

Chapter 1: MATLAB Basics

interactive
Basics (), Matrices,
Graphics, Scripts (-), Functions, Miscellany.

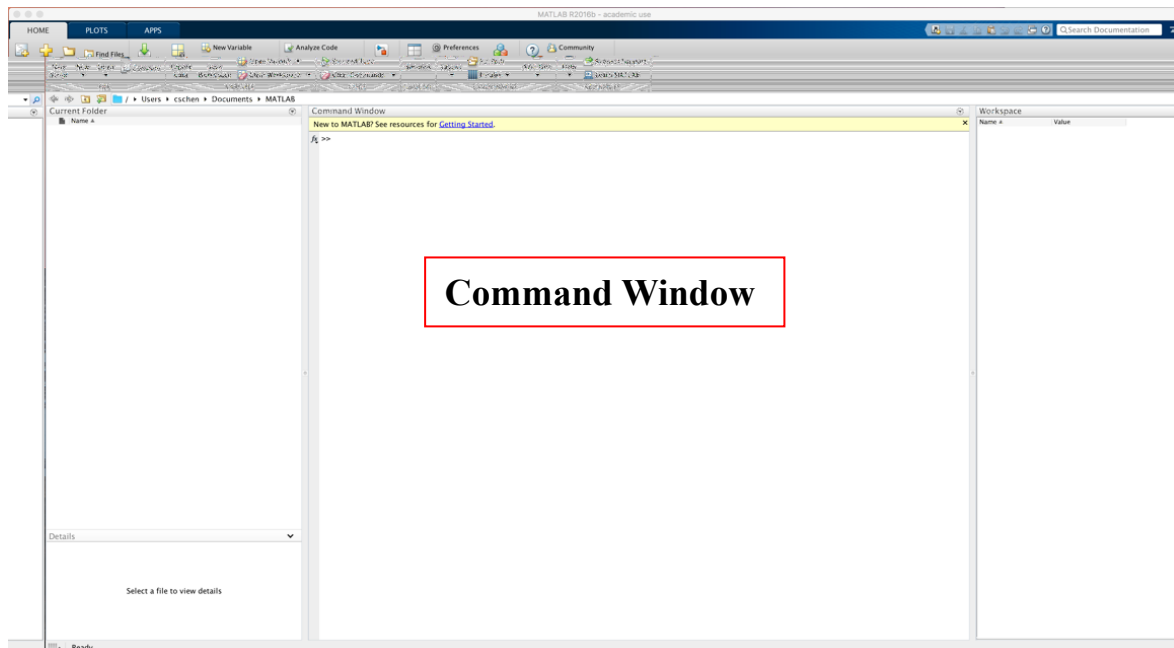
(MATLAB Guide, Second Edition,) (Introduction to MATLAB
for Engineers, 3rd Edition)

1.1 Getting Started with MATLAB

[install → ask TA]

Command Window,

>> ++,



().

(Demo)

```
>> a=[1 2 3]
a =
     1     2     3
```

(From A to A+)

,

.

●

(

a

A

.)

●

•

●

•

●

 $\circ$

, ,

, ,

•

.

●

•

1.2 Help

1.3 Things about MATLAB that are very nice to know

→ •

Ctrl-C (

).

→ •

ans	
<u>eps</u>	
Inf	
Nan	
pi	π

→ 2.1

→ 2.1001

•

.(

.)

```
>> a=[1 2 3]
```

```
a =
```

```
1      2      3
```

```
>> sqrt(a)
```

```
ans =
```

```
1.0000    1.4142    1.7321
```

```
>> format long
```

```
>> ans
```

```
ans =
```

```
1.0000000000000000    1.414213562373095    1.732050807568877
```

```
>> format
```

```
>> ans
```

```
ans =
```

```
1.0000    1.4142    1.7321
```

```
format
```

Command	Description and example
format short	Four decimal digits (the default); 13.6745.
format long	16 digits; 17.27484029463547.
format short e	Five digits (four decimals) plus exponent; 6.3792e+03.
format long e	16 digits (15 decimals) plus exponent; 6.379243784781294e-04.
format bank	Two decimal digits; 126.73.
format +	Positive, negative, or zero; +.
format rat	Rational approximation; 43/7.
format compact	Suppresses some blank lines.
format loose	Resets to less compact display mode.

