//inizializziamo una matrice generica con la lettera maiuscola
//elenco gli elementi riga per riga. Gli elementi son separati da virgole, le righe da ;

```
A = [1,2,3;4 5 6;7 8 9]
```

```
A =

    1    2    3
    4    5    6
    7    8    9
```

```
A = [1,2,3;4 5;7 8 9] //se ad una riga assegno meno elementi matlab restituisce un errore di dimensione
```

//matlab vede la matrice come una serie di colonne, e vede la terza colonna incompleta

{Error using vertcat

Dimensions of arrays being concatenated are not consistent.

}

//inizializzo un vettore colonna

```
x = [1;4;6;23;98]
```

```
x =

    1
    4
    6
   23
   98
```

//il vettore colonna si puo esprimere come vettore riga e trasporlo con ‘, usando gli spazi al posto del ;

//NB: l’apice fa la trasposta dei complessi coniugati, quindi se la matrice è reale allora risulta uguale alla trasposta

```
y = [21 34 2 1 -9]'
```

```
y =
```

21

34

2

1

-9

//funzioni per inizializzare matrici particolari

//Marice IDENTITA'

I = eye(7) //in questo caso la matrice sarà 7x7

I =

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

//matrice DI ZERI

B = zeros(5,4) //5 righe e 4 colonne

B =

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

B = zeros(5,5) //quadrata

B =

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

B = zeros(5) //seconda modalità di rappresentazione per una matrice quadrata

B =

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

//matrice di UNO

C = ones(3,4)

C =

1 1 1 1

1 1 1 1

1 1 1 1

C = ones(3)

C =

1 1 1

```
   1   1   1
   1   1   1
```

//se voglio creare una matrice di tutti 23 posso moltiplicare 23*la matrice ones

```
D = 23*ones(6)


D =


   23   23   23   23   23   23
   23   23   23   23   23   23
   23   23   23   23   23   23
   23   23   23   23   23   23
   23   23   23   23   23   23
   23   23   23   23   23   23
```

//COMANDI DI TEST

//generazione di numeri pseudocasuali

```
R = rand(5,6)
```

//ci da una matrice di numeri casuali tra 0 e 1 della dimensione che gli diamo noi
//tutti i numeri hanno la stessa probabilità di uscire

```
R =


   0.8147   0.0975   0.1576   0.1419   0.6557   0.7577
   0.9058   0.2785   0.9706   0.4218   0.0357   0.7431
   0.1270   0.5469   0.9572   0.9157   0.8491   0.3922
   0.9134   0.9575   0.4854   0.7922   0.9340   0.6555
   0.6324   0.9649   0.8003   0.9595   0.6787   0.1712
```

//se dovessi fare nuovamente rand, si baserà lo stesso seme, e quindi sarà uguale.

//voglio generare dei numeri che vanno da 2 a 100: rand(2,98) perché 2+98=100
//quindi il primo numero è la partenza, il secondo è il numero di elementi dell'intervallo.
//se avessi messo rand(2,100) la finestra sarebbe stata fino a 102

//rand con numeri interi: do un range(posso usare gli estremi questa volta) e la dim della matrice

```
S = randi([2,100],6,8)
```

S =

71	70	77	72	13	76	56	82
5	33	80	76	51	27	15	26
29	96	20	29	97	52	16	93
6	5	50	69	35	71	27	36
11	45	46	66	59	90	85	21
83	39	65	18	24	96	27	26

S = randi(100,6,8) //in questo caso, non avendo messo il primo numero dell'intervallo, lui mette 0

S =

62	92	8	57	32	69	16	11
48	29	6	47	53	75	83	97
36	76	54	2	17	46	54	1
84	76	78	34	61	9	100	78
59	39	94	17	27	23	8	82
55	57	13	80	66	92	45	87

T = randn(4,5) //n sta per normal distribution, ovvero la gaussiana a campana. Si ha probabilità maggiore nel valor medio, e la prob è minore allontanandoci da esso
//in questo caso abbiamo molti valori vicini a zero, perché la media è quella
//se volessi cambiare il punto centrale, basta sommare a randn

