Appunti di Analisi Numerica*

Giuseppe Profiti

1 ottobre 2006

1 Distribuzione dei numeri finiti

Come visto negli esercizi della lezione precedente, c'è un numero finito di mantisse associate a diversi esponenti: ciascun esponente identifica un intervallo in cui quei numeri sono compresi.

Dentro ogni intervallo c'è sempre lo stesso numero di mantisse, gli intervalli però aumentano d'ampiezza con l'aumentare di p e quindi i numeri si diradano man mano che ci si allontana da 0.

Esempio

$$
\begin{aligned}
.100 \times 2^{-1} &= (1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 0 \cdot 2^{-3}) \times 2^{-1} = \left(\frac{1}{2} + \frac{1}{8}\right) \cdot \frac{1}{2} = \frac{4}{16} \\
.101 \times 2^{-1} &= (1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3}) \times 2^{-1} = \left(\frac{1}{2} + \frac{1}{4}\right) \cdot \frac{1}{2} = \frac{5}{16} \\
.110 \times 2^{-1} &= (1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3}) \times 2^{-1} = \left(\frac{1}{2} + \frac{1}{4}\right) \cdot \frac{1}{2} = \frac{6}{16} \\
.111 \times 2^{-1} &= (1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3}) \times 2^{-1} = \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8}\right) \cdot \frac{1}{2} = \frac{7}{16}
\end{aligned}
$$

Sono equidistanti nell'intervallo $\left[\frac{1}{4}, \frac{1}{2}\right)$.

Con 2^0 sono in $\left[\frac{1}{2}, 1\right)$ e ho come numeri $\frac{4}{8}, \frac{5}{8}, \frac{6}{8}, \frac{7}{8}$: sono sempre equidistanti ma l'intervallo ha una grandezza doppia.

Oltre al diradarsi man mano che ci si allontana dall'origine esiste un gap tra lo 0 e il primo numero finito (positivo o negativo): c'è un vuoto che dipende dall'esponente più piccolo.

2 Numeri finiti (sintesi e considerazioni)

$\alpha \in \Re \qquad \alpha = \pm(0.\alpha_1\alpha_2\ldots) \times \beta^p$
Se $\lambda \leq p \leq \omega, \quad \alpha_1 \neq 0, \quad \alpha_i = 0 \quad \forall i > t$
allora $\alpha \in \mathcal{F}(\beta, t, \lambda, \omega)$
Il reale è esattamente rappresentabile con un numero finito.
In alternativa

1. $p \in [\lambda, \omega]$ non è possibile rappresentarlo

   - sottocaso $p < \lambda$: underflow
   - sottocaso $p > \omega$ overflow

   Underflow può essere recuperato, ad esempio assegnandogli 0 (ma poi ho problemi se divido). Con overflow c'è errore e si blocca.

2. $p \in [\lambda, \omega]$ ma $\alpha \notin \mathcal{F}(\beta, t, \lambda, \omega)$ i.e. $\alpha_i \neq 0 \quad i > t$. In questo caso si associa un numero finito ad α per troncamento o arrotondamento: $\alpha \rightarrow fl(\alpha)$

3 Memorizzazione dei numeri finiti

Il numero è memorizzabile in un numero fisso di bit, ad esempio l bit.

bit è inteso come *casella*

- 1 bit di segno (bit 0)
- r bit per l'esponente (bit da 1 a r)
- i restanti bit formano la mantissa (bit da r+1 a l-1)

3.1 Segno

Il segno è memorizzato con 0 se $\alpha > 0$ e con $\beta - 1$ se $\alpha < 0$.

3.2 Esponente

Se si usasse il primo bit per memorizzare il segno si sprecherebbero molti esponenti. Si usa quindi una tecnica per traslazione.

Esempio

Con $\beta = 10$ e 2 bit per l'esponente si usano i numeri da 00 a 99 per memorizzare rispettivamente gli esponenti da -50 a 49. Per memorizzare si aggiunge un δ (in quesot caso 50) a p, quando si legge dalla memoria si sottrae.

3.3 Mantissa

Se $t < l - 1$ si riempiono gli spazi a destra con 0, altrimenti si tronca o si arrotonda.

3.4 Conversione di base

Un numero finito in una base potrebbe non essere finito in un'altra. Ad esempio un numero finito in base 10 può diventare periodico in base 2.

Esempio

Si prende $(0.1)_10$, per convertire in base 2 basta moltiplicare per 2 e usare la parte intera come cifra.

$$\begin{aligned} 0.1 \times 2 &= \mathbf{0}.2 \\ 0.2 \times 2 &= \mathbf{0}.4 \\ 0.4 \times 2 &= \mathbf{0}.8 \\ 0.8 \times 2 &= \mathbf{1}.6 \\ 0.6 \times 2 &= \mathbf{1}.2 \\ 0.2 \times 2 &= \mathbf{0}.4 \\ &\dots \end{aligned}$$

$(0.1)_10 = (0.\mathbf{00011}00110\dots)_2$.

