CALCOLO NUMERICO → Risoluzione equazioni lineari
Risoluzione sistemi di equazioni non lineari
Calcolo di autovalori e autovettori

1|03|2021

RAPPRESENTAZIONE NUMERICA E ARITMETICA DI MACCHINA

L'obiettivo è quello di risolvere problemi matematici attraverso metodi numerici, potendo poi implementare i metodi numerici nel calcolatore mediante un particolare linguaggio di programmazione. La macchina non è un semplice spettatore passivo, ha delle caratteristiche legate alla sua aritmetica, per cui il primo punto da cui partire è proprio quello della costruzione di un modello dell'aritmetica di macchina per poi poter utilizzare questo modello per capire quali algoritmi è possibile utilizzare e quali invece no, in quanto non adatti ad essere utilizzati per un implementazione in macchina.
Il primo problema è quello della rappresentazione dei numeri. Di solito quando lavoriamo con i numeri utilizziamo una rappresentazione decimale

$$(1,34)_{10} \longrightarrow 1\cdot 10^{0} + 3\cdot 10^{-1} + 4\cdot 10^{-2}$$

CIFRE DELLA RAPPRESENTAZIONE "di"

↳ Numeri compresi tra 0 e 9 : $0 \le di \le 9$

A questo punto sorge una prima difficoltà, noi operiamo in base dieci, ma in generale quando operiamo con un calcolatore questo preferisce una rappresentazione in una base diversa dalla base dieci. La base di numerazione B è un numero naturale maggiore strettamente di 1.

B = BASE DI NUMERAZIONE : $B \in \mathbb{N}$; $B > 1$ ⟶ $0 \le di \le B-1$

Dobbiamo lavorare quindi con una base differente dalla base dieci, le aritmetiche di macchina oggi sono standardizzate in base 2, prima esistevano calcolatori dell'IBM che operavano in base 8 e alcuni in base 16. Cosa succede quando lavoro con una base differente dalla base 10 ? Concretamente nulla :

$(1.011)_2 = 1\cdot 2^{0} + 0\cdot 2^{-1} + 1\cdot 2^{-2} + 1\cdot 2^{-3}$ → Base 2

$(1.011)_3 = 1\cdot 3^{0} + 0\cdot 3^{-1} + 1\cdot 3^{-2} + 1\cdot 3^{-3}$ → Base 3

La rappresentazione non è più una somma di potenze di 10, ma sarà una somma di potenze di 2, 3, B, dove B è la generica base di numerazione.
A questo punto ci chiediamo come possiamo rappresentare in modo univoco ogni numero reale, noi ci poniamo il problema dei numeri reali, in quanto in generale andiamo a lavorare in aritmetica reale, se andassimo a lavorare in aritmetica complessa sulla macchina, basta ricordare che la macchina usa la naturale associazione di un numero complesso alla coppia di numeri reali (a,b), quindi rappresenta un numero complesso mediante una coppia di numeri reali

$(a,b) = a + ib$ dove i = unità immaginaria, $i^2 = -1$

Quindi il problema fondamentale è chiarire qual è la rappresentazione univoca di ogni numero reale, per poter poi passare/convertire questa rappresentazione in macchina.

TEOREMA DI RAPPRESENTAZIONE DEI NUMERI IN BASE B (Solo enunciato)

Sia $B \in \mathbb{N}$; $B > 1$ una base di numerazione

Sia $x \in \mathbb{R}$; $x \neq 0$ (Numero reale non nullo)

Allora $\exists!$ $p \in \mathbb{Z}$ (esponente)

$\{d_i\}_{i \in \mathbb{N}}$; $i \geq 1$ successione di cifre "d_i"

con $d_1 \neq 0$ e "d_i" non definitivamente uguale a $B-1$

Tali per cui : $x = \text{sgn}(x)\, B^{\overbrace{p}^{\text{ESPONENTE}}} \underbrace{\sum_{i=1}^{\infty} d_i B^{-i}}_{\text{MANTISSA}}$ dove $\text{sgn}(x) = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \end{cases}$

RAPPRESENTAZIONE NORMALIZZATA IN VIRGOLA MOBILE (Floating point representation)

