Corso di Calcolo Numerico

Corso di Laurea in MATEMATICA

A.A. 2023-24

... anche quest'anno possiamo divertirci ...

Prof. Stefano De Marchi

Università degli studi di Padova
Dipartimento di Matematica "Tullio Levi-Civita"
http://www.math.unipd.it/~demarchi

Lezione 1

Sommario

Informazioni sul corso
Programma

Cos'è il calcolo numerico

Aritmetica del computer

Docenti

- prof. Stefano De Marchi: lezioni frontali
  email: stefano.demarchi@unipd.it;
  ricevimento: su appuntamento o per email
  studio: Dipartimento di Matematica "Tullio Levi-Civita", Torre Archimede, Piano 3, stanza 3CD6.
- dott. Giacomo Elefante: laboratorio
  email: giacomo.elefante@unipd.it;
  ricevimento: su appuntamento o per email
  studio: Dipartimento di Matematica "Tullio Levi-Civita", Torre Archimede, Piano 5, stanza 5BC1.
- Tutor:dott. Marco Barbieri
- Pagine moodle del corso
  - https://stem.elearning.unipd.it/course/view.php?id=6331

Orario delle lezioni

- Mercoledì e Venerdì: 14:30-16:30 aula 1A150 (48h)
- Laboratorio: 16h, da giovedi' 5 ottobre in LABP140 ore 9:30-11:30, Il docente vi dirà dove e quando si terranno le altre lezioni

Importante

Se dovessero intervenire modifiche, sospensioni e informazioni dell'ultimo minuto, saranno comunicate attraverso Moodle

Organizzazione del corso

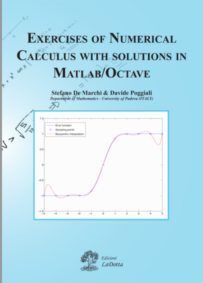
6 CFU=56 h

di cui
6 CFU = 40h frontali in aula
3 CFU = 16h di esercitazioni di laboratorio dove si userà MATLAB

Si richiede uno studio costante individuale!

Materiale didattico

1. Introduzione al Calcolo Numerico (in Italian) Ristampa corretta 2018, pp. 260, ISBN: 9788874889396 Ed. Esculapio (Bo)
2. Exercises of Numerical Calculus https://www.amazon.it/Exercises-numerical-calculus-solutions-MATLAB/dp/8898648480



Moodle e altro

Moodle

Slides, proposte di esercitazioni/quiz, informazioni varie.

Online: Oltre alle cose che si possono trovare in rete, al link https://www.math.unipd.it/~demarchi/didattica.html si trova materiale di esercizi ed esami proposti negli anni didattici precedenti

Modalità d'esame

- L'esame finale avviene mediante prova in Moodle Esami divisa in parte scritta (S) e parte di programmazione Matlab (P).
- La prova consisterà in 4 esercizi di teoria (con eventuali algoritmi) da $7/30$ punti; un esercizio in Matlab con punteggio max di $4/30$.
- vedremo se valutare i quiz che metteremo online in qualche modo premiale per chi li fa bene.
- $0 \leq S \leq 28$, $0 \leq P \leq 4$. Se $S \geq 18$ e $P \geq 1$, Il voto finale V, in trentesimi (con eventuale lode), sarà quello che verrà registrato.

$$\boxed{V = S + P \leq 32}$$

- Se $V \geq 26$ si può chiedere di fare l'orale (incremento max di 4 punti).

Appelli

Date : 24/1, 16/2 (questi in LabP140), 21/6, 15/7 e 26/8 (questi 3 in LabTA). Sempre ore 14-17.

Programma del corso: I

- **Aritmetica del computer**: rappresentazione dei numeri, operazioni macchina, errori, stabilità e condizionamento.
- **Soluzioni di equazioni non lineari**: metodi iterativi, successioni convergenti. Metodo di bisezione. Metodi di punto fisso. Metodo di Newton. Test di arresto. Cenni sui sistemi non lineari.
- **Soluzione di sistemi lineari**: costo computazionale; errori e condizionamento; stime dell'errore. Metodi diretti: eliminazione di Gauss. Fattorizzazioni di matrici: LU, Cholesky e QR (cenni). Calcolo del determinante e dell'inversa di una matrice. Metodi iterativi: Jacobi, Gauss-Seidel, SOR. Test di arresto.
- **Cenni sul calcolo numerico di autovalori**: autovalore massimo (metodo delle potenze) e minimo (metodo delle potenze inverse).

Programma del corso: II

- **Interpolazione e approssimazione polinomiale**: Vandermonde e Lagrange. Analisi dell'errore. Differenze Divise e interpolazione di Newton. Fenomeno di Runge. Punti di Chebsyhev. Stabilità e costante di Lebesgue. Splines e curve di Beziér. Approssimazione ai minimi quadrati discreti.
- **Quadratura**: Formule interpolatorie di Newton-Cotes, semplici e composite, cenni alle formule Gaussiane.