Esercizio

In un qualsiasi linguaggio di programmazione, eseguire un for che sommi 10 volte il valore 0.1 a una variabile inizializzata a 0. Stampare il valore e confrontarlo con 1.

In python, eseguendo lo script[1] la stampa produce 1 ma il confronto dà errore. Da consolle invece risulta 0.9999999999999989

3.5 Esempi

$\mathcal{F}(10, 5, -50, 49)$, con 8 bit (ogni casella contiene un numero da 0 a 9), di cui 1 di segno e 2 di esponente.

[1] il link è vicino a quello di questo documento

$$\alpha = 0.1039 \times 10^{-6} \rightarrow 0\ \mathbf{44}\ 10390$$
$$\alpha = 0.0532 = 0.532 \times 10^{-1} \rightarrow 0\ \mathbf{49}\ 53200$$
$$\alpha = -27.141 = -0.27141 \times 10^{2} \rightarrow 9\ \mathbf{52}\ 27141$$
$$\alpha = -27.1416 = -0.271416 \times 10^{2} \rightarrow 9\ \mathbf{52}\ 27142 \text{ (arrotondamento)}$$

3.6 Standard ANSI/IEEE 754

Lo standard IEEE 754 (1985) è stato definito in modo tale che il codice eseguito su due diverse macchine abbia lo stesso risultato: i produttori di hardware devono seguire lo standard per garantire interoperabilità.

Stabilisce 4 formati floating point

- Basic
  - single, 32 bit (1+8+23)
  - double, 64 bit (1+11+52)
- Extended: specifica una misura minima (almeno X bit di mantissa)
  - single
  - double

É uno standard specifico per la base 2, quindi alcune cose sono particolari e ottimizzate.

Ad esempio in base 2 $\alpha_1 = 1$ sempre, quindi si può omettere e la mantissa guadagna 1 spazio. Nel caso Basic single si avrà quindi una mantissa di 24 bit (di cui solo 23 memorizzati: $\alpha_2 \ldots \alpha_2 4$).

Basic single può memorizzare $\mathcal{F}(2, 24, -126, 127)$

Il range di esponenti è ridotto (sono esclusi -127 e 128, cioé 0 e 255 in memoria) che sono riservati per casi particolari (gradual underflow e infinity).

Basic double può memorizzare $\mathcal{F}(2, 53, -1022, 1023)$ (anche qui due esponenti sono riservati)

Nota: anche NaN (not a number) è memorizzabile.

4 Errore assoluto e relativo

Sia $fl(\alpha)$ un'aprossimazione di $\alpha \in \Re$

Definizione 1 (di errore assoluto)

$$|\alpha - fl(\alpha)|$$

Definizione 2 (di errore relativo)

$$\frac{|\alpha - fl(\alpha)|}{|\alpha|} \; se \; \alpha \neq 0$$

L'errore assoluto cambia in base al valore di α a causa della distribuzione dei numeri finiti.

L'errore relativo normalizza ad α e quindi non soffre di questo problema.

Esempio/Esercizio

$\mathcal{F}(10, 3, \lambda, \omega) \quad \alpha = 1234567$ e $\alpha = 0.1234567 \times 10^{-2}$

Trovare l'errore assoluto e l'errore relativo. Considerare le informazioni fornite dai due tipi di errore e trarre considerazioni.

Primo numero:

$$\alpha = 1234567 = 0.1234567 \times 10^7 \qquad fl(\alpha) = 0.123 \times 10^7$$

$$|\alpha - fl(\alpha)| = |(0.1234567 - 0.123) \times 10^7| = 0.0004567 \times 10^7 = 0.4567 \times 10^4$$

$$\frac{|\alpha - fl(\alpha)|}{|\alpha|} = \frac{|(0.1234567 - 0.123) \times 10^7|}{|0.1234567 \times 10^7|} = \frac{0.0004567}{0.1234567} = 0.369927$$

Secondo numero:

$$\alpha = 0.1234567 \times 10^{-2} \qquad fl(\alpha) = 0.123 \times 10^{-2}$$

$$|\alpha - fl(\alpha)| = |(0.1234567 - 0.123) \times 10^{-2}| = 0.0004567 \times 10^{-2} = 0.4567 \times 10^{-6}$$

$$\frac{|\alpha - fl(\alpha)|}{|\alpha|} = \frac{|(0.1234567 - 0.123) \times 10^{-2}|}{|0.1234567 \times 10^{-2}|} = \frac{0.0004567}{0.1234567} = 0.369927$$

Considerazioni: l'errore relativo è uguale per entrambi i numeri, quindi questo non dipende da α.