- NORMALIZZATA : la prima cifra deve essere $\neq 0$; $d_1 \neq 0$, serve per garantire l'unicità della rappresentazione

$(0,01)_{10} = + 10^{0} \cdot (0,01)_{10}$ RAPPRESENTAZIONE NON AMMESSA , in quanto la prima cifra dopo la virgola deve essere $\neq 0$

$+ 10^{-1} \cdot (0,1)_{10}$

- VIRGOLA MOBILE : Utilizzando l'esponente sposto la posizione della virgola

$(0,01)_{10} \longrightarrow 10^{-1} \cdot (0,1)$

↳ Sposta la posizione della virgola

Abbiamo detto che non vogliamo che le cifre "di" siano da un certo punto in poi tutte uguali a B-1, ma questo perchè ? Consideriamo ad esempio :

$(0,11\ldots\ldots1)_2 = (0,\bar{1})_2$ RAPPRESENTAZIONI

$(0,99\ldots\ldots9)_{10} = (0,\bar{9})_{10}$ NON AMMESSE

Perchè sono rappresentazioni non ammesse ? Quelle considerate sono rappresentazioni non ammesse perchè viene meno l'unicità della rappresentazione

Consideriamo ad esempio :

$$0,\bar{9} = (0,99\ldots\ldots9)_{10} = \sum_{i=1}^{+\infty} 9 \cdot 10^{-i} = 9 \sum_{i=1}^{+\infty} 10^{-i} = 9 \sum_{i=1}^{+\infty} \left(\frac{1}{10}\right)^{i} = 9 \lim_{K \to \infty} \sum_{i=1}^{+\infty} \left(\frac{1}{10}\right)^{i}$$

SERIE GEOMETRICA DI RAGIONE $\frac{1}{10}$

OSSERVAZIONE

$\sum_{i=1}^{+\infty} (-1)^i$ ⟶ Per dare un significato alla sommatoria bisogna considerare $\lim_{K \to \infty} \sum_{i=1}^{k} (-1)^i$ NON CONVERGE

CONVERGENZA DELLA SERIE $\sum_{i=1}^{k} \left(\frac{1}{10}\right)^{i} = \sum_{i=1}^{k} a^i = a + a^2 + \ldots + a^k$

Se considero : $(1-a)(a + a^2 + \ldots + a^k) = a + a^2 + \ldots + a^k - (a^2 + a^3 + \ldots + a^{k+1}) = a - a^{k+1}$

Quindi : $(1-a)(a + a^2 + \ldots + a^k) = a - a^{k+1}$

$$(a + a^2 + \ldots + a^k) = \frac{a - a^{k+1}}{(1-a)}$$

Ricaviamo immediatamente che :

$$\lim_{K\to\infty} \sum_{i=1}^{+\infty} \left(\frac{1}{10}\right)^i = \lim_{K\to\infty} \frac{1/10 - (1/10)^{K-1}}{1 - 1/10} = \frac{1/10}{1 - 1/10} = \frac{1/10}{9/10} = \frac{1}{9}$$

Quindi : $9 \lim_{K\to\infty} \sum_{i=1}^{+\infty} \left(\frac{1}{10}\right)^i = 9\frac{1}{9} = 1 \longrightarrow$ ma era anche uguale a $0,\overline{9} = (0,9\ldots.9)_{10}$ quindi non vi è unicità, di conseguenza viene eliminata la rappresentazione $(0,99\ldots.9)_{10}$ dicendo che non vengono accettate rappresentazioni con : "di" definitivamente uguale a $B-1$

Quindi come rappresentazione : $0,\overline{9}$ Si $\frac{1}{10}$

Abbiamo quindi detto che un generico numero reale $x = 0$ viene rappresentato nella seguente forma :

$$x = \pm B^p \sum_{i=1}^{+\infty} di\, B^{-i}$$

Escludiamo $x = 0$, in quanto lo zero non ha una rappresentazione normalizzata, qualsiasi potenza è sempre diversa da zero e se consideriamo $d1 \neq 0$, non possiamo in nessun caso riuscire a rappresentare lo zero. Quindi lo zero verrà rappresentato in un modo particolare in macchina in quanto non ha una rappresentazione normalizzata. Questo teorema di importante, dal punto di vista dell'implementazione in macchina ci dice che un numero reale x R è caratterizzato da 3 concetti :

- Segno: $\pm$
- Esponente: 'p'
- Mantissa: sequenza di cifre 'di'

In macchina i numeri devono essere rappresentati su registri di lunghezza finita, quindi il numero sarà memorizzato in un registro di lunghezza finita come sequenza di bit, in base 2. Quando pensiamo ad un registro, immaginando di voler lavorare con una base B generica, pensiamo ad una successione di celle di memoria. Quindi abbiamo una sequenza finita di celle di memoria.