Prerequisiti necessari

1. Analisi Matematica: soprattutto studio di funzioni, derivate, integrali, successioni, o-piccolo e O-grande.
2. Algebra lineare e geometria: spazi vettoriali, calcolo matriciale (determinanti) e soluzione di sistemi lineari.

Sommario

Informazioni sul corso

Cos'è il calcolo numerico

Aritmetica del computer

A cosa serve il Calcolo Numerico

- Calcolare soluzioni approssimate di zeri di funzione, sistemi linerai, integrali e derivate
- Risolvere modelli fisici con metodi numerici
- Risolvere problemi inversi (ricostruire immagini da TAC, Risonananza,...)
- Problemi di ottimizzazione (scheduling di aeroporti, raccolta differenziata)
- Geometria computazionale (costruire curve piane o nello spazio, superfici, volumi per CAD, CAGD, fluidodinamica,...)

Ingredienti

1. Analisi di algoritmi stabili ed efficienti
2. Programmazione (**MATLAB**, Python, C, Java,....)

MATLAB

MATLAB significa **MAT**rix **LAB**oratory: sarà il software che useremo per implementare gli algoritmi numerici

Licenze Matlab e Office 365 di Ateneo

1. https://asit.unipd.it/servizi/contratti-software-licenze/matlab
2. https://asit.unipd.it/servizi/servizi-utenti-istituzionali/contratti-software-licenze/microsoft-office-365

Esempio PATRIOT

- Durante la prima del guerra del golfo (Ago. 90- Feb. 91), il sistema antimissile americano "Patriot" non è riuscito a intercettare un missile Scud iracheno. Il missile causò la morte di 28 soldati e il ferimento di 100 persone.
- "Patriot" monitorava la posizione dei missili da intercettare sulla base dei rilevamenti precedenti, basandosi su posizione e velocità del missile.
- Un report del General Accounting office spiega che l' errore è stato dovuto essenzialmente al calcolo inaccurato del tempo a causa di errori di arrotondamento.

Continua

Motivo del guasto

Erronea conversione di un numero in formato floating-point a 64 bit (8 byte) (correlato alla velocità orizzontale del razzo rispetto alla piattaforma) in un intero a 16 bit (2 byte): il numero risultante, maggiore del massimo intero rappresentabile pari a 32768, provocò un overflow nella misura della velocità orizzontale con il conseguente spegnimento dei razzi ed esplosione del missile.

Termini da chiarire

- rappresentazione dei numeri,
- bit, byte
- aritmetica floating-point
- errori di arrotondamento

Continua

Motivo del guasto

Erronea conversione di un numero in formato floating-point a 64 bit (8 byte) (correlato alla velocità orizzontale del razzo rispetto alla piattaforma) in un intero a 16 bit (2 byte): il numero risultante, maggiore del massimo intero rappresentabile pari a 32768, provocò un overflow nella misura della velocità orizzontale con il conseguente spegnimento dei razzi ed esplosione del missile.

Termini da chiarire

- rappresentazione dei numeri,
- bit, byte
- aritmetica floating-point
- errori di arrotondamento

Sommario

Informazioni sul corso

Cos'è il calcolo numerico

Aritmetica del computer
- Rappresentazione dei numeri macchina
- Standard IEEE 754-1985
- Trasformazione da decimali a binario

Rappresentazione dei numeri in *virgola mobile*

Dato $a \in \mathbb{R}$, scriveremo

$$\boxed{a = s \cdot p \cdot N^q} \tag{1}$$

$s = \pm 1$ è il segno, p si chiama mantissa (o significando), N è la base ($N = 2$, base binaria) e $q \in \mathbb{Z}$ esponente.

- La notazione non è unica. Infatti

$$a = pN^q = p_1 N^{q-1} = p_2 N^{q+1}$$

con $p_1 = Np$ e $p_2 = p/N$. Esempio:
$a = 1.3 \cdot 10^1 = 0.13 \cdot 10^2 = 13.0 \cdot 10^3$.

- Se la mantissa p è tale che

$$\frac{1}{N} \leq |p| < 1$$

allora la rappresentazione (1) si dice *normalizzata*

Rappresentazione dei numeri in *virgola mobile*

Dato $a \in \mathbb{R}$, scriveremo

$$\boxed{a = s \cdot p \cdot N^q} \tag{1}$$

$s = \pm 1$ è il segno, p si chiama `mantissa` (o significando), N è la base ($N = 2$, base binaria) e $q \in \mathbb{Z}$ `esponente`.

