Introduzione a Matlab per Ingegneria dell'Energia
Esercizi con risoluzione [1]

A. Sommariva[2]

Abstract

Alcuni esercizi per familiarizzare con Matlab e loro correzione.

Revisione: 23 marzo 2020

---

1. Esercizio 1

E' noto che per $|x| < 1$ si ha

$$\frac{1}{1-x} \approx 1 + x + x^2 + x^3 + \ldots.$$

- Fissato $h = 0.01$ e i punti $x = -1+h, -1+2h, \ldots, 1-2h, 1-h$ si valutino in tali punti le funzioni

  $$\frac{1}{1-x}$$

  e

  $$1 + x + x^2 + x^3$$

  e si plottino i corrispettivi grafici in una stessa finestra.

  Suggerimento: usare i comandi Matlab `hold on` e `hold off` (aiutarsi con l'help di Matlab).

- Si plotti in scala semilogaritmica il grafico dell'errore assoluto $\left|\frac{1}{1-x} - (1 + x + x^2 + x^3)\right|$. Perchè quest'ultima quantità è particolarmente piccola per $x = 0$?

  Suggerimento: riguardo all'implementazione in Matlab, si faccia attenzione all'utilizzo delle operazioni puntuali di Matlab.

1.1. Risoluzione

```
h=0.01;
x=-1+h:h:1-h;
fx=1./(1-x);
gx=1+x+x.^2+x.^3;

plot(x,fx,'k-');
hold on;
plot(x,gx,'r-');
hold off;

pause;

semilogy(x,abs(fx-gx));
```

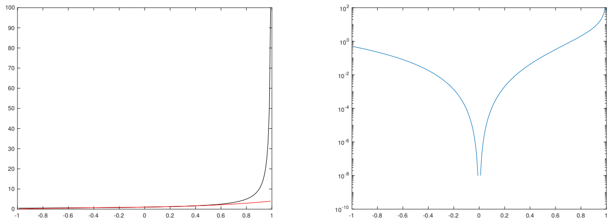

Figura 1: Grafico della funzione $f(x) = \frac{1}{1-x}$ (in nero), $p(x) = 1+x+x^2+x^3$ (in rosso), nell'intervallo $[-1+0.01, 1-0.01]$ ed errore assoluto, in scala semilogaritmica.

2. Esercizio 2

Eulero [1, p.47] ha dimostrato che tanto

$$\sum_{k=1}^{\infty} \frac{1}{k^2}$$

quanto

$$(\ln 2)^2 + \sum_{k=1}^{\infty} \frac{1}{k^2 \cdot 2^{k-1}}$$

convergono a $\frac{\pi^2}{6}$.

1. Calcolare per $N = 1000$ le somme

   $$\sum_{k=1}^{N} \frac{1}{k^2}$$

   e

   $$(\ln 2)^2 + \sum_{k=1}^{N} \frac{1}{k^2 \cdot 2^{k-1}}.$$

   Valutare

   $$\left|\sum_{k=1}^{N} \frac{1}{k^2} - \frac{\pi^2}{6}\right|$$

   e

   $$\left|(\ln 2)^2 + \sum_{k=1}^{N} \frac{1}{k^2 \cdot 2^{k-1}} - \frac{\pi^2}{6}\right|.$$

   Per effettuare la somma usare il comando `sum`.

2. Stampare il valore $\pi^2/6$, le due somme calcolate e gli errori assoluti (sempre con 1 cifra prima, 15 dopo la virgola, in notazione esponenziale).

