **può** verificarsi quando si esegue la sottrazione di due numeri di macchina molto vicini tra loro.

Definizione

Si dice **cancellazione numerica** *il fenomeno di* perdita di cifre significative *che si verifica quando si opera una* sottrazione *tra due numeri di macchina "quasi uguali" tra loro (ovvero, il risultato è più piccolo di ciascuno dei due operandi)*

Sostanzialmente consiste in una **enorme amplificazione degli errori** di approssimazione sugli operandi.

Esempio (Cancellazione 1)

Si considerino i seguenti numeri in formato floating-point normalizzato

$$x_1 = 0.19101972 \cdot 10^3, \qquad x_2 = 0.19101708 \cdot 10^3$$

e si voglia eseguire l'operazione $x_1 \ominus x_2$ *in un'aritmetica in base* $\beta = 10$, 6 *cifre di mantissa, operante con troncamento* ($\varepsilon_{\mathrm{m}} = 10^{-5}$)

$$\mathrm{fl}(x_1) = 0.191019 \cdot 10^3, \qquad \mathrm{fl}(x_2) = 0.191017 \cdot 10^3.$$

Che errore abbiamo commesso per ora?

$$|\mathrm{fl}(x_1) - x_1| = 0.720000 \cdot 10^{-3}$$

$$\frac{|\mathrm{fl}(x_1) - x_1|}{|x_1|} = \frac{0.720000 \cdot 10^{-3}}{0.19101972 \cdot 10^3} \simeq 0.37692 \cdot 10^{-5} < \varepsilon_{\mathrm{m}}$$

Esempio (segue)

$$\frac{|\mathrm{fl}(x_2) - x_2|}{|x_2|} = \frac{0.800000 \cdot 10^{-4}}{0.19101708 \cdot 10^{3}} \simeq 0.41881 \cdot 10^{-6} < \varepsilon_{\mathrm{m}}$$

Calcoliamo la differenza in macchina:

$$\mathrm{fl}(x_1) \ominus \mathrm{fl}(x_2) = 0.000002 \cdot 10^{3} = 0.200000 \cdot 10^{-2}$$

Errore commesso? Differenza esatta:

$$x_1 - x_2 = 0.264000 \cdot 10^{-2}$$

Errore relativo:

$$\frac{|(\mathrm{fl}(x_1) \ominus \mathrm{fl}(x_2)) - (x_1 - x_2)|}{|(x_1 - x_2)|} = 0.2424,$$

Esempio (segue)

Cos'è accaduto? Approssimando le mantisse di x_1 e x_2 si sono "buttate via" le cifre che seguono la sesta (con un errore di approssimazione entro i limiti della precisione di macchina).La differenza effettuata tra x_1 e x_2 ha però amplificato molto la perdita di informazione dovuta all'approssimazione, fino a farla risalire alla prima cifra significativa del risultato!

$$\mathrm{fl}(x_1) = 0.191019 \cdot 10^3, \quad \mathrm{fl}(x_2) = 0.191017 \cdot 10^3$$

$$\Downarrow$$

$$\mathrm{fl}(x_1) \ominus \mathrm{fl}(x_2) = 0.000002 \cdot 10^3 = 0.200000 \cdot 10^{-2}.$$

Esempio (segue)

Con arrotondamento?

$$\mathrm{fl}(x_1) = 0.191020 \cdot 10^3, \qquad \mathrm{fl}(x_2) = 0.191017 \cdot 10^3$$

$$\mathrm{fl}(x_1) \ominus \mathrm{fl}(x_2) = 0.000003 \cdot 10^3 = 0.300000 \cdot 10^{-2}$$

$$\frac{|(\mathrm{fl}(x_1) \ominus \mathrm{fl}(x_2)) - (x_1 - x_2)|}{|(x_1 - x_2)|} = 0.1363$$

Osservazione

Nell'esempio precedente la sottrazione di macchina non introduce alcun errore di approssimazione, ma fornisce il risultato esatto. Quindi **la sottrazione non genera problemi in sé**, *ma* **amplifica** *errori di approssimazione già esistenti sugli operandi.*

É possibile evitare la cancellazione numerica? Una possibilitá consiste nel l'usare (quando possibile) forme alternative per il calcolo di una espressione.

Esempio (Cancellazione 2)

$$f'(x_0) \simeq \frac{f(x_0+h)-f(x_0)}{h}, \qquad h \text{ piccolo}$$

Sia $f(x) = \sin(x)$

$$\underbrace{\frac{\sin(x_0+h)-\sin(x_0)}{h}}_{Algoritmo I} = \underbrace{\frac{2}{h}\cos\frac{2x_0+h}{2}\sin\frac{h}{2}}_{Algoritmo II}$$

Stabilità di un algoritmo e condizionamento di un problema

Obiettivo: studiare come errori sui dati di un problema si propagano sui risultati.

Definizione

Un problema è **ben posto** *quando ammette una ed una sola soluzione e la soluzione dipende con continuità dai dati.*

Altrimenti il problema è **mal posto**.

Nel seguito assumeremo di lavorare sempre su problemi ben posti.