Chapter 2: MATLAB Matrices

linear algebra.

encourages and expects

m-by-n

()

().

m rows

n columns.

2.1 Matrix Generation

[]

square

brackets. spaces or commas to separate columns,
separate rows

semicolons or new lines to
column

```
>> A = 1 2 3; 4 5 6; 7 8 9
A =
     1     2     3
     4     5     6
     7     8     9

>> b=[0;1;0]
b =
     0
     1
     0
```

row →

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

column ↓ ↓ ↓

```
>> size(A)
ans =
     3     3
>> size(b)
ans =
     3     1
# row ↑ ↑ # column
```

, as long as the sizes are compatible

```
>> [A b]
ans =
     1     2     3     0
     4     5     6     1
     7     8     9     0
```

Q:

| >> [A; b]

A:

| >> [A;b]
| ??? Error using ==> vertcat
| CAT arguments dimensions are not consistent.

(')

| >> A = [1 2 3; 4 5 6; 7 8 9]

A =

1 2 3
4 5 6
7 8 9

| >> '

ans =

1 4 7
2 5 8
3 6 9

()

0..1

| >> rand(3)
| ans =

0.8147 0.9134 0.2785
0.9058 0.6324 0.5469
0.1270 0.0975 0.9575

rand(3,1)

101010

(0,1).

, rand(M,N)

Example:



