II esercitazione di
Laboratorio Computazionale Numerico

1. a) Sia $x = [-3, 5, 8, 0, 1, 5, -2, 4]$

- imporre il 6° elemento uguale a 100
- imporre il 1°, 2°, 3° elemento uguali rispettivamente a 5, 6, 7
- togliere il 4° elemento
- togliere con un solo comando dal 4° al 7° elemento compresi
- aggiungere in testa 1, 2, 3
- aggiungere in coda 10, 11, 12.

b) Sia A la matrice identità di dimensione 4x4

- sostituire all'elemento (1,1) l'elemento (3,4)
- aggiungere una colonna di elementi uguali ad 1 in testa
- aggiungere una colonna di elementi uguali ad 1 in coda
- aggiungere una riga di elementi uguali ad 4 in testa
- aggiungere una riga di elementi uguali ad 4 in coda
- togliere la 3a riga
- togliere la 3a colonna

2. *Dopo aver definito il vettore $x = [1 : -0.1 : 0]$ spiegare il significato dei seguenti comandi Matlab:*

```
>> x([1 4 3]);
>> x([1:2:7 10])=zeros(1,5);
>> x([1 2 5])=[0.5*ones(1,2) -0.3];
>> y=x(end:-1:1);
```

3. *Usare le variabili e le operazioni vettoriali per osservare la convergenza in $\mathbb{N}$ delle successioni*

$$\left(1+\frac{1}{n}\right)^n \to e\,, \qquad \frac{4n}{n+2} \to 4\,, \qquad \log\left(1+\sqrt{\frac{n}{n+1}}\right) \to \log 2\,.$$

4. Osservare la convergenza nel calcolo dei limiti delle seguenti funzioni

- $x \cdot (\sqrt{(x^2+1)} - x)$
- $x \cdot \sqrt{(x^2+1)} - x^2$
- $x/(\sqrt{(x^2+1)} + x)$

5. *Utilizzare il comando* `diag` *per generare la matrice tridiagonale A di dimensione 9×9 i cui elementi della diagonale principale coincidono con -2 e quelli delle codiagonali con 1. Successivamente scambiare in A dapprima le righe 3 e 6, e di seguito, le colonne 1 e 4.*

6. *Definire la matrice*

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

e comprendere il significato dei seguenti comandi Matlab:

```
>> size(A);
>> B=A.*A;
>> B=A*A;
>> B=A'*A;
>> A(1:2,4),A(:,3),A(1:2,:),A(:,[2 4]),A([2 3 3]);
>> A(3,2)=A(1,1);
>> A(1:2,4)=zeros(2,1);
>> A(2,:)=A(2,:)-A(2,1)/A(1,1)*A(1,:);
```

7. *Definire la matrice*

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 \\ 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \end{bmatrix}$$

e successivamente:

a) generare le matrici S triangolare superiore e I triangolare inferiore i cui elementi non nulli coincidano con gli elementi omonimi di A; successivamente, porre tutti gli elementi della diagonale principale della matrice S uguali a 0 e quelli della matrice I uguali a 1;

b) generare le matrici B_1, B_2 e B_3 rispettivamente tridiagonale, bidiagonale superiore e bidiagonale inferiore, i cui elementi coincidano con gli elementi omonimi di A.

8. Al variare del parametro $p = 10^\alpha$, con $\alpha = 1:10$, calcolare mediante le note formula risolutive, le radici dell' equazione di quarto grado

$$x^4 - b\,x^2 + 1 = 0,$$

con $b = \frac{1+p^2}{p}$. In seguito, tradurre tali formule in istruzioni di assegnazione Matlab in una funzione matlab che ha α come parametro di input e le 4 soluzioni come output. Predisporre una tabella con gli errori relativi commessi da Matlab nel calcolo numerico delle radici dell'equazione assegnata. Motivare i risultati ottenuti.