- La notazione non è unica. Infatti

  $$a = pN^q = p_1 N^{q-1} = p_2 N^{q+1}$$

  con $p_1 = Np$ e $p_2 = p/N$. Esempio:
  $a = 1.3 \cdot 10^1 = 0.13 \cdot 10^2 = 13.0 \cdot 10^3$.
- Se la mantissa p è tale che

  $$\frac{1}{N} \leq |p| < 1$$

  allora la rappresentazione (1) si dice *normalizzata*

Continua

Fissata N, un numero a è identificato dalla coppia (p, q). Nel caso $a = 0$, $(p, q) = (0, 0)$. In generale

s	q	$\|p\|$

Table: Rappresentazione dei numeri in un calcolatore

dove $s = 0$ se il segno è $+$ e $s = 1$ se è $-$; q lo spazio per l'esponente e $|p|$ lo spazio per la mantissa normalizzata.

Definizione

Si chiama `numero macchina` *un numero tale che p e q sono rappresentabili esattamente negli spazi riservati.*

Esempio: se $|p|$ è lunga $t = 5$, i numeri macchina sono tutti e soli quelli che hanno la mantissa normalizzata con al più $t = 5$ cifre

Esempio

Supponiamo base decimale, $N = 10$, $\boxed{\pm .xxx \pm ee}$ ($t = 3$ con segno e 2 cifre per esponente)

- $-99 \leq q \leq +99$
- intervallo di rappresentazione

$$[-0.999 \cdot 10^{+99}, -0.100 \cdot 10^{-99}] \cup [0.100 \cdot 10^{-99}, 0.999 \cdot 10^{+99}]$$

- a sx del minimo di sx: overflow negativo (-Inf)
- a dx del massimo si dx: overflow positivo (Inf)
- in $(-0.100 \cdot 10^{-99}, 0.100 \cdot 10^{-99})$ underflow (NaN) (numeri non rappresentabili)

Base, Esponente, Singola Precisione (SP) e Doppia Precisione (DP)

Base binaria $\boxed{N=2}$

Esponente $\boxed{m \leq q \leq M}$ dove il minimo $m < 0$ e il massimo $M > 0$, dipendono dall'aritmetica usata da un calcolatore.

Posto $q^* = q - m \geq 0$ allora $0 \leq q^* \leq M - m$

- SP (Singola Precisione, 32bit=4byte)

1 s 1	2 q 9	10 $\lvert p \rvert$ 32

Table: Rappresentazione in singola precisione: 1+8+23

- DP (Doppia Precisione, 64bit=8byte)

1 s 1	2 q 12	13 $\lvert p \rvert$ 64

Table: Rappresentazione in doppia precisione: 1+11+52

Curiosità

1 BIT= solo valore 0 o 1;
1 BYTE = sequenza di 8 BIT contingui. Esempio: 10010100.
Byte significa BIt octaTE.

1Kb=2^{10} bytes= 1.024 bytes; (10^3 bytes)
1Mb=2^{20} bytes= 1.048.576 bytes; (10^6 bytes)
1Gb=2^{30} bytes= 1.073.741.824 bytes; (10^9 bytes)
1Tb=2^{40} bytes= 1.099.511.627.776 bytes; (10^{12} bytes)
Ulteriore curiosità: il cervello umano può memorizzare un Peta byte, 1Pb, ovvero 10^{15} bytes.

Curiosità

1 BIT= solo valore 0 o 1;
1 BYTE = sequenza di 8 BIT contingui. Esempio: 10010100.
Byte significa BIt octaTE.

1Kb=2^{10} bytes= 1.024 bytes; (10^3 bytes)
1Mb=2^{20} bytes= 1.048.576 bytes; (10^6 bytes)
1Gb=2^{30} bytes= 1.073.741.824 bytes; (10^9 bytes)
1Tb=2^{40} bytes= 1.099.511.627.776 bytes; (10^{12} bytes)
Ulteriore curiosità: il cervello umano può memorizzare un Peta byte, 1Pb, ovvero 10^{15} bytes.

Standard IEEE 754-1985

$$\boxed{a = \pm 1.a_2 a_3 \dots a_t a_{t+1} \cdot 2^q}$$

- la mantissa ha t cifre (a_2 a a_{t+1}) con la prima sempre uguale ad 1!
- la mantissa normalizzata è $1 \leq |p| < 2$ (nota: 1 e 2 sono due potenze successive della base binaria)
- bias (scostamento), per avere q^* sempre positivi. Esempio: $q^* = 200$ indica l'esponente $200 - 127 = 73$.

