

Esercitazione 2:

Errori macchina

Stabilità numerica

Esercizio 1

Studiare dal punto di vista numerico (stabilità e condizionamento) le radici dell'equazione di II grado $ax^2 + bx + c = 0$, con $\sqrt{b^2 - 4ac} \geq 0$. Le radici vengono calcolate mediante le formule:

1. Formulazione non numericamente stabile per $b^2 \gg 4ac$:

$$x_{1,2\text{algo1}} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a},$$

2. Formulazione numericamente stabile per $b^2 \gg 4ac$:

$$x_{1\text{algo2}} = \frac{-b - \text{sign}(b)\sqrt{b^2 - 4ac}}{2a},$$

$$x_{2\text{algo2}} = \frac{c}{ax_{1\text{algo2}}},$$

3. Mediante la routine MatLab `roots([a b c])` per la ricerca dei zeri dei polinomi.

Provare l'algoritmo per valori degli zeri pari a $x_{1\text{esatta}} = 10^{+4}$ e $x_{2\text{esatta}} = 10^{-6}$ dove $a = 1$, $b = -(x_{2\text{esatta}} + x_{1\text{esatta}})$ e $c = x_{1\text{esatta}}x_{2\text{esatta}}$.

Commentare i risultati ottenuti.

La stampa dei risultati può essere ottenuta dai seguenti comandi MatLab:

```
% Stampa informazioni a video
disp('Visualizzo radici trovate');
fprintf('%10s = %20.15e    %10s = %20.15e\n', ...
        'x1-esatta', x1-esatta, 'x2-esatta', x2-esatta);
```

```

fprintf('%10s = %20.15e    %10s = %20.15e\n',...
        'x1_algo1',x1_algo1,'x2_algo1',x2_algo1);
fprintf('%10s = %20.15e    %10s = %20.15e\n',...
        'x1_algo2',x1_algo2,'x2_algo2',x2_algo2);
fprintf('%10s = %20.15e    %10s = %20.15e\n\n',...
        'x1_algo3',x1_algo3,'x2_algo3',x2_algo3);

disp('Visualizzo errori relativi');
fprintf('%10s = %20.15e    %10s = %20.15e\n',...
        'er_x1_algo1',er_x1_algo1,'er_x2_algo1',er_x2_algo1);
fprintf('%10s = %20.15e    %10s = %20.15e\n',...
        'er_x1_algo2',er_x1_algo2,'er_x2_algo2',er_x2_algo2);
fprintf('%10s = %20.15e    %10s = %20.15e\n',...
        'er_x1_algo3',er_x1_algo3,'er_x2_algo3',er_x2_algo3);

```

Risultati

Visualizzo radici trovate

x1_esatta = 1.000000000000000e+004	x2_esatta = 1.000000000000000e-006
x1_algo1 = 1.000000000000000e+004	x2_algo1 = 1.000000338535756e-006
x1_algo2 = 1.000000000000000e+004	x2_algo2 = 1.000000000000000e-006
x1_algo3 = 1.000000000000000e+004	x2_algo3 = 1.000000000000000e-006

Visualizzo errori relativi

er_x1_algo1 = 0.000000000000000e+000	er_x2_algo1 = 3.385357559179783e-007
er_x1_algo2 = 0.000000000000000e+000	er_x2_algo2 = 0.000000000000000e+000
er_x1_algo3 = 0.000000000000000e+000	er_x2_algo3 = 2.117582368135751e-016

Esercizio 2

Fornire un programma per il calcolo di EPS. Ricordare perchè il valore EPS è importante (ricordarsi della relazione $\frac{a - fl(a)}{a} \leq \text{EPS}$).