2.1. Risoluzione

Salviamo nel file `esercizioA2.m`:

```
% esercizio 2

N=1000;

k=1:1:1000;

s1=1./(k.^2); % argomenti da sommare.
S1=sum(s1);   % somma.
err1=abs(pi^2/6-S1); % errore assoluto

s2=1./( (k.^2) .* (2.^(k-1)) ); % argomenti da sommare.
S2=(log(2))^2+sum(s2);   % somma
err2=abs(pi^2/6-S2); % errore assoluto

% Stampa risultati.

fprintf('\n SOMMA : %1.15e',pi^2/6);
fprintf('\n VAL I : %1.15e',S1);
fprintf('\n VAL II: %1.15e',S2);
fprintf('\n ');
fprintf('\n EA I : %1.15e',err1);
fprintf('\n EA II: %1.15e',err2);
fprintf('\n \n');
```

e otteniamo come risultati

```
>> esercizioA2

 SOMMA : 1.644934066848226e+00
 VAL I : 1.643934566681561e+00
 VAL II: 1.644934066848226e+00

 EA I : 9.995001666649461e-04
 EA II: 2.220446049250313e-16

>>
```

Il risultato mostra che la seconda delle due somme converge molto più rapidamente della prima a $\pi^2/6$.

3. Esercizio 3

Il fattoriale di un numero n, è $n! = 1 \cdot 2 \ldots n$.
Si scriva

1. una function `fattoriale_gamma` che utilizzi la funzione Matlab `gamma` che valuta la funzione Γ, e che la proprietà
$$\Gamma(n) = (n-1)!$$
2. una funzione `fattoriale` che calcoli il prodotto $1 \cdot 2 \ldots n$ mediante il comando Matlab `prod` (aiutarsi con l'help),
3. una funzione `esercizioA3.m` che utilizzi le funzioni `fattoriale_gamma`, `fattoriale` per calcolare il valore del fattoriale di 20.

3.1. Risoluzione

Facciamo entrambi i codici, sia con la funzione Γ che col prodotto $1 \cdot 2 \ldots \cdot n$.

Il primo è :

```
function y=fattoriale_gamma (n)

y=gamma(n+1);
```

mentre per il secondo, utilizzando la funzione Matlab `prod` (si veda l'help di Matlab):

```
function y=fattoriale (n)

v=1:1:n;
y=prod(v);
```

Per controllare che siano coincidenti, scriviamo il seguente file, che salviamo in `esercizioA3.m`:

```
n=20;

y1=fattoriale_gamma(n);
y2=fattoriale(n);

fprintf('\n n           : %3.0f',n);
fprintf('\n fatt. gamma: %1.50g',y1);
fprintf('\n fatt. prod.: %1.50g',y2);
fprintf('\n \n');
```

Dalla command window, otteniamo:

```
>> esercizioA3

 n          :  20
 fatt. gamma: 2432902008176640000
 fatt. prod.: 2432902008176640000

>>
```

4. Esercizio 4

Si implementi una funzione `fattoriale_for` che calcola il fattoriale *fatt* di un numero naturale n mediante un ciclo `for`. Di seguito si calcoli nella command window il fattoriale di $n = 20$, utilizzando tale routine.

4.1. Risoluzione

Salviamo il seguente codice in `fattoriale_for.m`

```
function y=fattoriale_for(n)

y=1;
for i=1:n
    y=y*i;
end
```

e quindi nella command window,

```
>> y=fattoriale_for(20);
>> fprintf('\n \t %1.50g \n',y)
   2432902008176640000
>>
```

Si osservi che il risultato coincide coi precedenti.

5. Esercizio 5

La posizione di un punto P nello spazio tridimensionale può essere rappresentata da una terna del tipo (x, y, z), dove x, y e z sono le componenti lungo i tre assi cartesiani. Se due punti P_1 e P_2 sono rappresentati dai valori (x_1, y_1, z_1) e (x_2, y_2, z_2),

1. scrivere una function `distanza` che calcoli la distanza euclidea tra questi due punti, cioè
$$d(P_1, P_2) := \sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2)};$$
2. calcolare analiticamente e mediante la function `distanza`, $d(P_1, P_2)$ qualora
$$P_1 = (1, 1, 1), \;\; P_2 = (3, 4, 3).$$