9. a)Dopo aver visualizzato in Matlab il valore `realmin` $\simeq 10^{-m}$, predisporre e visualizzare in `format long` e un vettore di 21 elementi logaritmicamente equispaziati fra 10^{-m} e 10^{-m-20}. Commentare i risultati ottenuti.

b) Visualizzare i valori: `realmax`, `realmax` $\cdot\, 10$, `realmax` $\cdot\, (1 + eps)$

c) Visualizzare i valori: $1 + 10^{-17}$, $1 + 10^{17}$.

```
% Esercizio 1 - Esercitazione 2
clear all
close all
clc
% a
x=[-3 5 8 0 1 5 -2 4]
x(6)=100
x(1:3)=[5 6 7]
x(4)=[]
x(4:7)=[]
x=[1 2 3 x]
x=[x 10 11 12]
% b)
A=eye(4)
A(1,1)=A(3,4)
x=ones(4,1)
A=[x,A]
A=[A,x]
x=4*ones(1,6)
A=[x;A]
A=[A;x]
A(3,:)=[]
A(:,3)=[]
```

COMMAND WINDOW

```
x =
   -3     5     8     0     1     5    -2     4
x =
   -3     5     8     0     1   100    -2     4
x =
    5     6     7     0     1   100    -2     4
x =
    5     6     7     1   100    -2     4
x =
    5     6     7
x =
    1     2     3     5     6     7
x =
    1     2     3     5     6     7    10    11    12
A =

    1     0     0     0
    0     1     0     0
    0     0     1     0
    0     0     0     1

A =

    0     0     0     0
    0     1     0     0
    0     0     1     0
    0     0     0     1

x =

    1
    1
    1
    1

A =

    1     0     0     0     0
    1     0     1     0     0
    1     0     0     1     0
    1     0     0     0     1

A =

    1     0     0     0     0     1
    1     0     1     0     0     1
    1     0     0     1     0     1
    1     0     0     0     1     1

x =

    4     4     4     4     4     4

A =

    4     4     4     4     4     4
    1     0     0     0     0     1
    1     0     1     0     0     1
    1     0     0     1     0     1
    1     0     0     0     1     1

A =

    4     4     4     4     4     4
    1     0     0     0     0     1
    1     0     1     0     0     1
    1     0     0     1     0     1
    1     0     0     0     1     1
    4     4     4     4     4     4

A =

    4     4     4     4     4     4
    1     0     0     0     0     1
    1     0     0     1     0     1
    1     0     0     0     1     1
    4     4     4     4     4     4

A =

    4     4     4     4     4
    1     0     0     0     1
    1     0     1     0     1
    1     0     0     1     1
    4     4     4     4     4
```

```
% Esercizio 2 - Esercitazione 2
clear all
close all
clc

x=[1:-0.1:0]

x([1 4 3])
% selezione del primo, quarto e terzo elemento.

x_1=x;
x_1([1:2:7 10])=zeros(1,5)
% gli elementi dal primo fino al settimo intervallando di un passo = 2 e il
% decimo elemento uguali a zero. visto che gli elementi selezionati sono 5,
% la funzione zeros con (1,5) parametri costruisce una matrice 1x5
% (o vettore riga) con 5 elementi di valore 0 e quindi compatibile
% con l'assegnamento voluto.

x_2=x;
x_2([1 2 5])=[0.5*ones(1,2) -0.3]
% sostituisce il primo e il secondo elemento con un vettore riga di 2
% elementi di valore 0.5 e il quinto elemento con il numero -0.3.

y=x(end:-1:1)
% costruisce il vettore y con gli elementi di x invertendo l'ordine dei
% valori, cioè effettua una solezione degli elementi di x iniziando
% dall'ultimo e deincrementando la posizione di 1, fino al valore 1.
```

COMMAND WINDOW

```
x =

  Columns 1 through 5

   1.000000000000000   0.900000000000000   0.800000000000000   0.700000000000000   0.600000000000000
    Columns 6 through 10

 0.500000000000000      0.400000000000000   0.300000000000000   0.200000000000000   0.100000000000000
    Column 11
   0

 ans =
   1.000000000000000   0.700000000000000   0.800000000000000
 x_1 =
   Columns 1 through 6
                   0   0.900000000000000                   0   0.700000000000000                   0   0.500000000000000
  Columns 7 through 11
                   0   0.300000000000000   0.200000000000000                   0                   0
 x_2 =

  Columns 1 through 6

  0.500000000000000   0.500000000000000   0.800000000000000   0.700000000000000  -0.300000000000000
 0.500000000000000
  Columns 7 through 11
   0.400000000000000   0.300000000000000   0.200000000000000   0.100000000000000                   0

 y =
  Columns 1 through 6

   0   0.100000000000000   0.200000000000000   0.300000000000000   0.400000000000000   0.500000000000000
  Columns 7 through 11

   0.600000000000000   0.700000000000000   0.800000000000000   0.900000000000000   1.000000000000000
```

```
% Esercizio 3 - Esercitazione 2
diary esercizio_3.txt
clear all
close all
clc
format long

n=logspace(1,20,10)
% Convergenza (1+(1/n))^n -> exp(1)
y=(1+(1./n)).^n;

% Convergenza (4*n)/(n+2) -> 4
y_1=(4.*n)./(n+2);

% Convergenza log(1+sqrt(n/(n+1))) ->log(2)
y_2=log(1+sqrt(n./(n+1)));

tab_res=[y' y_1' y_2']

% la successione y converge a 2.71 (numero di nepero), ma quando la n è
% sufficiente grande , 1/n viene approssimato dalla macchina a 0,
% risultando quindi 1.
% la successione y_1 converge a 4
% la successione y_2 converge a 0.69( log(2) )
format short
```