	SP	DP
# bit	32 (4 bytes)	64 (8 bytes)
segno s	1 bit	1 bit
$\lvert p\rvert$	23	52
q	8	11
bias	127 (011....1)	1023 (011....1)
max q	127	1023
min q	-126	-1022

Table: Occupazione dei registri di memoria in SP e DP

Esponenti speciali

SP: $-127 \leq q \leq 128$, con -127 e 128 usati per numeri particolari

DP: $-1023 \leq q \leq 1024$, con -1023 e 1024 usati per numeri particolari

- SP: gli intervalli di rappresentazione sono $(-2^{128}, -2^{-126}]$, $[2^{-126}, 2^{128})$ con $(-2^{-126}, 2^{-126})$, intervallo di underflow
- SP: il più piccolo numero rappresentabile è $\underbrace{1.00\ldots0}_{mantissa} \cdot 2^{-126} \approx 10^{-38}$

0	$\underbrace{0 \ldots\ldots 01}_{-126+127=1}$	0...0

Table: Il più piccolo numero in SP

Nota: l'esponente con tutte le cifre 0 (zero) si usa solo per indicare lo 0 oppure l'underflow.

Continua

- SP: il più grande numero rappresentabile è
  $\underbrace{1.11\ldots1}_{mantissa} \cdot 2^{127} = (1+1-2^{-23})\cdot 2^{127} \approx 10^{38}$

0	$\underbrace{1\ldots\ldots10}_{127+127=254}$	$1\ldots1$

Nota: l'esponente 255 (zero) si usa solo per indicare l' overflow.

Similmemte in DP

- il più piccolo numero floating point rappresentabile è
  $\underbrace{1.00\ldots\ldots0}_{1+51\ cifre} \cdot 2^{-1022} = 2^{-1022} \approx 10^{-308}$
- il più grande numero floating point rappresentabile è
  $\underbrace{1.11\ldots\ldots1}_{1+51\ cifre} \cdot 2^{1023} = (1+1-2^{-52})\cdot 2^{1023} \approx 10^{308}$

Riassumiamo (SP)

IEEE 754: estremi degli intervalli

- *Più grande normalizzato* $\sim 2^{128}$:

  X 11111110 11111111111111111111111

  $\pm$ 2^{127} ~ 2

- *Più piccolo normalizzato* 2^{-126} :

  X 00000001 00000000000000000000000

  $\pm$ 2^{-126} 1

Da numero decimale a binario

Vogliamo trasformare 39.345 in binario

- Parte intera: si procede con divisioni successive per 2 prendendone i resti da dx a sx

$$\underbrace{39}_{1} : 2 = \underbrace{19}_{1} : 2 = \underbrace{9}_{1} : 2 = \underbrace{4}_{0} : 2 = \underbrace{2}_{0} : 2 = 1$$

  pertanto $(39)_{10} = (100111)_2$

- Parte frazionaria: si procede con moltiplicazioni successive per 2 prendendone la parte intera. Vediamone i primi passi

$$\begin{aligned} 0.345 \cdot 2 &= 0.69 \rightarrow 0 \\ 0.69 \cdot 2 &= 1.38 \rightarrow 1 \\ 0.38 \cdot 2 &= 0.76 \rightarrow 0 \\ 0.76 \cdot 2 &= 1.52 \rightarrow 1 \end{aligned}$$

  il processo potrebbe non terminare ... Nel nostro caso una approssimazione è $(0.345)_{10} = (0.0101100)_2$

Esempio standard IEEE 754-1985

Supponiamo di volere rappresentare in SP il numero decimale $a = 43.6875$ nello standard EEE 754-1985.

(i) Trasformare il numero, senza segno, in forma binaria. La parte intera, $(43)_{10}$, diventa, $43_{10} = (101011)_2$. La parte frazionaria $0.6875_{10} = (1011)_2$. Complessivamente $(43.6875)_{10} = (101011.1011)_2$.

(ii) Successivamente spostiamo la virgola verso sinistra, lasciando solo un 1 alla sinistra: $1.010111011 \cdot 2^5$.

La mantissa viene quindi riempita con zeri a destra, fino a completare i 23 bit. Il numero che si ottiene è $1.01011101100000000000000$.

(iii) L'esponente 5, dobbiamo convertirlo in forma binaria e adattarlo allo standard. Per la singola precisione, dobbiamo aggiungere 127 (detto anche `bias`) ovvero $5 + 127 = 132_{10} = 10000100_2$.

Link: https://www.h-schmidt.net/FloatConverter/IEEE754.html

Esercizi proposti (SP, 32bit)

1. da base 2 a base 10

$$(1\ 10000001\ 01000000000000000000000)_2 = (\ )_{10}$$

$$(0\ 10000011\ 10011000000000000000000)_2 = (\ )_{10}$$

2. da base 10 a base 2

$$(-13.75)_{10} = (\ )_2$$

$$(8.5)_{10} = (\ )_2$$