Lo pseudo-codice dell'algoritmo è il seguente:

```
1  % Inizializzazione
2  myeps ← 1;
3  % Ciclo principale divisione per 2
4  while 1 + myeps > 1 do
5      myeps ← myeps / 2
6  end
7  % Diverso da ed e' per questo che moltiplico per due
8  myeps ← 2 · myeps
```

Risultati

myeps =

2.220446049250313e-016

Esercizio 3

Per calcolare il seguente integrale:

$$I_n = e^{-1} \int_0^1 x^n e^x dx$$

si possono utilizzare le seguenti due formulazioni equivalenti

1. Formulazione non numericamente stabile: Per $n = 0$ si ha che

$$I_0 = e^{-1} \int_0^1 e^x dx = e^{-1}(e^1 - 1).$$

Per $n = 1$, è facile verificare che, essendo

$$\int x \exp(x) dx = (x - 1) \exp(x) + c,$$

dal secondo teorema del calcolo integrale si ha che $I_1 = e^{-1}$ e più in generale integrando per parti che

$$I_{n+1} = e^{-1} \left(x^{n+1} e^x \Big|_0^1 - (n+1) \int_0^1 x^n e^x dx \right) = 1 - (n+1) I_n.$$

Da cui la formula di ricorrenza:

$$I_{n+1} = 1 - (n+1) I_n$$

con

$$I_1 = e^{-1}.$$

2. Formulazione numericamente stabile: Dal fatto che l'integranda $x^n \exp x > 0$ per $x \in (0, 1]$ si ha che

$$I_n > 0.$$

Inoltre essendo $x \leq 1$ implica che $x^2 \leq x$ e più in generale che $x^{n+1} \leq x^n$ da cui $x^{n+1} \exp(x) \leq x^n \exp(x)$ e quindi

$$I_{n+1} \leq I_n.$$

La successione I_n è positiva e non crescente e quindi ammette limite L finito in particolare si dimostra che questo limite è zero.

Da cui la formula di ricorrenza:

$$I_n = \frac{1 - I_{n+1}}{n+1}$$

con

$$I_\infty = 0.$$

Commentare i risultati ottenuti.

Il pseudo-codice che implementa la prima successione è il seguente.

```
1  % Inizializzazione di s a (vettore) zero di 100 elementi (utilizzo zeros)
2   $s \leftarrow [0, 0, \dots, 0]$ ;
3  % caso base
4   $s(1) \leftarrow \exp(-1)$ ;
5  % Ciclo iterativo
6  for  $n \leftarrow 1$  to  $\text{length}(s) - 1$  do
7       $s(n + 1) \leftarrow 1 - (n + 1)s(n)$ 
8  end
```

Il pseudo-codice che implementa la seconda successione è il seguente.

```
1  % Inizializzazione di s a (vettore) zero di 100 elementi (utilizzo zeros)
2   $t \leftarrow [0, 0, \dots, 0]$ ;
3  % caso base (scrittura opzionale)
4   $t(\text{end}) \leftarrow 0$ ;
5  % Ciclo iterativo
6  for  $n \leftarrow \text{length}(t)$  to 2 with step  $-1$  do
7       $t(n - 1) \leftarrow (1 - t(n))/n$ 
8  end
```

Esempi MatLab di ciclo for incrementali sono i seguenti.

```
% Ciclo for incrementale (passo 1) da 1 a 5
for i=1:5
    i
end
% Ciclo for incrementale (passo 2) da 1 a 5
for i=1:2:5
    i
end
```

Esempi MatLab di ciclo for decrementali sono i seguenti.

```
% Ciclo for decrementale (passo -1) da 5 a 1
for i=5:-1:1
    i
end
% Ciclo for decrementale (passo -2) da 5 a 1
for i=5:-2:1
    i
end
```

La visualizzazione grafica della soluzione può essere effettuata mediante i seguenti comandi MatLab.

```
% Visualizzazione delle soluzioni
figure;
clf;
indvis = 1:30;
h=semilogy(indvis,abs(s(indvis)),'k-');
set(h,'LineWidth',2);
hold on;
h=semilogy(indvis,abs(t(indvis)),'m-');
set(h,'LineWidth',2);
hold off;
legend('s','t')
title('Esempio di succ. da formulazione integrale');
xlabel('Iterazione');
```