—

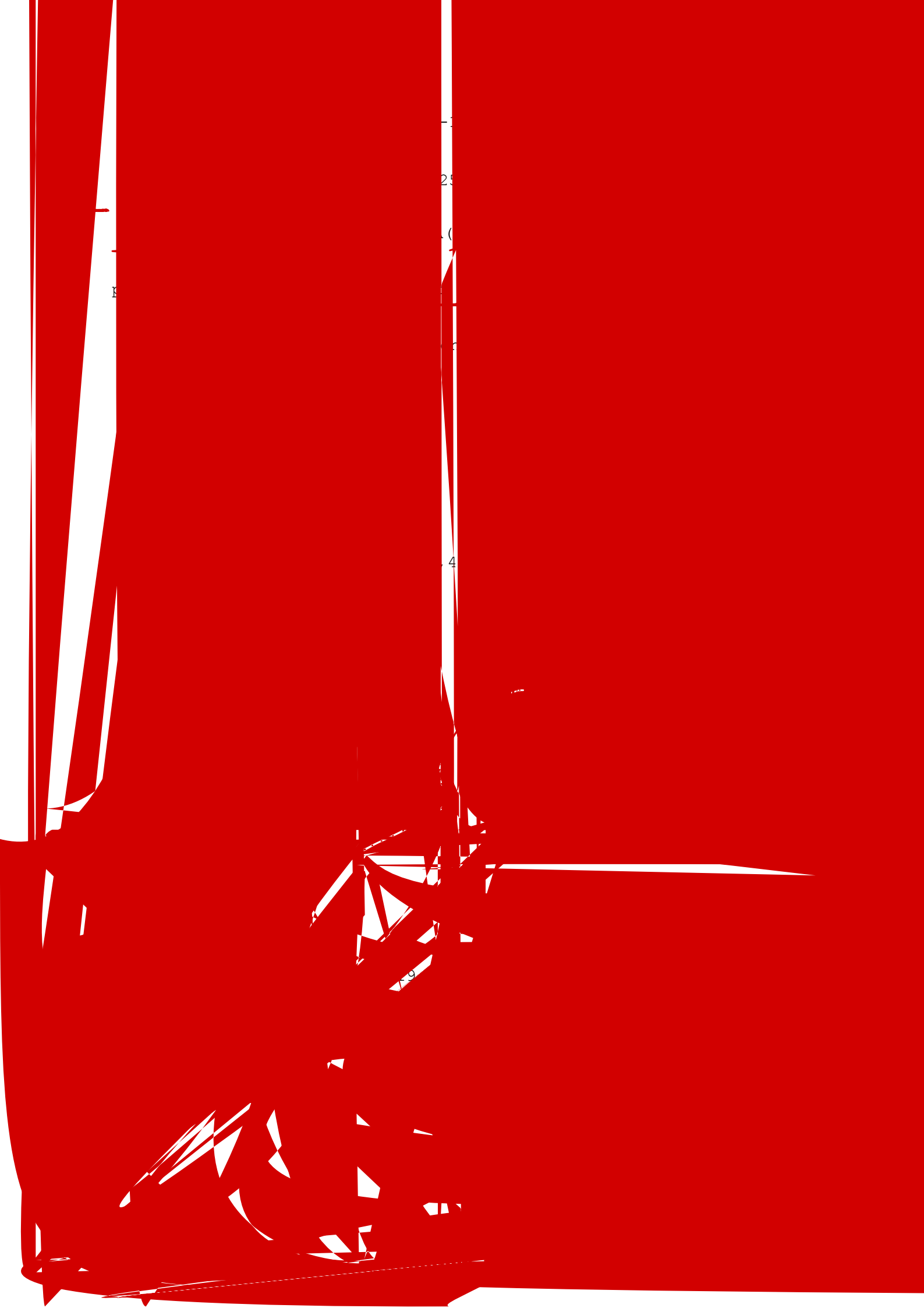
=

1..6

ceil

floor





```

Q:
| >> A(2:3,2:3)
A:

```

Q:
| >> A(:,1)
A:

Q:
| >> A(2,:) **A:**

```
Q:
| >> A(end:-1:1, end)
A:
```

Q:
| >> A([1 3],[2 3])
A:

```
linspace (          ). linspace(a,b,n)          n
      a      b.  n          .
```

```
>> linspace(-1,1,9)
ans =
    Columns 1 through 8
    -1.0000   -0.7500   -0.5000   -0.2500    0    0.2500    0.5000    0.7500
    Column 9
    1.0000
```

$$[\] \quad , \quad - \quad - \quad . \quad [\]$$



() cross.

$x \times y$

```
>> x=[-1 0 1]';y=[3 4 5]';  
>> x'*y  
ans =  
      2  
>> dot(x,y)  
ans =  
      2  
>> cross(x,y)  
ans =  
     -4  
      8  
     -4
```

-

$Ax = b$

| >> $x=A \backslash B$

Example

(x_1, x_2, x_3)

$$\begin{cases} 3x_1 + 2x_2 - x_3 = 10 \\ -x_1 + 3x_2 + 2x_3 = 5 \\ x_1 - x_2 - x_3 = -1 \end{cases}$$

$$\begin{bmatrix} 3 & 2 & -1 \\ -1 & 3 & 2 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 5 \\ -1 \end{bmatrix}$$

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

A =

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

```
>> b = [10; 5; -1]
```

```
b =
```

```
10  
5  
-1
```

```
>> x = A\b
```

```
x =
```

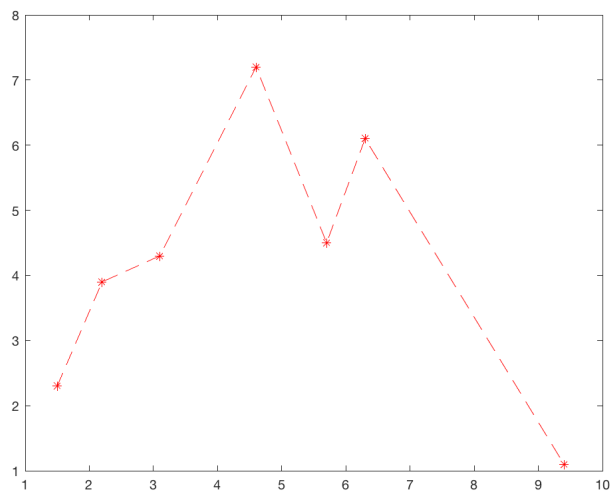
```
-2.0000  
5.0000  
-6.0000
```

```
>> check = A*x
```

