# Calcolo Numerico

Federico Matteoni

A.A. 2019/20

# Indice

# Introduzione

Prof.: Luca Gemignani

**Calcolo Numerico** Metodi numerici per risolvere problemi matematici con il calcolatore. In questo corso ce ne occuperemo dal punto di vista numerico, perché metodi di risoluzione diversi performano in maniera diversa sulla macchina. Cerchiamo di capire quali sono i metodi di interesse e cosa aspettarci dalla loro implementazione.
Il **metodi numerici approssimano la soluzione di problemi matematici**.
Inoltre, il computer **impatta** sul calcolo perché lavora con approssimazioni dei numeri, che su moli elevate di dati e di elaborazioni finiscono per perturbare il risultato ottenuto.

**Tipici problemi**

$$Ax = b$$

$$Ax = \lambda x$$

$$f(x) = 0$$

$$\int_a^b f(x)dx$$

**Matlab** Matrix Laboratory, strumento di implementazione per verificare e constatare i risultati teorici.

**Informazioni d'esame** Compitini, che se complessivamente passati rendono l'orale facoltativo. In alternativa, appelli scritti + orale.

# Capitolo 1

# Aritmetica di macchina

Modello per capire cosa aspettarci dal punto di vista degli errori dell'esecuzione.

**Esempio** Per calcolare il limite

$$\lim_{x\to\infty} \sqrt{x+1} - \sqrt{x}$$

ottengo un caso indeterminato $(\infty - \infty)$. Posso semplificare l'espressione ad esempio razionalizzando, e con pochi passaggi ottengo la seguente uguaglianza

$$\sqrt{x+1} - \sqrt{x} = \frac{1}{\sqrt{x+1} + \sqrt{x}}$$

Quindi dal punto di vista matematico, le due espressioni sono equivalenti. Ciò **non è sempre vero per la rappresentazione ed il calcolo in macchina**: le due espressioni forniranno risultati completamente diversi. Si rende **necessario**, quindi, **capire quale metodo si comporta meglio** rispetto agli altri.

## 1.1 Rappresentazione in base dei numeri in maniera univoca

**Rappresentare i numeri** Siamo comunemente abituati a rappresentare un numero in diverse forme.
Ad esempio, $0.1 = \frac{1}{10} = 10 \cdot 0.01$. In generale, per ogni numero ho diversi metodi di rappresentazione $\Rightarrow$ In macchina dobbiamo poter **rappresentare i numeri in maniera univoca**.

**Base di numerazione** $B \in N, B > 1$, poiché in base 1 non si può contare.
Una base $B$ ha cifre nell'insieme $\{0, 1, \ldots, B-1\}$

**Teorema** Dato $x \in R$, con $x \neq 0$
$\exists!$ $(\{d_i\}_{i\geq 1}$ con $d_1 \neq 0$ e $d_i$ non definitivamente uguali a $B-1) \wedge (p \in Z) \mid x = segno(x) \cdot B^p \cdot \sum_{i=1}^{\infty} d_i \cdot B^{-i}$
Considerazioni

Se $x \in C$ allora viene rappresentato come coppia di numeri reali, quindi il problema si riconduce sempre alla loro rappresentazione

Lo 0 viene rappresentato in modo speciale, poiché non esiste una sua rappresentazione normalizzata

$d_{ii\geq 1}$ è una **successione** di cifre

La rappresentazione è **normalizzata** se $d_1 \neq 0$, cioè se la prima cifra è diversa da 0

Non può avere tutte cifre uguali a $B-1$ da un certo punto in poi, la rappresentazione "collassa" al numero successivo

$p$ è detto **esponente**

$\sum_{i=1}^{\infty} d_i \cdot B^{-i}$ è detta **mantissa**

Questa rappresentazione si chiama **in virgola mobile** o **floating point**

| BASE | MANTISSA |
|---|---|

**Esempi** Ponendo $B = 10$

$x = 0.01 \Rightarrow x = +10^{-1} \cdot (0.1)$

$x = 1.35 \Rightarrow x = +10^{1} \cdot (0.135)$

$x = 0.0023 \Rightarrow x = +10^{-2} \cdot (0.23)$

**Computer** Nei computer ho **registri di lunghezza finita**, quindi essi vengono partizionati: una parte viene riservata alla rappresentazione dell'**esponente** e il resto alla rappresentazione della **mantissa**. L'insieme dei numeri di macchina F è quindi così definito

$$F(B,t,m,M) = \{\pm B^p \cdot \sum_{i=1}^{t} d_i \cdot b^{-i} \;\Big|\; d_1 \neq 0 \wedge -m \leq p \leq M\} \cup \{0\}$$

dove

$t$ sono le **cifre della mantissa**

L'esponente p è compreso tra i valori $-m$ e $M$

Lo 0 è incluso ma rappresentato a parte

**Esempio** Ipotizzando di usare registri da 32 bit, posso stanziare 8 bit per l'esponente $p$ (quindi 1 bit per il segno e 7 bit per il valore) e i restanti 24 bit per la mantissa (1 per il segno, 23 per il valore). Quanti numeri posso rappresentare?

$p$ di 7 bit $\Rightarrow 2^7 - 1 = 127 \Rightarrow -127 \leq p \leq 127$ ma lo 0 è rappresentato due volte

$x = \pm 2^p \sum_{i=1}^{2} 3d_i \cdot 2^{-1}$, con $d_i \in \{0,1\}$, $d_1 \neq 0 \Rightarrow d_1 = 1$ sempre

Vedremo che con una serie di accorgimenti è possibile aumentare i numeri esattamente rappresentabili.