Consideriamo la risoluzione numerica di un problema ben posto. Nello studio della propagazione degli errori, occorre distinguere tra il ruolo assunto:

1. dal problema (**condizionamento del problema**)
2. dal particolare algoritmo usato per risolvere il problema (**stabilitá dell'algoritmo**)

Generico problema: assegnato il dato d, trovare x tale che

$$x = f(d) \tag{1}$$

Siano:

- δd una perturbazione del dato $d \longrightarrow$ dato $d + \delta d$
- $\bar{x} = f(d + \delta d)$ la soluzione esatta del problema con dato $d + \delta d$
- $\tilde{x}$ la risposta dell'algoritmo al dato $d + \delta d$.

NB: generalmente $\bar{x} \neq \tilde{x}$

Condizionamento

Come **il problema** reagisce alle inevitabili perturbazioni su dati?
Sia $\delta x = \bar{x} - x$

δx sarà grande o picolo rispetto a δd?

Domanda:

Le inevitabili perturbazioni sui dati del problema (δd) come si trasmettono sui risultati, **prescindendo dal particolare algoritmo che si vuole usare per risolvere il problema**?

Risposta:

Dipende dal problema!

Definizione (qualitativa!)

Un problema è **ben condizionato** *se le perturbazioni sui dati non influenzano eccessivamente i risultati.*

Un problema è **mal condizionato** *se le perturbazioni sui dati influenzano i risultati in misura molto grande.*

Definizione (Numero di condizionamento)

Se si ha una relazione del tipo

$$\frac{\|\delta x\|}{\|x\|} \leq K \frac{\|\delta d\|}{\|d\|}$$

o

$$\frac{\|\delta x\|}{\|x\|} \simeq K \frac{\|\delta d\|}{\|d\|}$$

per una qualche costante $K = K(d)$, il fattore $K(d)$ si definisce **numero di condizionamento del problema**.

Definizione (Numero di condizionamento)

Un problema si dice **ben condizionato** *se $K(d)$ è piccolo, sarà* **mal condizionato** *se $K(d)$ è grande.*

Esempio (Condizionamento della somma fra due numeri)

Consideriamo il problema di calcolare la somma di due numeri

$$x = a + b$$

(i dati sono $d_1 = a$ e $d_2 = b$, la soluzione del problema $x = a + b$).
Ci domandiamo se è un problema ben condizionato.
Siano x la soluzione esatta, $\bar{a} = a + \delta a$, $\bar{b} = b + \delta b$. Quindi

$$x + \delta x = a + \delta a + b + \delta b$$

da cui si ottiene

$$\delta x = \delta a + \delta b$$

Esempio (segue)

Pertanto

$$\frac{|\bar{x}-x|}{|x|} = \frac{|\delta x|}{|x|} = \frac{|\delta a+\delta b|}{|a+b|} \leq \frac{|\delta a|}{|a+b|} + \frac{|\delta b|}{|a+b|}$$
$$= \frac{|a|}{|a+b|}\frac{|\delta a|}{|a|} + \frac{|b|}{|a+b|}\frac{|\delta b|}{|b|}$$

Le costanti

$$K_a = \frac{|a|}{|a+b|}, \qquad K_b = \frac{|b|}{|a+b|}$$

sono i **coefficienti di amplificazione** *delle perturbazioni relative* $\frac{|\delta a|}{|a|}$ $\frac{|\delta b|}{|b|}$ *per questo problema. Possiamo prendere come numero di condizionamento* $K = \max(K_a, K_b)$.
Com'è dunque il condizionamento del problema?

Esempio (segue)

Si osserva che se $a+b \to 0$, si ha $K_a, K_b \to \infty$. Quindi il problema è mal condizionato se $a+b$ è piccolo.
Ma quando $a+b$ è piccolo? Quando a e b sono vicini in modulo e di segno opposto... in pratica la situazione in cui si verifica la **cancellazione numerica***!*
Quindi la cancellazione numerica può essere interpretata come il mal condizionamento della somma algebrica quando $a+b$ è piccolo (rispetto a ciascuno dei due addendi).

Più in generale si può stimare il numero di condizionamento di un problema usando **sviluppi di Taylor** ($\rightsquigarrow$ libro di testo).

Stabilità (ovvero: ruolo dell'algoritmo nella propagazione degli errori)

Il risultato finale di un algoritmo **dipende in maniera fondamentale** da come le perturbazioni, cioè i successivi errori compiuti ad ogni passo, si amplificano o si smorzano durante la risoluzione dei singoli problemi elementari.

Definizione (intuitiva)

Un algoritmo si dice **numericamente stabile** *se la successione delle operazioni di macchina non amplifica eccessivamente gli errori di arrotondamento.*

In pratica, tutte le operazioni intermedie e il risultato finale dell'algoritmo devono presentare un **errore relativo controllabile con la precisione di macchina**.

Definizione (quantitativa)

Se

$$\frac{\|\tilde{x} - \bar{x}\|}{\|\bar{x}\|} \simeq \varepsilon_{\mathrm{m}}$$

l'algoritmo si dice stabile

Partendo dagli stessi dati si possono avere algoritmi in cui l'errore finale é dell'ordine della precisione di macchina (stabili), altri in cui ciò non avviene (instabili).

L'algoritmo I dell'esempio Cancellazione 2 è **instabile** perché l'errore finale non è controllabile in termini della sola precisione di macchina, mentre l'algoritmo II genera un'errore controllabile con ε_{m}, quindi é **stabile**.