5.1. Risoluzione

Scriviamo la function `distanza`

```
function d=distanza(P1,P2)

d=sqrt(sum((P1-P2).^2));

% --- codice equivalente ---
% d=0;
% for ii=1:3
%     d=d+(P1(ii)-P2(ii))^2; % operazione tra scalari.
% end
% d=sqrt(d);
```

e dalla command window

```
>> P1=[1 1 1];
>> P2=[3 4 3];
>> d=distanza(P1,P2)
d =
   4.123105625617661
>>
```

In effetti

$$\begin{aligned} d &= \sqrt{(1-3)^2 + (1-4)^2 + (1-3)^2} \\ &= \sqrt{17} \approx 4.123105625617661. \end{aligned}$$

6. Esercizio 6

Dato un triangolo rettangolo con un angolo $\theta \in (0, \pi/2)$ e ipotenusa $r > 0$,

1. definire la function `cateti` che calcola la lunghezza dei suoi cateti,
2. calcolare analiticamente e mediante `cateti` la lunghezza dei cateti di un triangolo in cui $\theta = \pi/6$ e $r = 1$.

6.1. Risoluzione

Per risolvere l'esercizio serve un po' di trigonometria e ricordare che i cateti hanno rispettivamente lunghezza $r \cdot \cos(\theta)$, $r \cdot \sin(\theta)$.

Quindi salviamo in `cateti.m`

```
function [cateto1,cateto2]=cateti(r,theta)

cateto1=r*cos(theta);
cateto2=r*sin(theta);
```

e da command window

```
>> [cateto1,cateto2]=cateti(1,pi/6);
>> cateto1
cateto1 =
   0.866025403784439
>>
>> cateto2
cateto2 =
   0.500000000000000
>>
```

In effetti sappiamo che i cateti sono lunghi

$$\cos(\pi/3) = \sqrt{3}/2 \approx 0.866025403784439,$$
$$\sin(\pi/3) = 0.5.$$

7. Esercizio 7

Un numero naturale n è *primo* se e solo se è divisibile solo per se stesso e per 1.

Il comando Matlab `primes`, applicato a un numero naturale n, fornisce un vettore contenente tutti i numeri primi inferiori o uguali a n.

La cardinalità dei numeri primi inferiori o uguali a n è quindi `length(primes(n))`.

Esistono due *stime* di tale quantità (cf. [2]).

La prima è $n/\log(n)$ (qui log è il logaritmo in base e) e la seconda è il logaritmo integrale, implementato nella routine Matlab `logint`.

Quale esercizio, si scriva una funzione `demo_primes` che

- assegni a `nmax` il valore 1000,
- per ogni numero naturale `n` da 1 a `nmax`
  1. calcoli la cardinalità dei numeri primi inferiori o uguali a n e la immagazzini nella n-sima componente del vettore `card_primi`;
  2. calcoli $n/log(n)$ e la immagazzini nella n-sima componente del vettore `approssimazione1`;
  3. calcoli `logint(n)` e la immagazzini nella n-sima componente del vettore `approssimazione2`;
- esegua il grafico delle coppie
  1. (k,`card_primi(k)`),
  2. (k,`approssimazione1(k)`),

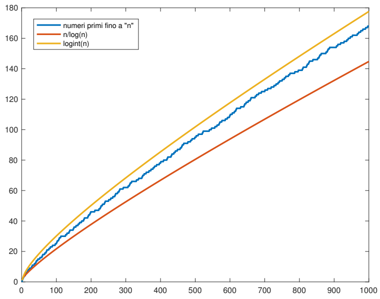

Figura 2: Numeri primi più piccoli di un naturale n e due sue approssimazioni.

3. (k,`approssimazione2(k)`)),

per $k = 1, \ldots,$`nmax`, e una legenda avente per stringhe

- `'numeri primi fino a "n"'`,
- `'n/log(n)'`,
- `'logint(n)'`.

Risulta più accurata la prima approssimazione oppure la seconda?

Nota. La routine può essere un po' lenta, visto che il calcolo di `logint(n)` non è rapidissimo.