Il fatto che i registri abbiano lunghezza finita ci suggerisce immediatamente che non è possibile rappresentare in macchina tutti i numeri, ma è possibile rappresentare in macchina solo un insieme finito di numeri, che vengono chiamati numeri di macchina.
Quindi il problema fondamentale è : <u>quando inserisco un numero reale in macchina, la macchina cosa prende di quel numero reale ?</u>
La macchina prende un approssimazione del numero reale, con determinate caratteristiche che esamineremo.
In quanto il numero è caratterizzato da 3 quantità, il registro viene diviso in 3 sotto registri :

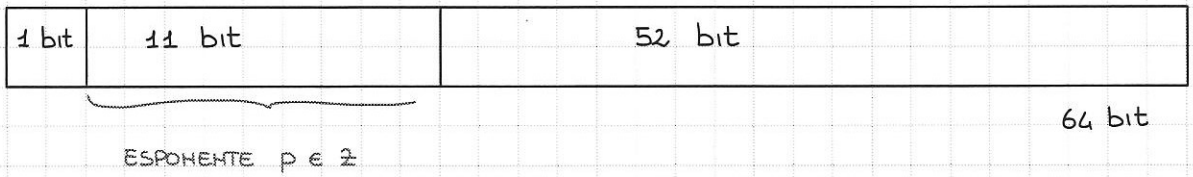
+ / −	ESPONENTE	MANTISSA

Standard IEEE (institute of electrical and electronics engineers) 754, negli anni 80 si instituì un gruppo che fissò uno standard per l'aritmetica di macchina, ovvero quali dovevano essere le caratteristiche dell'aritmetica di macchina per evitare che ogni azienda sviluppasse la propria aritmetica.

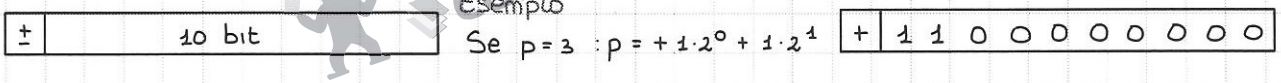
Lo standard IEEE 754 regola quindi l'aritmetica di macchina, ovvero la rappresentazione dei numeri di macchina, l'approssimazione dei numeri reali in numeri di macchina e le operazioni tra numeri di macchina. Questo standard viene definito anche 'standard per l'aritmetica in doppia precisione in virgola mobile' e viene utilizzato da matlab.

REGISTRI DA 64 Bit

(Singola precisione : REGISTRI 32 bit)



La prima ipotesi avanzata per rappresentare l'esponente fu quella di utilizzare un bit per il segno e 10 bit per rappresentare il numero



Questa ipotesi venne però scartata immediatamente, in quanto così facendo uno stesso numero viene rappresentato due volte, ed in particolare viene rappresentato due volte lo zero, + 0 e - 0. Quindi si pensò di utilizzare tutti gli 11 bit disponibili, potendo rappresentare tutti i numeri naturali :

Da 0.....0 = 0

a 1.....1 = $1\cdot 2^0 + 1\cdot 2^1 + 1\cdot 2^2 + \dots + 1\cdot 2^{10} = 2^{11} - 1 = 2047$

Quindi così facendo rappresentiamo tutti i numeri tra 0 e 2047, ma chiamando $\hat{p}$ questi numeri, l'esponente p in realtà sarà dato da :