COMMAND WINDOW

```
n =

   1.0e+20 *

  Columns 1 through 6

   0.000000000000000   0.000000000000000   0.000000000000002   0.000000000000215   0.000000000027826
0.000000003593814

  Columns 7 through 10

   0.000000464158883   0.000059948425032   0.007742636826811   1.000000000000000


tab_res =

   2.593742460100002   3.333333333333334   0.669603484700277
   2.717230241342687   3.993815467456437   0.692953708255537
   2.718273680717571   3.999952041834978   0.693145681854935
   2.718281759981242   3.999999628672928   0.693147168955974
   2.718282184502655   3.999999997124949   0.693147180470100
   2.718383024182343   3.999999999977740   0.693147180559250
   2.717522354266583   3.999999999999828   0.693147180559940
   3.785289746477336   3.999999999999999   0.693147180559945
   1.000000000000000   4.000000000000000   0.693147180559945
   1.000000000000000   4.000000000000000   0.693147180559945
```

```
% Esercizio 4 - Esercitazione 2
diary esercizio_4.txt
clear all
close all
clc
format long

x=logspace(1,10,10);
% Funzione x*(sqrt((x^2)+1)-x)
y=x.*(sqrt((x.^2)+1)-x);

% Funzione x.*sqrt((x.^2)+1)-(x.^2)
y_1=x.*sqrt((x.^2)+1)-(x.^2);

% funzione x./(sqrt((x.^2)+1)+x)
y_2=x./(sqrt((x.^2)+1)+x);

tab_res=[y' y_1' y_2']

format short
```

COMMAND WINDOW

```
tab_res =

   0.498756211208899   0.498756211208899   0.498756211208903
   0.499987500624854   0.499987500625139   0.499987500624961
   0.499999875046342   0.499999875086360   0.499999875000062
   0.500000005558832   0.500000000000000   0.499999998750000
   0.499999441672117   0.500000000000000   0.499999999987500
   0.500003807246685   0.500000000000000   0.499999999999875
   0.502914190292358   0.500000000000000   0.499999999999999
                   0                   0   0.500000000000000
                   0                   0   0.500000000000000
                   0                   0   0.500000000000000
```

```
% Esercizio 5 - Esercitazione 2
diary esercizio_5.txt
clear all
close all
clc

A=diag(-2*ones(1,9),0)+diag(ones(1,8),-1)+diag(ones(1,8),1)
% scambio le righe 3 e 6 e poi le colonne 1 e 4
tmp=A(3,:);
A(3,:)=A(6,:);
A(6,:)=tmp;
tmp=A(:,1);
A(:,1)=A(:,4);
A(:,4)=tmp
```