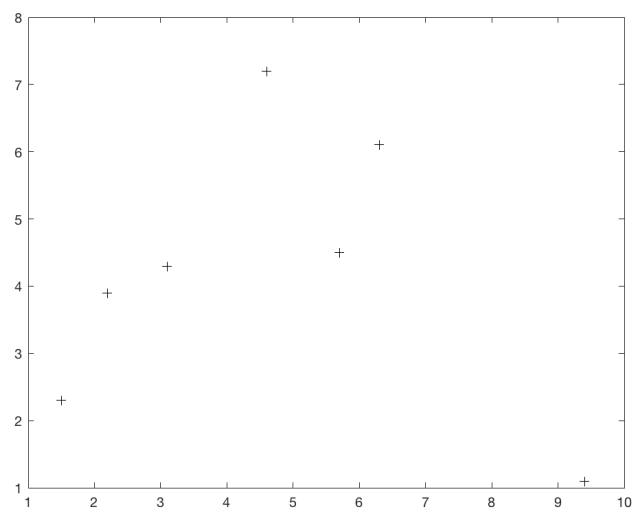
```
check =
```

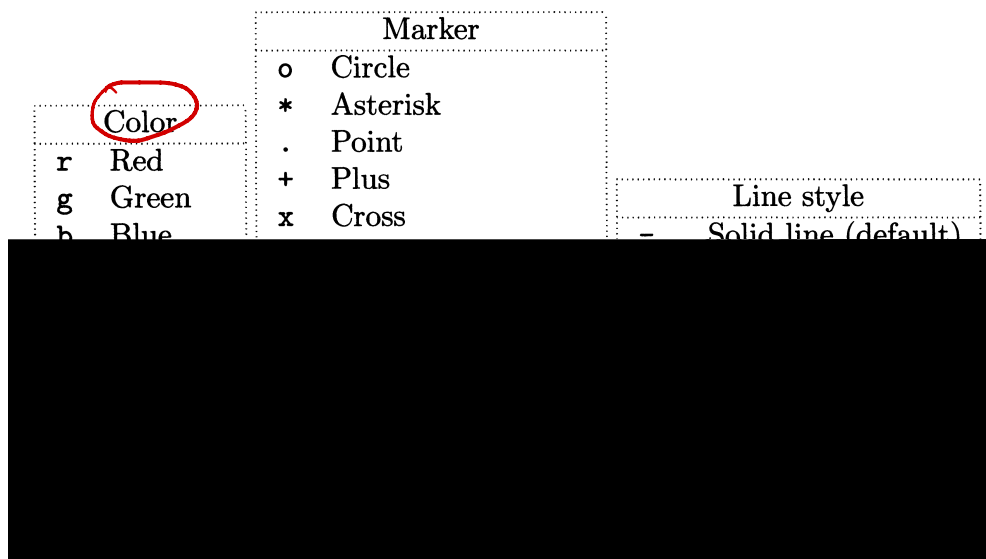
```
10.0000  
5.0000  
-1.0000
```



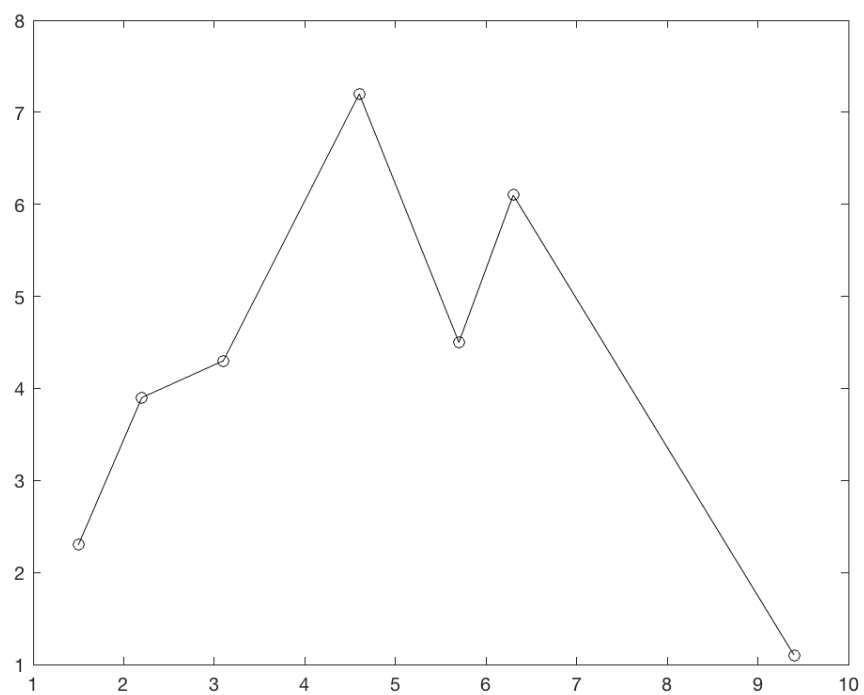


```
| >> plot(x,y,'k+',  
           ==)
```