T =

-1.0582	-0.2779	-0.8236	0.0335	-0.2991
-0.4686	0.7015	-1.5771	-1.3337	0.0229
-0.2725	-2.0518	0.5080	1.1275	-0.2620
1.0984	-0.3538	0.2820	0.3502	-1.7502

whos

Name	Size	Bytes	Class	Attributes
A	3x3	72	double	
B	5x5	200	double	
C	3x3	72	double	
D	6x6	288	double	
I	7x7	392	double	
R	5x6	240	double	
S	6x8	384	double	
T	4x5	160	double	
x	5x1	40	double	
y	5x1	40	double	

//OPERAZIONI TRA MATRICI

//somma

E = A+C

E =

2 3 4

5 6 7

8 9 10

//somma tra vettori

z = x+y

z =

22

38

8

24

89

//nota bene: matlab fa operazioni non consentite! Ovvero somma un vettore colonna ad un vettore riga

```
w = [3 1 54 -2 1]

w =

   3    1   54   -2    1
-!-!-!-!-!-!-!-!-!-!
x+w

ans =

   4    2   55   -1    2
   7    5   58    2    5
   9    7   60    4    7
  26   24   77   21   24
 101   99  152   96   99
-!-!-!-!-!-!-!-!-!-!
whos
 Name      Size            Bytes  Class     Attributes

 A         3x3                72  double
 B         5x5               200  double
 C         3x3                72  double
 D         6x6               288  double
 E         3x3                72  double
 I         7x7               392  double
 R         5x6               240  double
 S         6x8               384  double
 T         4x5               160  double
 ans       5x5               200  double
 w         1x5                40  double
 x         5x1                40  double
 y         5x1                40  double
```

z 5x1 40 double

//moltiplicazione

/posso fare T*B ma non posso fare B*T perché in questo caso devo avere corrispondenza tra colonne della prima matrice e righe della seconda

U = T*B

U =

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

B*T

{Error using *

Incorrect dimensions for matrix multiplication. Check that the number of

columns in the first matrix matches the number of rows in the second matrix.

To perform elementwise multiplication, use '.*'.

Related documentation

}

//potenza solo matrici quadrate

C^2

ans =

3 3 3

3 3 3

3 3 3

C

C =

1 1 1

1 1 1

1 1 1

//dot power – potenza elemento per elemento. Posso farla anche se la matrice non è quadrata!
//NB: se si fa C^.2 escono elementi complessi perché 0.2=1/5, e si sta facendo la radice quinta di C intero

C.^2

ans =

1 1 1

1 1 1

1 1 1

x.^2

ans =

1

16

36

529

9604

x

x =

1

4

6

23

98

//divisioni

help /

/ Slash or right matrix divide. //questo comando fa A moltiplicato l'inverso di B

A/B is the matrix division of B into A, which is roughly the

same as A*INV(B) , except it is computed in a different way.

More precisely, A/B = (B'\A')'. See MLDIVIDE for details.

C = mrdivide(A,B) is called for the syntax 'A / B' when A or B is an

object.

See also mldivide, rdivide, ldivide.

Documentation for mrdivide

Other functions named mrdivide

help \

\ Backslash or left matrix divide. //questo comando fa l'inverso di A per B

A\B is the matrix division of A into B, which is roughly the

same as INV(A)*B , except it is computed in a different way.

If A is an N-by-N matrix and B is a column vector with N

components, or a matrix with several such columns, then

X = A\B is the solution to the equation A*X = B. A warning

message is printed if A is badly scaled or nearly singular.

A\EYE(SIZE(A)) produces the inverse of A.

If A is an M-by-N matrix with M < or > N and B is a column

vector with M components, or a matrix with several such columns,

then X = A\B is the solution in the least squares sense to the

under- or overdetermined system of equations A*X = B. The

effective rank, K, of A is determined from the QR decomposition

with pivoting. A solution X is computed which has at most K

nonzero components per column. If K < N this will usually not

be the same solution as PINV(A)*B. A\EYE(SIZE(A)) produces a

generalized inverse of A.

C = mldivide(A,B) is called for the syntax 'A \ B' when A or B is an

object.

See also ldivide, rdivide, mrdivide.