Risultati

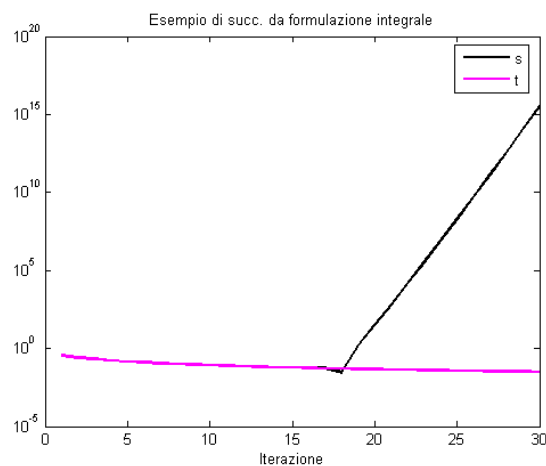


Figura 1: Risultati esercizio 3: Confronto delle due successioni.

Esercizio 4

Esistono diverse successioni numeriche che convergono a π . Si implementino le successioni $\{u_n\}$, $\{z_n\}$, definite rispettivamente come

$$\begin{cases} s_1 = 1, & s_2 = 1 + \frac{1}{4} \\ u_1 = 1, & u_2 = 1 + \frac{1}{4} \\ s_{n+1} = s_n + \frac{1}{(n+1)^2} \\ u_{n+1} = \sqrt{6 s_{n+1}} \end{cases}$$

e

$$\begin{cases} z_1 = 1, & z_2 = 2 \\ z_{n+1} = 2^{n-\frac{1}{2}} \sqrt{1 - \sqrt{1 - 4^{1-n} \cdot z_n^2}} \end{cases}$$

che *teoricamente* convergono a π .

La prima successione deriva dalla relazione:

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{1}{4\pi} \int_{-\pi}^{\pi} x^2 dx = \frac{\pi^2}{6}$$

mentre la seconda è la successione di Archimede: In una circonferenza di raggio unitario vengono iscritti successivamente poligoni regolari di 2^n lati (con $n = 2, 3, \dots$). L'area A_n di ciascun poligono (che tende ad aumentare delle suddivisioni a π) è data dal prodotto del numero di triangoli (2^n) per l'area di ciascun triangolo ($\frac{1}{2} \sin(\theta)$), con $\theta = \frac{2\pi}{2^n}$ l'angolo al centro di ogni triangolo. Le formula precedente si ottiene dalle seguenti relazioni:

$$A_n = 2^n \frac{1}{2} \sin(\theta_n) \quad , \quad A_{n+1} = 2^{n+1} \frac{1}{2} \sin(\theta_{n+1})$$

$$\theta_{n+1} = \frac{1}{2} \theta_n \quad , \quad \sin(\theta_n) = \frac{1 - \cos(\theta_n)}{2} \quad , \quad \cos(\theta_n) = \sqrt{1 - \sin^2(\theta_n)}$$

Si implementi poi la successione, diciamo $\{y_n\}$, che si ottiene razionalizzando, cioè moltiplicando numeratore e denominatore per

$$\sqrt{1 + \sqrt{1 - 4^{1-n} \cdot z_n^2}}$$

e si calcolino u_m , z_m e y_m per $m = 2, 3, \dots, 40$ (che teoricamente dovrebbero approssimare π). Si disegni in un unico grafico l'andamento dell'errore relativo di u_n , z_n e y_n rispetto a π .
Commentare i risultati ottenuti.