COMMAND WINDOW

```
A =

    -2     1     0     0     0     0     0     0     0
     1    -2     1     0     0     0     0     0     0
     0     1    -2     1     0     0     0     0     0
     0     0     1    -2     1     0     0     0     0
     0     0     0     1    -2     1     0     0     0
     0     0     0     0     1    -2     1     0     0
     0     0     0     0     0     1    -2     1     0
     0     0     0     0     0     0     1    -2     1
     0     0     0     0     0     0     0     1    -2


A =

     0     1     0    -2     0     0     0     0     0
     0    -2     1     1     0     0     0     0     0
     0     0     0     0     1    -2     1     0     0
    -2     0     1     0     1     0     0     0     0
     1     0     0     0    -2     1     0     0     0
     1     1    -2     0     0     0     0     0     0
     0     0     0     0     0     1    -2     1     0
     0     0     0     0     0     0     1    -2     1
     0     0     0     0     0     0     0     1    -2
```

```
% Esercizio 6 - Esercitazione 2
diary esercizio_6.txt
clear all
close all
clc

A=[1 2 3 4;5 6 7 8;9 10 11 12];

size(A)
% restituisce un vettore di 2 elementi che come primo ha il numero di righe
% della matrice passata come parametro alla funzione size e come secondo i
% numero di colonne
B=A.*A
% assegna alla matrice B il prodotto elemento per elemento della matrice A
% con se stessa
%B=A*A
% Error using  * Inner matrix dimensions must agree.
% * equivale al prodotto righe per colonne, l'errore dato dal fatto che il
% numero di colonne della prima matrice non è uguale al numero di righe
% della seconda matrice
B_1=A'*A
% assegna a B il prodotto righe per colonne della trasposta della matrice A
% (4x3) e la matrice A(3x4), il numero di colonne della prima matrice
% corrisponde al numero di righe della seconda matrice quindi è possibile
% effetturare la moltiplicazione righe per colonne.
A(1:2,4),A(:,3),A(1:2,:),A(:,[2 4]),A([2 3 3])
% seleziona gli elementi sulla prima e seconda riga, della quarta colonna.
% seleziona tutta la terza colonna.
% seleziona tutta la prima e seconda riga.
% seleziona tutti gli elementi della seconda e quarta colonna
% seleziona il secondo, il terzo(due volte) elemento.
A_1=A;
A_1(3,2)=A(1,1)
% assegna al elemento in terza riga e seconda colonna il primo elemento
% della matrice.
A_2=A;
A_2(1:2,4)=zeros(2,1)
% assegna agli elementi della prima e seconda riga, nell'ultima colonna il
% valore 0
A_3=A;
A_3(2,:)=A(2,:)-A(2,1)/A(1,1)*A(1,:)
% assegna alla seconda riga la differenza tra la seconda riga e il rapporto
% del elemento nella seconda riga e prima colonna (2) con il prodotto del
% primo elemento della matrice (1) con la prima riga (1 2 3 4).
```

COMMAND WINDOW

```
ans =

     3     4


B =

     1     4     9    16
    25    36    49    64
    81   100   121   144

B_1 =

   107   122   137   152
   122   140   158   176
   137   158   179   200
   152   176   200   224


ans =

     4
     8


ans =

     3
     7
    11


ans =

     1     2     3     4
     5     6     7     8


ans =

     2     4
     6     8
    10    12


ans =

     5     9     9


A_1 =

     1     2     3     4
     5     6     7     8
     9     1    11    12

A_2 =

     1     2     3     0
     5     6     7     0
     9    10    11    12


A_3 =

     1     2     3     4
     0    -4    -8   -12
     9    10    11    12
```

```
% Esercizio 7 - Esercitazione 2
diary esercizio_7.txt
clear all
close all
clc

A=[1:1:8]';
A=[A A A A A A A A]

%a)
S=triu(A)
I=tril(A)
S=triu(A,1)
I=tril(A,-1)+eye(8)
%b)
B_1=diag(diag(A))+diag(diag(A,1),1)+diag(diag(A,-1),-1)
B_2=diag(diag(A))+diag(diag(A,1),1)
B_3=diag(diag(A))+diag(diag(A,-1),-1)
```