Q: `plot`



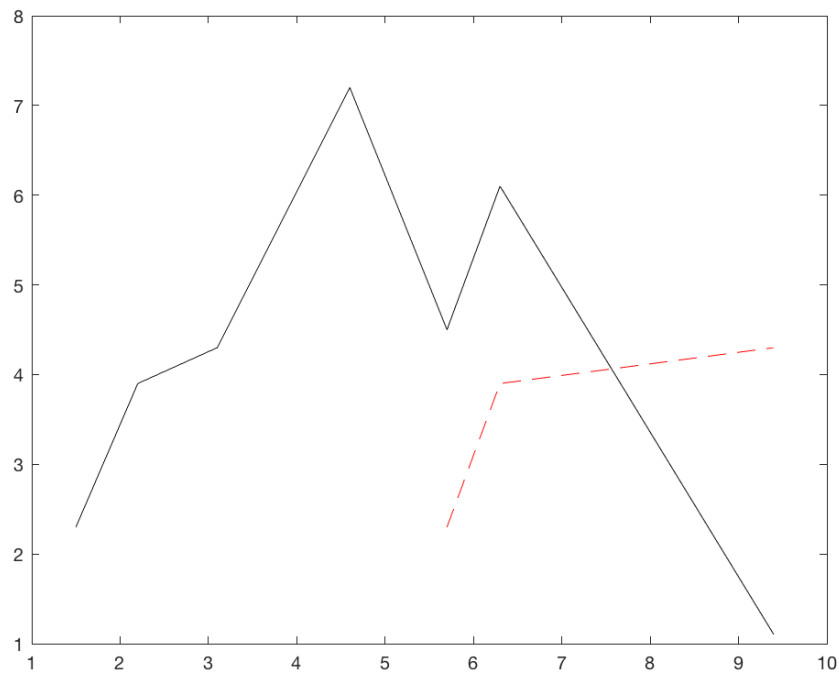
A:

More than one set of data

```
>> x=[1.5 2.2 3.1 4.6 5.7 6.3 9.4];
>> y=[2.3 3.9 4.3 7.2 4.5 6.1 1.1];
```

```
>> b=x(5:7);
>> c=y(1:3);
>> ( ' ' ' ' )
```

(), () (), ()



(hold off). (hold on

```
>> x=[1.5 2.2 3.1 4.6 5.7 6.3 9.4];
>> y=[2.3 3.9 4.3 7.2 4.5 6.1 1.1];
>> ( ' ' )
>> b=x(5:7);
>> c=y(1:3);
>> ( ' ' )
```

(From A to A+)

hold on , hold
off , hold
hold off.

loglog plot

logarithmically.

semilogx semilogy,

- - , ,

.

Chapter 4: More MATLAB Basic Elements

4.1 Script

Script
Functions > .m

script files.