↳ Numeri naturali che è possibile rappresentare nel registro

$p = \hat{p} - 1022 \longrightarrow p \in [-1021 ; 1024]$

↓ Esponente nella macchina

Mettiamo da parte 2 configurazioni

$\hat{p} = 0.....0$; $\hat{p} = 1.....1$

A questo punto guardando la mantissa noi abbiamo 52 bit a disposizione, ma secondo lo standard IEEE in quanto sto rappresentando i numeri in forma normalizzata, la prima cifra sarà necessariamente uguale a 1 e quindi la macchina non la rappresenta, di conseguenza i numeri normalizzati che la macchina riesce a rappresentare sono :

$$x = \pm 2^p \sum_{i=1}^{53} d_i 2^{-i}$$

con $d_1 = 1$ non rappresentato

$p \in [-1021 ; 1024]$

NUMERI NORMALIZZATI DI MACCHINA

Parto da "d_2" ; in quanto d_1 è sempre 1

Osservazione : Ci siamo tenuti da parte due configurazioni per l'esponente, che sono [$\hat{p}$ = 0000000] e [$\hat{p}$ = 1 1 1 1 1 1 1], quindi ci chiediamo ora cosa succede in questi due casi.

- Se $\hat{p} = 0.....0 \longrightarrow x = \pm 2^{-1022} \sum_{i=1}^{52} d_i 2^{-i}$ se $d_i = 0 \ \forall i$ → Rappresento lo zero DENORMALIZZATO, in questo caso anche d_1 può essere uguale a zero. Questo vale per tutti i numeri denormalizzati

- Se $\hat{p} = 1 \ldots\ldots 1 \longrightarrow d_1 \ldots\ldots d_{52} = 0$: Rappresento $\pm \infty$

↘ Altrimenti : NAN , Not a number , serve a gestire forme indeterminate $\frac{0}{0}$; $\frac{\infty}{\infty}$

NUMERI NORMALIZZATI

$$X = \pm 2^p \sum_{i=1}^{53} d_i 2^{-i}$$ ↓

con $d_1 = 1$; non rappresentata in macchina

$p \in [-1021 ; 1024]$

I numeri normalizzati sono simmetrici rispetto all'origine, possiamo osservare che esiste un più piccolo numero di macchina normalizzato positivo, che definiamo w :

$$\omega = 2^{-1021} 2^{-1} = 2^{-1022}$$

Esiste anche un più grande numero di macchina normalizzato positivo, che definiamo Ω :

$$\Omega = 2^{1024} \sum_{i=1}^{53} 1 \cdot 2^{-1}$$

Quindi i numeri di macchina normalizzati si distribuiscono tra :



Qual è la distanza tra un numero di macchina normalizzato e il suo successivo ?

Consideriamo un numero normalizzato :

$$x = 2^p \sum_{i=1}^{53} d_i 2^{-i}$$ il successivo numero di macchina è $$y = 2^p \left(\sum_{i=1}^{53} d_i 2^{-i} + 1 \cdot 2^{-53} \right)$$

Aggiungo un unità all'ultima cifra

Dipende dall'esponente p

DISTANZA TRA 2 NUMERI DI MACCHINA : 2^{p-53}

I numeri di macchina non sono uniformemente distribuiti, ma sono addensati vicino a w e diventano più diradati quanto più procedono da w a Ω e simmetricamente per numeri di macchina negativi.

Consideriamo uno distribuzione dei numeri di macchina fatta in questo modo in quanto dal punto di vista del calcolo scientifico non siamo interessati alla quantità di errore che si introduce ma alla quantità relativa di errore, cioè alla percentuale di errore. Se misuro la distanza della mia scrivania da centauri e faccio un errore di 1 cm, non lo considero ma se sto misurando la distanza tra due atomi in una certa reazione chimica, un errore di 1 cm è significativo; quindi l'obiettivo è quello di fare errori piccoli su numeri piccoli ed errori più grandi su numeri grandi, dove con errore si intende la percentuale di errore e quindi l'errore relativo. L'aritmetica di macchina è strutturata in maniera tale da avere piccoli errori relativi.

Ai numeri normalizzati devo associare i numeri denormalizzati :

$$X = \pm 2^{-1022} \sum_{i=1}^{52} d_i 2^{-i}$$

I numeri denormalizzati, compreso lo zero, si trovano tra -w e w, siccome in qualche modo voglio cercare di ottenere la massima precisione possibile sui numeri piccoli, cerco di popolare lo spazio tra -w e w, con lo zero ed altri numeri denormalizzati.