COMMAND WINDOW

```
A =

     1     1     1     1     1     1     1     1
     2     2     2     2     2     2     2     2
     3     3     3     3     3     3     3     3
     4     4     4     4     4     4     4     4
     5     5     5     5     5     5     5     5
     6     6     6     6     6     6     6     6
     7     7     7     7     7     7     7     7
     8     8     8     8     8     8     8     8

S =

     1     1     1     1     1     1     1     1
     0     2     2     2     2     2     2     2
     0     0     3     3     3     3     3     3
     0     0     0     4     4     4     4     4
     0     0     0     0     5     5     5     5
     0     0     0     0     0     6     6     6
     0     0     0     0     0     0     7     7
     0     0     0     0     0     0     0     8

I =

     1     0     0     0     0     0     0     0
     2     2     0     0     0     0     0     0
     3     3     3     0     0     0     0     0
     4     4     4     4     0     0     0     0
     5     5     5     5     5     0     0     0
     6     6     6     6     6     6     0     0
     7     7     7     7     7     7     7     0
     8     8     8     8     8     8     8     8

S =

     0     1     1     1     1     1     1     1
     0     0     2     2     2     2     2     2
     0     0     0     3     3     3     3     3
     0     0     0     0     4     4     4     4
     0     0     0     0     0     5     5     5
     0     0     0     0     0     0     6     6
     0     0     0     0     0     0     0     7
     0     0     0     0     0     0     0     0

I =

     1     0     0     0     0     0     0     0
     2     1     0     0     0     0     0     0
     3     3     1     0     0     0     0     0
     4     4     4     1     0     0     0     0
     5     5     5     5     1     0     0     0
     6     6     6     6     6     1     0     0
     7     7     7     7     7     7     1     0
     8     8     8     8     8     8     8     1

B_1 =

     1     1     0     0     0     0     0     0
     2     2     2     0     0     0     0     0
     0     3     3     3     0     0     0     0
     0     0     4     4     4     0     0     0
     0     0     0     5     5     5     0     0
     0     0     0     0     6     6     6     0
     0     0     0     0     0     7     7     7
     0     0     0     0     0     0     8     8

B_2 =

     1     1     0     0     0     0     0     0
     0     2     2     0     0     0     0     0
     0     0     3     3     0     0     0     0
     0     0     0     4     4     0     0     0
     0     0     0     0     5     5     0     0
     0     0     0     0     0     6     6     0
     0     0     0     0     0     0     7     7
     0     0     0     0     0     0     0     8

B_3 =

     1     0     0     0     0     0     0     0
     2     2     0     0     0     0     0     0
     0     3     3     0     0     0     0     0
     0     0     4     4     0     0     0     0
     0     0     0     5     5     0     0     0
     0     0     0     0     6     6     0     0
     0     0     0     0     0     7     7     0
     0     0     0     0     0     0     8     8
```

```
% Esercizio 8 - Esercitazione 2
diary esercizio_8.txt
clear all
close all
clc
format long

alfa=[1:1:10];

% Abbiamo calcolato in modo analitico le radici dell'equazioni
ve1=sqrt(10.^alfa);
ve2=-sqrt(10.^alfa);
ve3=sqrt(1./(10.^alfa));
ve4=-sqrt(1./(10.^alfa));

mat_esa=[ve1' ve2' ve3' ve4']

% chiamata alla funzione RadEq4Gr
[v1,v2,v3,v4]=RadEq4Gr(alfa);
mat_rad_f=[v1' v2' v3' v4']

% tabella errore relativi commessi nel calcolo delle radici dell'equazione.
% calcoliamo i valori con la funzione RadEq4Gr e poi confrontiamo
% con le radici ottenute dai calcoli da noi effettuati mediante le formule
% note per poi determinare l'errore relativo.
Err_Ass=abs(mat_esa-mat_rad_f);
Err_Rel=Err_Ass./mat_esa
% Possiamo osservare che l'errore relativo aumenta con l'aumentare dei
% coefficienti del polinomio

format short
```

FUNZIONE MATLAB

```
function [x_1, x_2, x_3, x_4] = RadEq4Gr( alfa )
% Funzione che calcola le radici di un equazione di quarto grado

p=(10).^alfa;

% (x^4)-b*(x^2)+1=0
vb=(1+p.^2)./p;

a=1;
b=-vb;
c=1;
delta=(b.^2)-(4*a*c);
x_1=sqrt((-b+sqrt(delta))/(2*a));
x_2=-sqrt((-b+sqrt(delta))/(2*a));
x_3=sqrt((-b-sqrt(delta))/(2*a));
x_4=-sqrt((-b-sqrt(delta))/(2*a));


end
```