M-files,

are interpreted as though they were typed at the command

prompt.

```
sort,
mean, median, std,
examMarks.m
```

```
% exam mark
exmark = [12 0 5 28 87 3 56];
exsort = sort(exmark)
exmean = mean(exmark)
exmed = median(exmark)
exstd = std(exmark)
```

examMarks

```
>> examMark
exsort =

     0     3     5    12    28    56    87
exmean =

    27.2857
exmed =

    12
exstd =

    32.8010
```

s
I-f

ions

y fr

a

pt.

rip is
o.
1) %

help/crip
his script is
or

B s

```

        , 'Enter an input value:',
        .
    ,
(
)

```

```

>> my_val = input('Enter an input value:');
Enter an input value: 3.4
>> my_val

my_val =

    3.4000

```

```

>> my_val = input('Enter an input value:');
Enter an input value: 1 2 3
>> my_val

my_val =

     1     2     3

```

4.2 Displaying Output Data

4.2.1 The disp function

disp . disp

```

>> pi

ans =
    3.1416

>> disp(pi)
3.1416

```

```

disp
int2str (
num2str(
)

```

```
>> str = ['The value of pi = ' num2str(pi)];
>> disp(str)
The value of pi = 3.1416
```

4.2.2 Formatted output with the fprintf function

fprintf

fprintf

fprintf

```
| fprintf(format,data)
```

```
| fprintf('The value of pi is %f \n',pi)
```

'The value of pi is 3.141593'

%f

\n

fprintf

Format String	Results
%d	Display value as an integer.
%e	Display value in exponential format.
%f	Display value in floating-point format.
%g	Display value in either floating-point or exponential format, whichever is shorter.
\n	Skip to a new line.

```

        %6.2f \n',pi)

    'The value of pi is 3.14'
    %6.2f
- _____

```

Example —Temperature Conversion

```

>> temp_conversion
Enter the temperature in degrees Fahrenheit: 75
75.00 degrees Fahrenheit = 297.04 kelvin.

```

$$T(\text{in kelvin}) = \left[\frac{5}{9} T(\text{in } ^\circ\text{F}) - 32.0 \right] + 273.15$$