Teorema 1 $\forall \alpha \in \Re$ *risulta che*

$$|fl_T(\alpha) - \alpha| \leq \beta^{p-t} \quad (1)$$

$$|fl_A(\alpha) - \alpha| \leq \frac{1}{2}\beta^{p-t} \quad (2)$$

Nota: l'errore dipende dall'esponente del numero.
Il teorema si dimostra a partire dagli intervalli.

$$X \leq \alpha \leq Y \qquad fl_T(\alpha) \equiv X$$

$$X = \left(\sum_1^t \alpha_i \beta^{-i}\right)\beta^p \qquad Y = \left(\sum_1^t \alpha_i \beta^{-i} + \beta^{-t}\right)\beta^p$$

$$\begin{aligned} |\alpha - fl_T(\alpha)| &\leq Y - X = \beta^{p-t} \\ |\alpha - fl_A(\alpha)| &\leq \frac{Y-X}{2} = \frac{1}{2}\beta^{p-t} \end{aligned}$$

Definizione 3 (di unità di arrotondamento) *Dato* $\mathcal{F}(\beta, t, \lambda, \omega)$ *si dice* unità di arrotondamento u *la quantità*

$$u = \begin{cases} \beta^{1-t} & \textit{tronc.} \\ \frac{1}{2}\beta^{1-t} & \textit{arrot.} \end{cases}$$

Teorema 2 $\forall \alpha \in \Re \qquad \alpha \neq 0$

$$\left|\frac{\alpha - fl(\alpha)}{\alpha}\right| < u$$

u è la precisione di rappresentazione, l'errore massimo che posso compiere passando da $\Re$ a $\mathcal{F}$.

Se facendo un calcolo l'errore è maggiore di u ci sono dei problemi.

Stimiamo
alpha|

$$|\alpha| = (\alpha_1\beta^{-1} + \alpha_2\beta^{-2} + \ldots)\beta^p \geq \alpha_1\beta^{-1}\beta^p \geq \beta^{p-1}$$

$$\frac{|\alpha - fl(\alpha)|}{|\alpha|} \leq \frac{|\alpha - fl(\alpha)|}{\beta^{p-1}} \quad (3)$$

- per troncamento

$$= \frac{\beta^{p-t}}{\beta^{p-1}} = \beta 1 - t$$

- per arrotondamento

$$\leq \frac{\frac{1}{2}\beta^{p-t}}{\beta p - 1} = \frac{1}{2}\beta^{1-t}$$

Nota: nelle dispense non ci sono questi conteggi, l'unica cosa importante è u come massimo dell'errore

Corollario 1 $\forall \alpha \in \Re\, \alpha \neq 0$ *vale che*

$$fl(\alpha) = \alpha \cdot (1 + \epsilon) \ \textit{con} \ |\epsilon| < u$$

ϵ è l'errore di rappresentazione. Grazie al teorema precedente posso definirlo come

$$\epsilon = \frac{fl(\alpha) - \alpha}{\alpha}$$

Esplicitando α si ottiene il corollario. Si poteva anche scrivere

$$\epsilon = \frac{\alpha - fl(\alpha)}{\alpha}$$

da cui si ottiene $fl(\alpha) = \alpha \cdot (1 - \epsilon)$.

Per semplicità usiamo la versione $\alpha \cdot (1 + \epsilon)$.

4.1 Caratterizzazione di u

u è il più piccolo numero finito positivo tale che

$$fl(u) + 1 > 1$$

cioé

$$\forall v < u \ fl(v + 1) = 1 \text{ con } v \in \mathcal{F}$$

Caso troncamento

$$1 + u = 1 + \beta^{1-t} = (0.1 + \beta^{-t})\beta$$

$$\begin{array}{r} 0.1+ \\ 0.\underbrace{000\ldots01}_{t} = \\ \hline 0.\underbrace{100\ldots01}_{t} \end{array}$$

Troncando resta tutto ed è ¿ 1. Ponendo $v < u$

$$\begin{array}{r} 0.1+ \\ \underline{0.000\ldots01 =} \\ 0.\underbrace{100\ldots0}_{t}1 \equiv 1 \end{array}$$

Caso arrotondamento

$$1 + \frac{1}{2}\beta^{1-t} = (0.1 + \frac{\beta}{2}\beta^{-t-1})\beta$$

$$\begin{array}{r} 0.1+ \\ \overline{0.00\ldots00\tfrac{\beta}{2} =} \\ 0.10\ldots00\tfrac{\beta}{2}+ \\ \overline{0.00\ldots00\tfrac{\beta}{2} =} \\ \overline{0.\underbrace{10\ldots01}_{t} > 1} \end{array}$$