Documentation for mldivide

Other functions named mldivide

whos

Name	Size	Bytes	Class	Attributes
A	3x3	72	double	
B	5x5	200	double	
C	3x3	72	double	
D	6x6	288	double	
E	3x3	72	double	
I	7x7	392	double	
R	5x6	240	double	
S	6x8	384	double	
T	4x5	160	double	
U	4x5	160	double	
ans	5x1	40	double	

w	1x5	40	double
x	5x1	40	double
y	5x1	40	double
z	5x1	40	double

C = rand(5)

C =

0.2399	0.0497	0.3377	0.3897	0.9421
0.1233	0.9027	0.9001	0.2417	0.9561
0.1839	0.9448	0.3692	0.4039	0.5752
0.2400	0.4909	0.1112	0.0965	0.0598
0.4173	0.4893	0.7803	0.1320	0.2348

//faccio b * inverso di c

B/C

ans =

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

//vediamo se effettivamente risulta uguale

B*inv(C)

ans =

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

```
   0    0    0    0    0

   0    0    0    0    0
```

//faccio B per l'inverso di C

```
B\C

[Warning: Matrix is singular to working precision.]

[]
```

//ottengo dei valori NaN=not a number (0*inf, inf*inf etc) e dei valori inf che indica overflow
//NaN può essere usato però come valore, come se fosse un numero.

```
ans =


  Inf   Inf   Inf   Inf   Inf

  NaN   NaN   NaN   NaN   NaN

  NaN   NaN   NaN   NaN   NaN

  NaN   NaN   NaN   NaN   NaN

  NaN   NaN   NaN   NaN   NaN

inv(B)*C

[Warning: Matrix is singular to working precision.]

[]


ans =


  Inf   Inf   Inf   Inf   Inf

  Inf   Inf   Inf   Inf   Inf

  Inf   Inf   Inf   Inf   Inf

  Inf   Inf   Inf   Inf   Inf

  Inf   Inf   Inf   Inf   Inf


a = nan


a =


  NaN
```

```
2*nan

ans =

   NaN

D

D =

    23    23    23    23    23    23
    23    23    23    23    23    23
    23    23    23    23    23    23
    23    23    23    23    23    23
    23    23    23    23    23    23
    23    23    23    23    23    23
//vediamo come estrarre degli elementi da una radice
C

C =

    0.2399    0.0497    0.3377    0.3897    0.9421
    0.1233    0.9027    0.9001    0.2417    0.9561
    0.1839    0.9448    0.3692    0.4039    0.5752
    0.2400    0.4909    0.1112    0.0965    0.0598
    0.4173    0.4893    0.7803    0.1320    0.2348

whos C
  Name      Size            Bytes  Class     Attributes

  C         5x5               200  double
```

//estraggo l'elemento riga 1 col 2

C(1,2)

ans =

0.0497

//estraggo tutta una riga, ad esempio la seconda: indice di colonna=2; indice di colonna è un range dal primo all'ultimo

C(2,1:5)

ans =

0.1233 0.9027 0.9001 0.2417 0.9561

//per estrarre tutta la colonna posso anche omettere l'indice di colonna

C(2,:)

ans =

0.1233 0.9027 0.9001 0.2417 0.9561

//posso anche dire di estrarre dall'elemento 1 a end

C(2,1:end)

ans =

0.1233 0.9027 0.9001 0.2417 0.9561

//posso anche estrarre porzioni di colonna, ad esempio da 3 alla fine

C(2,3:end)

ans =

0.9001 0.2417 0.9561

//ovviamente gli indici possono essere dei range anche per le righe

```
C(:,3)

ans =

    0.3377
    0.9001
    0.3692
    0.1112
    0.7803

C(1:end,3)

ans =

    0.3377
    0.9001
    0.3692
    0.1112
    0.7803

C(1:5,3)

ans =

    0.3377
    0.9001
    0.3692
    0.1112
    0.7803
```

```
C
```

```
C =

    0.2399    0.0497    0.3377    0.3897    0.9421
    0.1233    0.9027    0.9001    0.2417    0.9561
    0.1839    0.9448    0.3692    0.4039    0.5752
    0.2400    0.4909    0.1112    0.0965    0.0598
    0.4173    0.4893    0.7803    0.1320    0.2348
```

```
//posso quindi estrarre dei pezzi di matrice dando dei range di righe e colonne
C(2:4,2:3)

ans =

    0.9027    0.9001
    0.9448    0.3692
    0.4909    0.1112
```

```
//posso anche selezionare elementi non contigui, ad esempio la riga 2 e la riga 4 e le colonne da 2 a 3
C([2,4],2:3)

ans =

    0.9027    0.9001
    0.4909    0.1112
```

```
//in questo caso per la riga 2 estraggo 4 elementi; riga 2,col 1; riga 2 col 3; riga 4 col1;riga 4 col3.
C([2,4],[1,3])

ans =

    0.1233    0.9001
    0.2400    0.1112
```