Ans

Chapter 5: Branches and Loops

control
() branches,
loops,
ALWAYS
DO
NOT

5.1 Branches: if and switch

5.1.1 for a small number of alternatives

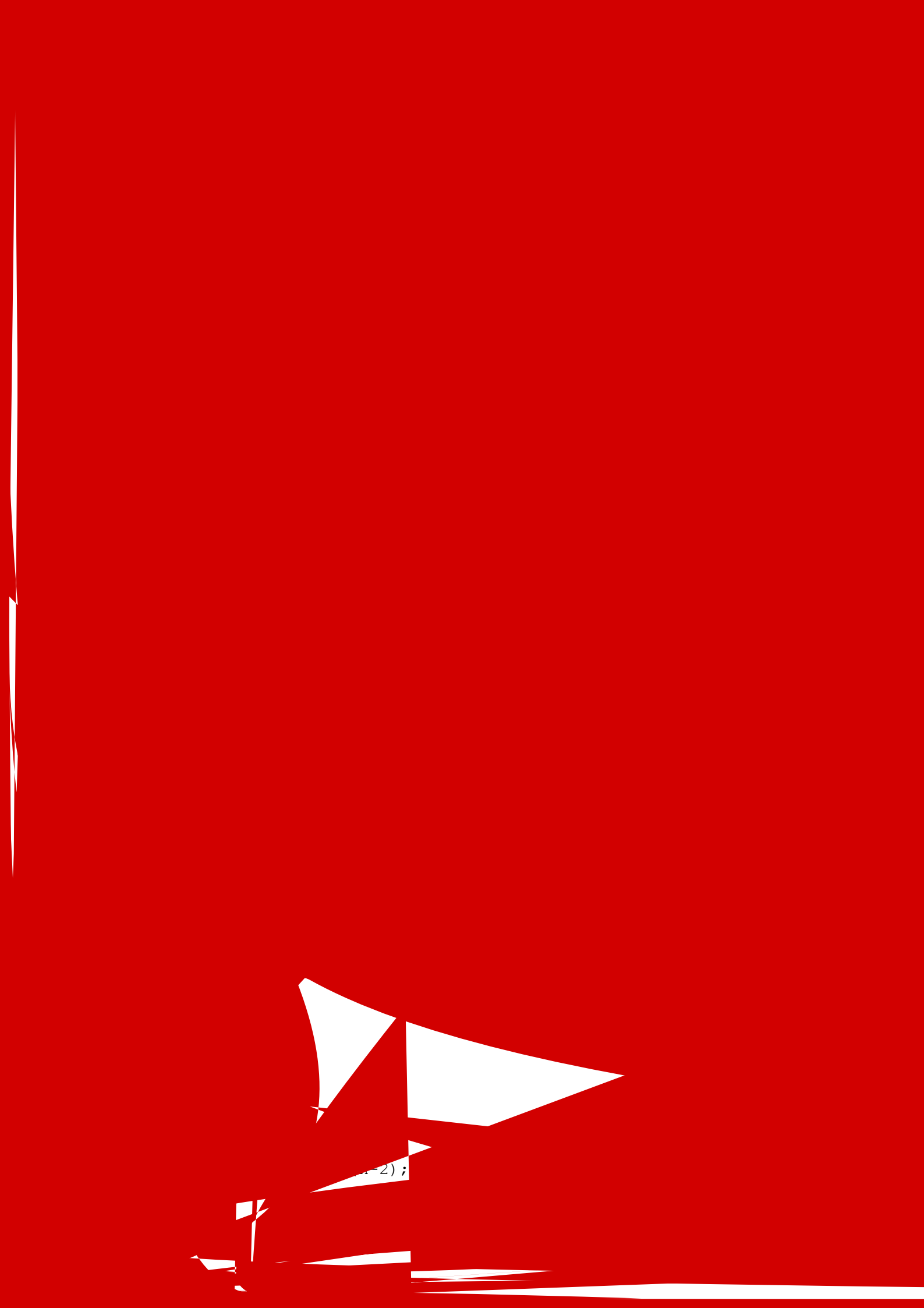
if

```
if logical expression 1
    statement group 1
    logical expression 2
    statement group 2
else
    statement group 3
```

```
else elseif
else elseif
```

```
x=-1
if x>=0
    disp('Absolute value is: ')
    y=x
elseif x < 0
    disp('Absolute value is: ')
    y=-x
else
```

er of alternative



```

end

>> f
f =
    1     1     2     3     5     8    13    21    34    55

```

n ,
colon 3:10

Exercise:

(forScript.m)

$$\sum_{k=1}^{15} 5k^2 - 2k$$

```

>> forScript
The sum for 15 terms is:
    5960

```

Ans:

→ for your !

5.2.2 while loop for repeating statements based on satisfying a condition rather than a fixed number of times

while
 . while

```

while expression
...
...
...
end

```

```

while abs(x) > 1
    x = x/2;
end

```

```

n = 0;
while abs(x) > 1
    x = x/2;
    n = n+1;
end

```

Exercise: (whileScript.m)

$$5k^2 + 2k, k=1,2,3,\dots$$

```

>> whileScript
The number of terms is:
    18

The sum is:
    10203

```

Ans:

⇒ left for your own exercise.

5.3 Implied loop and vectorization to avoid loops

good practice

Many MATLAB commands contain *implied loops*. For example, consider these statements:

```

x = [0:5:100];
y = cos(x)

```

for ,

```

for k = 1:21
    x = (k - 1)*5;
    y(k) = cos(x);
end

```

++,

Vectorization

column vector

$$d_{ij} = x_i - x_j.$$

```

n = length(x); % equivalent to max(size(x))
D = zeros(n); % memory preallocation
for j = 1:n
    for i = 1:n
        D(i,j) = x(i) - x(j);
    end
end

```

(From A to A+)

memory preallocation

(. . ,

)

```

n = length(x); % equivalent to max(size(x))
D = zeros(n); % memory preallocation
for j = 1:n
    D(:,j) = x - x(j)
end

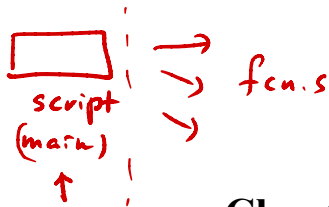
```

vectors and matrices

(

++

). ,



Chapter 6: User-Defined Functions

local workspace.