La prima successione può essere implementata in MatLab con il seguente codice.

```
% Numero di termini della successioni
nterm = 40;

% Pre-allocazione dello spazio in memoria
% per questioni di efficienza
u = repmat(NaN,1,nterm);
s = repmat(NaN,1,nterm);
z = repmat(NaN,1,nterm);
y = repmat(NaN,1,nterm);

% Successione 1
s(1) = 1;    u(1) = 1;
s(2) = 1.25; u(2) = s(2);
for n = 2:nterm-1
    s(n+1) = s(n)+(n+1)^(-2);
    u(n+1) = sqrt(6*s(n+1));
end
% Calcolo dell'errore relativo
rel_err_u = abs(u-pi)/pi;
```

La visualizzazione dei risultati può essere effettuata mediante i seguenti comandi MatLab:

```
% Visualizzazione degli errori relativi
h = semilogy(1:length(u),rel_err_u,'k. '); set(h,'LineWidth',2);
hold on;
h = semilogy(1:length(z),rel_err_z,'m+ '); set(h,'LineWidth',2);
h = semilogy(1:length(y),rel_err_y,'ro '); set(h,'LineWidth',2);
hold off
legend('u','z','y');
title('Successioni convergenti a pi');
xlabel('Iterazione');
```


Risultati

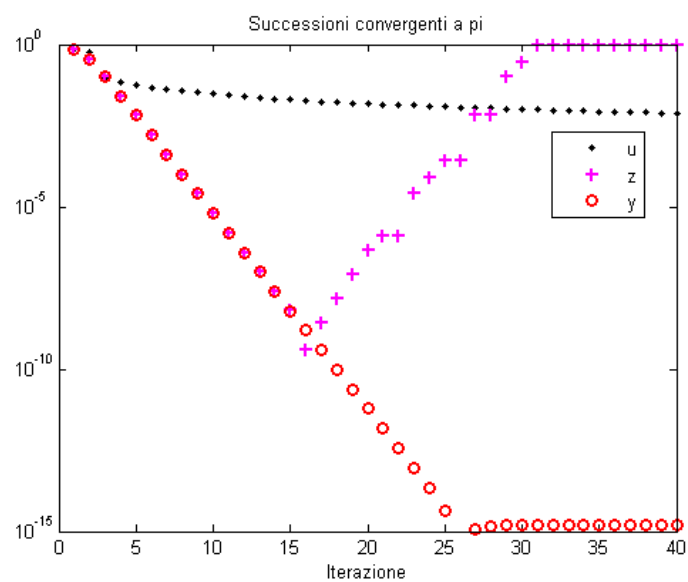


Figura 2: Risultati esercizio 4: Errori relativi.

Esercizio 5 (Corretta cottura di un uovo)¹

Quando un uovo viene fatto bollire le proteine coagulano e questa reazione avviene molto più velocemente all'aumentare della temperatura.

Nell'albume le proteine iniziano a coagulare a temperature superiori i 63°C , mentre nel tuorlo a temperature superiori i 70°C .

Per una cottura leggera, l'albume deve essere scaldato sufficientemente per coagulare alla temperatura di circa 63°C , mentre nella cottura più marcata il centro del tuorlo deve raggiungere i 70°C .

Per creare un modello matematico (semplice) del problema alcune semplificazioni devono essere adottate. Ad esempio, l'uovo viene considerato come una sfera omogenea di massa M e temperatura iniziale T_{egg} . Quando l'uovo viene messo a bollire in una pentola con acqua alla temperatura T_{wtr} sarà pronto quando la sua temperatura interna raggiungerà i $T_{\text{yolk}} = 63^\circ\text{C}$. Con queste ipotesi, il tempo di cottura t (in secondi) può essere dedotto risolvendo un'equazione di diffusione del calore che porta al seguente risultato:

$$t_{\text{cottura}} = \frac{M^{2/3} \cdot c \cdot \rho^{1/3}}{K \cdot \pi^2 \cdot (4\pi/3)^{2/3}} \cdot \log \left(0.76 \frac{T_{\text{egg}} - T_{\text{wtr}}}{T_{\text{yolk}} - T_{\text{wtr}}} \right)$$

dove ρ è la densità, c il calore specifico e K la conduttività termica dell'uovo.