COMMAND WINDOW   Esercizio 8 - Esercitazione 2

```
mat_esa =

   1.0e+05 *

   0.000031622776602  -0.000031622776602   0.000003162277660  -0.000003162277660
   0.000100000000000  -0.000100000000000   0.000001000000000  -0.000001000000000
   0.000316227766017  -0.000316227766017   0.000000316227766  -0.000000316227766
   0.001000000000000  -0.001000000000000   0.000000100000000  -0.000000100000000
   0.003162277660168  -0.003162277660168   0.000000031622777  -0.000000031622777
   0.010000000000000  -0.010000000000000   0.000000010000000  -0.000000010000000
   0.031622776601684  -0.031622776601684   0.000000003162278  -0.000000003162278
   0.100000000000000  -0.100000000000000   0.000000001000000  -0.000000001000000
   0.316227766016838  -0.316227766016838   0.000000000316228  -0.000000000316228
   1.000000000000000  -1.000000000000000   0.000000000100000  -0.000000000100000


mat_rad_f =

   1.0e+05 *

   0.000031622776602  -0.000031622776602   0.000003162277660  -0.000003162277660
   0.000100000000000  -0.000100000000000   0.000001000000000  -0.000001000000000
   0.000316227766017  -0.000316227766017   0.000000316227766  -0.000000316227766
   0.001000000000000  -0.001000000000000   0.000000100000000  -0.000000100000000
   0.003162277660168  -0.003162277660168   0.000000031622782  -0.000000031622782
   0.010000000000000  -0.010000000000000   0.000000010000038  -0.000000010000038
   0.031622776601684  -0.031622776601684   0.000000003156763  -0.000000003156763
   0.100000000000000  -0.100000000000000   0.000000000863167  -0.000000000863167
   0.316227766016838  -0.316227766016838                   0                   0
   1.000000000000000  -1.000000000000000                   0                   0


Err_Rel =

                   0                   0   0.000000000000002  -0.000000000000002
                   0                   0   0.000000000000100  -0.000000000000100
                   0                   0   0.000000000011823  -0.000000000011823
                   0                   0   0.000000001011358  -0.000000001011358
                   0                   0   0.000000169267864  -0.000000169267864
                   0                   0   0.000003807239438  -0.000003807239438
                   0                   0   0.001743943249748  -0.001743943249748
                   0                   0   0.136832542496890  -0.136832542496890
                   0                   0   1.000000000000000  -1.000000000000000
                   0                   0   1.000000000000000  -1.000000000000000
```

```
% Esercizio 9 - Esercitazione 2
clear all
close all
clc
diary esercizio_9.txt
% a)

% funzione che ritorna il piu piccolo numero rappresentabile in matlab
rmin=realmin

format long e
% logspace come parametri prende gli esponenti a cui elevare 10 e quindi m
% é il logaritmo in base 10 di realmin
m=log10(realmin)

% vettore di 21 elementi logaritmicamente equispaziati fra 10^(-m) e
% 10^(-m-20)
v=logspace(m,m-20,21)
% i valori ottenuti dalla colonna 17(2.4*10^-323) in poi non sono piu
% visualizzabili su matlab.perche oltre al minimo rappresentabile che è
% realmin, matlab toglie cifre di precisione dall'espondente per usarle
% nell'esponente.

format short
%b)
rmax=realmax
rmax2=realmax*10
% moltiplicando il piu grande numero rappresentabile su matlab per 10 si
% presenta un overflow, quindi non essendo rappresentabile ritorna il
% valore inf

rmax3=realmax*(1+eps)
% 1+eps è un numero maggiore di 1 quindi quando viene moltiplicato per
% realmax si presenta overflow, ritornando un numero maggiore di realmax.

%c)
val1=1+10^-17
% il risultato viene arrotondato per difetto a 1
val2=1+10^17
% il risultato viene arrotondato per eccesso a 10^17
```

COMMAND WINDOW

```
rmin =

  2.2251e-308


m =

    -3.076526555685888e+02
```

```
v =

  Columns 1 through 4

    2.225073858507188e-308    2.225073858507189e-309    2.225073858507203e-310    2.225073858506956e-311

  Columns 5 through 8

    2.225073858506956e-312    2.225073858521778e-313    2.225073858324152e-314    2.225073859806349e-315

  Columns 9 through 12

    2.225073844984380e-316    2.225074042610638e-317    2.225074042610638e-318    2.225074042610638e-319

  Columns 13 through 16

    2.225271668868974e-320    2.223295406285609e-321    2.223295406285609e-322    2.470328229206233e-323

  Columns 17 through 20

                         0                         0                         0                         0

  Column 21

                         0


rmax =

  1.7977e+308


rmax2 =

   Inf


rmax3 =

   Inf


val1 =

     1

val2 =

   1.0000e+17
```