C([2,4,5],[1,3])

ans =

0.1233 0.9001

0.2400 0.1112

0.4173 0.7803

C

C =

0.2399 0.0497 0.3377 0.3897 0.9421

0.1233 0.9027 0.9001 0.2417 0.9561

0.1839 0.9448 0.3692 0.4039 0.5752

0.2400 0.4909 0.1112 0.0965 0.0598

0.4173 0.4893 0.7803 0.1320 0.2348

//cerchiamo all'interno della matrice gli elementi maggiori di 0.5

find(C>0.5) //restituisce un vettore colonna con i numeri degli elementi(NON IL VALORE!)in cui il valore è maggiore di 0.5

ans =

7

8

12

15

21

22

23

[i,j] = find(C>0.5) //se find viene chiamato con 2 output, ottendo gli indici riga-colonna

//vettore riga

i =

2

3

2

5

1

2

3

//vettore colonna

j =

2

2

3

3

5

5

5

//concatenazione di elementi colonna – quindi scrittura di colonne su una matrice

[i,j]

ans =

2	2
3	2
2	3
5	3
1	5

2 5

3 5

[i;j] //in questo caso ottengo un vettore unico di dimensione i+j

ans =

2

3

2

5

1

2

3

2

2

3

3

5

5

5

whos C x

Name	Size	Bytes	Class	Attributes
C	5x5	200	double	
x	5x1	40	double	

//posso affiancare C con x perché hanno lo stesso numero di righe

Q = [C,x]

Q =

0.2399 0.0497 0.3377 0.3897 0.9421 1.0000

0.1233 0.9027 0.9001 0.2417 0.9561 4.0000

0.1839 0.9448 0.3692 0.4039 0.5752 6.0000

0.2400 0.4909 0.1112 0.0965 0.0598 23.0000

0.4173 0.4893 0.7803 0.1320 0.2348 98.0000

//se provo ad affiancare un vettore riga accanto a C, matlab restituisce errore

Q = [C,w]

{Error using horzcat

Dimensions of arrays being concatenated are not consistent.

}

Q = [C;w] //posso però affiancare w sotto C, perché w è un vettore riga

Q =

0.2399 0.0497 0.3377 0.3897 0.9421

0.1233 0.9027 0.9001 0.2417 0.9561

0.1839 0.9448 0.3692 0.4039 0.5752

0.2400 0.4909 0.1112 0.0965 0.0598

0.4173 0.4893 0.7803 0.1320 0.2348

3.0000 1.0000 54.0000 -2.0000 1.0000

//costruisco una matrice formata da elementi complessi

R = rand(5) + i*rand(5)

{Error using *

Incorrect dimensions for matrix multiplication. Check that the number of

columns in the first matrix matches the number of rows in the second matrix.

To perform elementwise multiplication, use '.*'.

Related documentation

}

clear i

//costruisco una matrice formata da elementi complessi sommando tra loro una matrice intera e una matrice formata da numeri immaginari

```
R = rand(5) + i*rand(5)
```

```
R =

  Columns 1 through 4

   0.9234 + 0.4886i   0.4389 + 0.5468i   0.2622 + 0.6791i   0.2967 + 0.8852i
   0.4302 + 0.5785i   0.1111 + 0.5211i   0.6028 + 0.3955i   0.3188 + 0.9133i
   0.1848 + 0.2373i   0.2581 + 0.2316i   0.7112 + 0.3674i   0.4242 + 0.7962i
   0.9049 + 0.4588i   0.4087 + 0.4889i   0.2217 + 0.9880i   0.5079 + 0.0987i
   0.9797 + 0.9631i   0.5949 + 0.6241i   0.1174 + 0.0377i   0.0855 + 0.2619i

  Column 5

   0.2625 + 0.3354i
   0.8010 + 0.6797i
   0.0292 + 0.1366i
   0.9289 + 0.7212i
   0.7303 + 0.1068i
```