7.1. *Risoluzione*

Salviamo il seguente codice in `demo_primes`.

```
function demo_primes

% Oggetto:
% Numeri primi piu' piccoli di "n" prefissato e alcune ...
    stime matematiche.
%
% Si consulti per dettagli:
% https://it.wikipedia.org/wiki/Teorema_dei_numeri_primi

nmax=1000;

for n=1:nmax
    % cardinalita' dei numeri primi piu' piccoli di "n"
    card_primi(n)=length(primes(n));
    % una stima di tale cardinalita'
    approssimazione1(n)=n/log(n);
    % stima piu' precisa di questa cardinalita'.
    approssimazione2(n)=logint(n); % logaritmo integrale...
     .
end

% ------------- plot  -------------
indici=1:nmax;
plot(indici,card_primi,indici,approssimazione1,...
    indici,approssimazione2,'Linewidth',2);
legend('numeri primi fino a "n"', 'n/log(n)','logint(n)'...
    );
```

Lanciatolo nella command-window otteniamo il grafico in figura.

Il logaritmo integrale offre una stima molto precisa di tale quantità . Ad esempio:

```
>> format long g
>> L=length(primes(10000))
L =
        1229
>> logint(10000)
ans =
          1246.13721589939
>> 10000/log(10000)
ans =
          1085.73620475813
>>
```

8. Esercizio 8

Si calcolino $a^2 - b^2$ per $a = 1.4 \cdot 10^{154}$ e $b = 1.3 \cdot 10^{154}$. E' uguale il valore numerico di $a^2 - b^2$ con quello di $(a-b) \cdot (a+b)$?

8.1. *Risoluzione*

Scriviamo quanto segue nella command-window.

```
>> a=1.4*10^(154); b=1.3*10^(154);
>> % primo valore
>> val1=a^2-b^2
val1 =
   Inf
>> % secondo valore
>> val2=(a-b)*(a+b)
val2 =
  2.7000e+307
>> % motivo per cui il primo valore non e' esatto
>> realmax
ans =
  1.7977e+308
>> a^2
ans =
   Inf
>> b^2
ans =
  1.6900e+308
>>
```

9. Esercizio 9

Valutare in Matlab la funzione

$$\frac{\exp(x) - 1}{x}$$

nelle ascisse 10^{-k}, $k = 1, \ldots, 16$.

Si domanda se i risultati sono coerenti col fatto che

$$\lim_{x \to 0} \frac{\exp(x) - 1}{x} = 1.$$

9.1. *Svolgimento*

Utilizziamo il fatto che Matlab può valutare una funzione per ogni componente di un vettore.

Da command-window

```
>> k=0:1:16;
>> x=10.^(-k);
>> f=@(x) (exp(x)-1)./x;
>> y=feval(f,x);
>> % errore rispetto al limite previsto
>> % Trasponiamo i vettori, perche' sono riga.
>> format short e
>> [x' abs(y-1)']
ans =
   1.0000e+00   7.1828e-01
   1.0000e-01   5.1709e-02
   1.0000e-02   5.0167e-03
   1.0000e-03   5.0017e-04
   1.0000e-04   5.0002e-05
   1.0000e-05   5.0000e-06
   1.0000e-06   4.9996e-07
   1.0000e-07   4.9434e-08
   1.0000e-08   6.0775e-09
   1.0000e-09   8.2740e-08
   1.0000e-10   8.2740e-08
   1.0000e-11   8.2740e-08
   1.0000e-12   8.8901e-05
   1.0000e-13   7.9928e-04
   1.0000e-14   7.9928e-04
   1.0000e-15   1.1022e-01
   1.0000e-16   1.0000e+00
>>
```

Bibliografia

[1] W. Dunham, *Euler, The Master of Us All*, The Mathematical Association of America, Dolciani Mathematical Expositions No 22, 1999.

[2] Wikipedia, Teorema dei numeri primi
https://it.wikipedia.org/wiki/Teorema_dei_numeri_primi