A brief MATLAB tutorial

Per Gustafsson pergu@it.uu.se
Polacksbacken rum 1320

Matlab

A high level programming language to perform numerical calculations and produce graphical output

Centered around matrices

Fast matrix manipulation

Slow ordinary programming language constructs (e.g. for-loops)

Syntax

Matlab uses standard infix notation

>> 2+5

Ans = 7

>> 9-3

Ans = 6

>> 12*10

Ans = 120

>> 5^3

Ans = 125

Syntax

Assigning variables

>> A = 2+14

A = 16

>> B = sqrt(A)

Ans = 4

>> C = B, pi, 2+3i

C = 4

Ans = 3.1416

Ans = 2.0000 + 3.0000 i

>> D = A+B+C;

Matrix creation (1)

>> A = 1:5

A =

1 2 3 4 5

>> A = 1:2:7

A =

1 3 5 7

>> A= [1 2 3 4 5]

A =

1 2 3 4 5

>> A = [1 2 3;1 2 3]

A =

1 2 3
1 2 3

Matrix creation (2)

>> zeros(3, 5)

ans =

0	*0*	*0*	*0*	*0*
0	*0*	*0*	*0*	*0*
0	*0*	*0*	*0*	*0*

>> ones(5, 3)

ans =

1	*1*	*1*
1	*1*	*1*
1	*1*	*1*
1	*1*	*1*
1	*1*	*1*

>> B = [1 2 3]

B =

1	*2*	*3*

>> A = [B;B]

A =

1	*2*	*3*
1	*2*	*3*

```
A =
   1   2   3
   1   2   3
```

Matrix manipulation

>> A'

ans =

```
   1   1
   2   2
   3   3
```

>> A(:,2)

ans =

```
   2
   2
```

>> A == 3

ans =

```
   0   0   1
   0   0   1
```

```
A =
   1    2    3
   1    2    3
```

Matrix arithmetics

>> A+A

ans =

```
   2    4    6
   2    4    6
```

>> A .* A

ans =

```
   1    4    9
   1    4    9
```

>> A*A

*??? Error using ==> **
Inner matrix dimensions must agree.

>> A*A'

ans =

```
  14   14
  14   14
```

```
A =
   1   2   3
   1   2   3
```

Some useful 'tricks' (1)

>> repmat(A, 1, 2)

ans =

```
   1   2   3   1   2   3
   1   2   3   1   2   3
```

>> randperm(4)

ans =

```
   2   4   3   1
```

>> A == 3

ans =

```
   0   0   1
   0   0   1
```

>> B(find(A == 2)) = 10

B =

```
   1  10   3
   1  10   3
```

```
A =
   1   2   3
   1   2   3
```

Some useful 'tricks' (2)

>> sum(A)

ans =

```
   2   4   6
```

>> sum(A, 2)

ans =

```
   6
   6
```

>> sort(randperm(4))

ans =

```
   1   2   3   4
```

Some useful 'tricks' (2)

```
>> B = [1 2 3; 4 5 6]

B =

    1    2    3
    4    5    6

>> mean(B)

ans =

    2.5000    3.5000    4.5000

>> std(B)

ans =

    2.1213    2.1213    2.1213

>> var(B)

ans =

    4.5000    4.5000    4.5000
```

Other useful instructions

- princomp(Data)
  - Returns the principal components.
- plot(A,B,Symbol))
  - Where symbol is one of ['+' 'o' '*' ...]
  - hold on/off
  - Grid on/off

Matlab control flow

```
for i = vector
   statements
end
```

```
while relation
   statements
end
```

example:

```
j = 0
for i = 1:10
   j=i+j;
end
```

example:

```
j = 0, i=1
while i <= 10
   j=i+j,i=i+1
end
```

Matlab control flow

```
if relation
   statements
elseif relation
   statements
else
   statements
end
```

```
example:
A = [0,1,2]
B = [1,2,3]
if A > B
   C = A - B
elseif B > A
   C = B - A
else
   C = A + B
end
```

Matlab functions

```
function[Ret1,...,RetM]=fun(Arg1,...ArgN)
   statements defining Ret1,...,RetN
end
```

This definition should be inside a file called fun.m

>> [x1,...,xM] = fun(a1,...,aN)

x1 = ... xM=..

Matlab functions

```
function Area=traparea(A,B,H)

 %calculates the area of a trapezoid
 %with length A, B of the parallell
 %sides and the distance H between the
 %sides

  Area=0.5*(A+B)*H

end
```

Matlab functions

```
function [Vol, Area]=cylinder(R,H)
 %calculates the volume and surface
 %area of a cylinder with radius R and
 %height H
  Area=2*pi*R*H;
  Vol=(Area*R)/2;
end
```

Think vectorised!

Instead of

```
for row = 1:nofRows
   for col = 1:nofCols
      A(row, col) = A(row, col) + 1
   end
end
```

Write

```
A = A + 1
```

More compact and often more efficient

Think vectorised!

Example (script run.m):

```
X=ones(500,500);
Y=ones(500,500);
tic   %start timer
for i=1:500
    for j=1:500
        Val = 0;
        for k = 1:500
            Val = X(i,k)*Y(k,j) + Val;
        end
        Z(i,j) = Val;
    end
end
toc    %stop timer and print elapsed time
tic    %start timer
Z=X*Y;
toc    %stop timer and print elapsed time
```

>> run

Elapsed time is 3.478420 seconds.

Elapsed time is 0.149770 seconds.