//faccio l'operazione del complesso coniugato con l'apice '

```
R'
```

```
ans =

  Columns 1 through 4

   0.9234 - 0.4886i   0.4302 - 0.5785i   0.1848 - 0.2373i   0.9049 - 0.4588i
   0.4389 - 0.5468i   0.1111 - 0.5211i   0.2581 - 0.2316i   0.4087 - 0.4889i
   0.2622 - 0.6791i   0.6028 - 0.3955i   0.7112 - 0.3674i   0.2217 - 0.9880i
   0.2967 - 0.8852i   0.3188 - 0.9133i   0.4242 - 0.7962i   0.5079 - 0.0987i
```

0.2625 - 0.3354i 0.8010 - 0.6797i 0.0292 - 0.1366i 0.9289 - 0.7212i

Column 5

0.9797 - 0.9631i

0.5949 - 0.6241i

0.1174 - 0.0377i

0.0855 - 0.2619i

0.7303 - 0.1068i

//se voglio fare solo il trasposto senza fare il complesso coniugato

R.'

ans =

Columns 1 through 4

0.9234 + 0.4886i 0.4302 + 0.5785i 0.1848 + 0.2373i 0.9049 + 0.4588i

0.4389 + 0.5468i 0.1111 + 0.5211i 0.2581 + 0.2316i 0.4087 + 0.4889i

0.2622 + 0.6791i 0.6028 + 0.3955i 0.7112 + 0.3674i 0.2217 + 0.9880i

0.2967 + 0.8852i 0.3188 + 0.9133i 0.4242 + 0.7962i 0.5079 + 0.0987i

0.2625 + 0.3354i 0.8010 + 0.6797i 0.0292 + 0.1366i 0.9289 + 0.7212i

Column 5

0.9797 + 0.9631i

0.5949 + 0.6241i

0.1174 + 0.0377i

0.0855 + 0.2619i

0.7303 + 0.1068i

v = C>0.5 //ottengo una matrice di elementi logici, con 0 dove non è verificata la condizione, 1 dove è verificata

v =

5×5 logical array

0 0 0 0 1

0 1 1 0 1

0 1 0 0 1

0 0 0 0 0

0 0 1 0 0

inv(C) //inversa: operazione molto instabile, non usarla

ans =

1.2895	-0.2668	-2.1920	4.4231	0.1565
-0.6155	0.6288	0.1034	1.4573	-0.7156
-0.5991	-0.1297	0.7961	-2.9637	1.7362
0.0100	-3.2775	5.1683	-6.1163	2.2022
0.9762	1.4372	-1.8708	2.3896	-1.5357

C^(-1)//maniera + stabile per calcolare l'inversa

ans =

1.2895	-0.2668	-2.1920	4.4231	0.1565
-0.6155	0.6288	0.1034	1.4573	-0.7156
-0.5991	-0.1297	0.7961	-2.9637	1.7362
0.0100	-3.2775	5.1683	-6.1163	2.2022
0.9762	1.4372	-1.8708	2.3896	-1.5357

C\eye(5) //inversa di C moltiplicato identità. Useremo questo metodo

ans =

1.2895	-0.2668	-2.1920	4.4231	0.1565
-0.6155	0.6288	0.1034	1.4573	-0.7156
-0.5991	-0.1297	0.7961	-2.9637	1.7362
0.0100	-3.2775	5.1683	-6.1163	2.2022
0.9762	1.4372	-1.8708	2.3896	-1.5357

det(C) //determinante matrice C

ans =

-0.0326

rank(A)//rango di A

ans =

2 //nota bene: il rango di A è 2, ma la sua dimensione è 3x3. Questo vuol dire che una riga o colonna di A è linearmente dipendente dalle altre

A

A =

1	2	3
4	5	6
7	8	9

det(A)

```
ans =

 -9.5162e-16

x

x =

     1
     4
     6
    23
    98

norm(x,1)//voglio calcolare la norma 1 del vettore x

ans =

   132

norm(x,inf) )//voglio calcolare la norma infinito del vettore x

ans =

    98

norm(x,2) )//voglio calcolare la norma 2 del vettore x

ans =

  100.9257
```

```
norm(x) )// calcola la norma 2 del vettore x

ans =

  100.9257

norm(x,3/2) )// calcola la norma 3/2 del vettore x

ans =

  106.8282

diary off
```