Alcuni valori significativi possono essere i seguenti:

- $M = 47\text{ g}$ per un uovo di dimensioni piccole; $M = 67\text{ g}$ per un uovo di dimensioni grandi;
- $\rho = 1.038\text{ g cm}^{-3}$ per la densità;
- $c = 3.7\text{ J g}^{-1}\text{ K}^{-1}$ per il calore specifico;
- $K = 5.4 \cdot 10^{-3}\text{ W cm}^{-1}\text{ K}^{-1}$ per la conduttività termica;
- $T_{\text{wtr}} = 100^\circ\text{C}$ è la temperatura dell'acqua di cottura;
- $T_{\text{egg}} = 4^\circ\text{C}$ per un uovo proveniente dal frigorifero; $T_{\text{egg}} = 21^\circ\text{C}$ per un uovo a temperatura ambiente.

Si vuole analizzare al variare della massa dell'uovo la differenza tra i tempi di cottura per un uovo a temperatura da frigorifero ed un uovo a temperatura ambiente.

¹Facoltativo. Vettorizzazione di una formula in MatLab.

Risultati

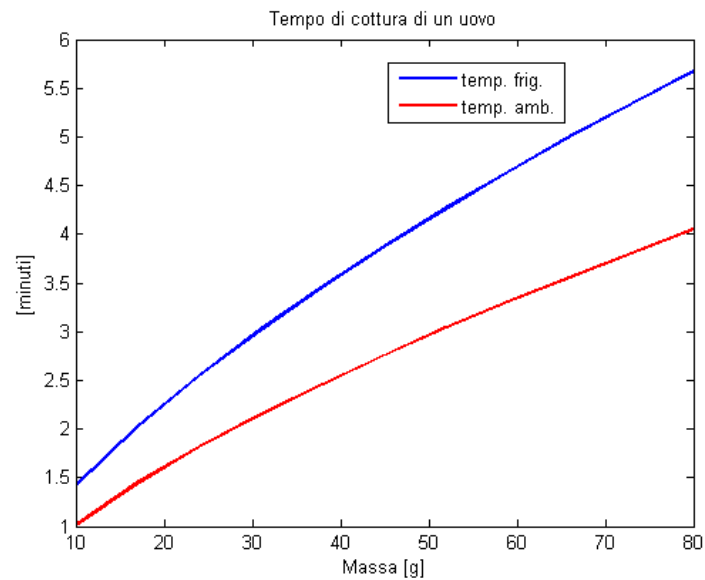


Figura 3: Risultati esercizio 5: Andamento dei tempi di cottura al variare della massa.

Probabilmente i tempi sono sottostimati dal momento che la coagulazione è un processo complicato che non avviene istantaneamente.

Esercizio 6 (Caduta di pressione in una tubazione idraulica)²

La caduta di pressione in una tubazione idraulica dove non sono presenti fenomeni di comprimibilità o inerzia del fluido viene modellata attraverso l'equazione di Darcy. Questa equazione lega la perdita di pressione con il quadrato della portata attraverso un coefficiente di attrito f . Questo fattore di attrito viene modellato in diversi modi a seconda del regime del fluido, da un valore costante per il regime laminare, dalla approssimazione di Haaland per il regime turbolento e da un'interpolazione lineare tra i due punti estremi dei due regimi. Come risultato di queste ipotesi, la perdita di carico in una tubazione viene simulata secondo le seguenti relazioni:

$$\Delta p = f \frac{L}{D_H} \frac{\rho}{2A^2} q|q|,$$

dove

$$f = \begin{cases} K_s / \text{Re} & \text{per } \text{Re} \leq \text{Re}_L \\ f_L + \frac{f_T - f_L}{\text{Re}_T - \text{Re}_L} (\text{Re} - \text{Re}_L) & \text{per } \text{Re}_L < \text{Re} < \text{Re}_T \\ \frac{1}{\left(-1.8 \log_{10} \left(\frac{6.9}{\text{Re}} + \left(\frac{r/D_H}{3.7} \right)^{1.11} \right) \right)^2} & \text{per } \text{Re} \geq \text{Re}_T \end{cases}$$