() (),

scoping,

divide and conquer
modularization

6.1 Basics

```
out1, out2 = myfun( in1, in2, in3 )
```

myfun **the name of the file on disk** (myfun.m). in1
input arguments, out1 output arguments.
ONLY

copies

(From A to A+)

```
function [ output_args ] = Untitled( input_args )
%UNTITLED Summary of this function goes here
% Detailed explanation goes here
```

end

end

optional. end

```

- , ( , , , )
end . end,
- .

```

Example: (quadform.m)

$$\underline{ax^2 + bx + c = 0}$$

```

function [x1,x2] = quadform (a,b,c)
%QUADFORM returns the roots of ax^2+bx+c from
% the quadratic formula
d = sqrt(b^2 - 4*a*c);
x1 = (-b + d) / (2*a);
x2 = (-b - d) / (2*a);

```

call

```

>> help quadform
QUADFORM returns the roots of ax^2+bx+c from
the quadratic formula

>> [a b]=quadform(1,2,1)
a =
-1
b =
-1
>> [a b]=quadform(1,3,2)
a =
-1
b =
-2
>> [a b]=quadform(3,2,1)
a =
-0.3333 + 0.4714i
b =
-0.3333 - 0.4714i

```

Exercise: circle

Ans:

$$\underline{\pi r^2} \quad \underline{2\pi r}$$

left for your own.

(From A to A+)

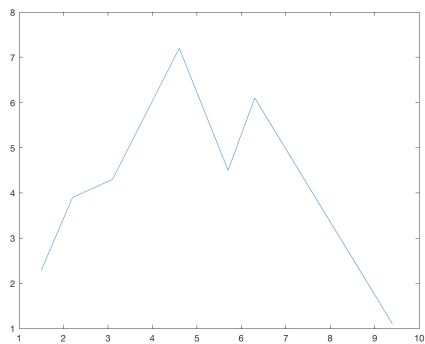
(劃分)

forget

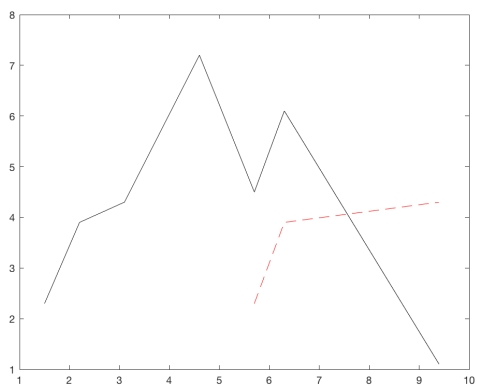
6.2 Optional Arguments

plot

```
>> x=[1.5 2.2 3.1 4.6 5.7 6.3 9.4];  
>> y=[2.3 3.9 4.3 7.2 4.5 6.1 1.1];  
      ( )
```



```
>> x=[1.5 2.2 3.1 4.6 5.7 6.3 9.4];  
>> y=[2.3 3.9 4.3 7.2 4.5 6.1 1.1];  
>> b=x(5:7);  
>> c=y(1:3);  
>> ( ' ' ' ' )
```



nargin

nargout

nargin

nargout

sqrtn

$a > 0,$

$\sqrt{a}$

$$x_{k+1} = \frac{1}{2} \left(x_k + \frac{a}{x_k} \right) \quad x_0 = a$$

$\sqrt{2}$

$$\frac{|x_{k+1} - x_k|}{|x_{k+1}|}$$

k	x_k	rel. change
1:	1.5000000000000000e+000	3.33e-001
2:	1.4166666666666665e+000	5.88e-002
3:	1.4142156862745097e+000	1.73e-003
4:	1.4142135623746899e+000	1.50e-006

function [x,iter] = sqrtn(a, tol)

tol

$$\left(\frac{|x_{k+1} - x_k|}{|x_{k+1}|} < \text{tol} \right) \left($$

eps)

iter

nargout