dove

- Δp è la caduta di pressione lungo la tubazione Pa;
- q è la portata del fluido $\text{m}^3 \text{s}^{-1}$;
- $\text{Re} = \frac{|q| \cdot D_H}{A \cdot \nu}$ è il numero di Reynolds;
- Re_L è il massimo numero di Reynolds per il regime laminare;
- Re_T è il massimo numero di Reynolds per il regime turbolento;
- K_s è il fattore di forma che caratterizza la sezione della tubazione;
- f_L è il fattore di frizione del regime laminare;
- f_T è il fattore di frizione del regime turbolento;
- A è l'area della sezione trasversale della tubazione m^2 ;
- D_H è il diametro idraulico m;
- L è la lunghezza della tubazione m;
- r è l'altezza della rugosità interna della tubazione m;

²Facoltativo. Vettorizzazione di una formula in MatLab. Può essere utile il comando `find`.

- ν è la viscosità cinematica $\text{m}^2 \text{s}^{-1}$.

Nel caso dell'acqua che scorre in una tubazione a sezione circolare di diametro D_H si hanno si possono assumere i seguenti valori caratteristici dei parametri:

- $\text{Re}_L = 2000$;
- $\text{Re}_T = 4000$;
- $K_s = 64$;
- $A = \frac{\pi \cdot D_H^2}{4} \text{ m}^2$;
- $r = 1.5 \cdot 10^{-5} \text{ m}$;
- $\nu = 1.004 \cdot 10^{-6} \text{ m}^2 \text{s}^{-1}$;
- $\rho = 1000 \cdot \text{kg m}^{-3}$;

I parametri f_L e f_T sono determinati garantendo la continuità delle equazioni.

Calcolare al variare della portata q da $-7/3600 \text{ m}^3 \text{s}^{-1}$ a $7/3600 \text{ m}^3 \text{s}^{-1}$ la caduta di pressione in una tubazione lunga $L = 6 \text{ m}$ e con diametro interno pari a $D_H = 0.1 \text{ m}$.

Nota

Il comando find può essere utilizzato nel seguente modo:

```
% Calcolo Reynolds
Re = abs(q)*Diam/(A*nu);

% Regime laminare
ind1 = find(Re <= ReL);
f(ind1) = Ks./Re(ind1);
```

Risultati

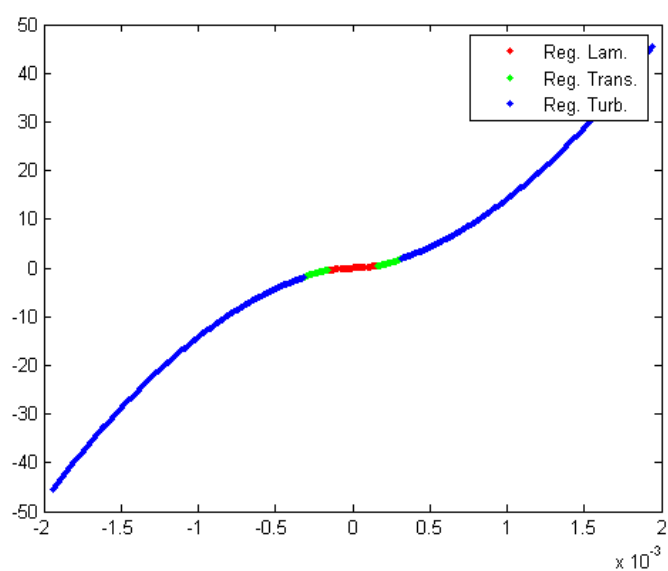


Figura 4: Caduta di pressione in una tubazione.