Advanced Topics in MATLAB®

Abishek Gopal

agopal@tamu.edu

Fall 2019 HPRC Short Course

Nov 15, 2019

Disadvantages of Matlab scripts

- No private variables, all variables displayed in the workspace
- Running two different scripts with common variable names could lead to potential problems
- Does not accept input arguments
- Does not return output arguments

Functions

A **function** is a group of statements that together perform a task.

- Encapsulates internal data and procedure
- Accepts input arguments and returns outputs
- Can be called from scripts or the Command Window

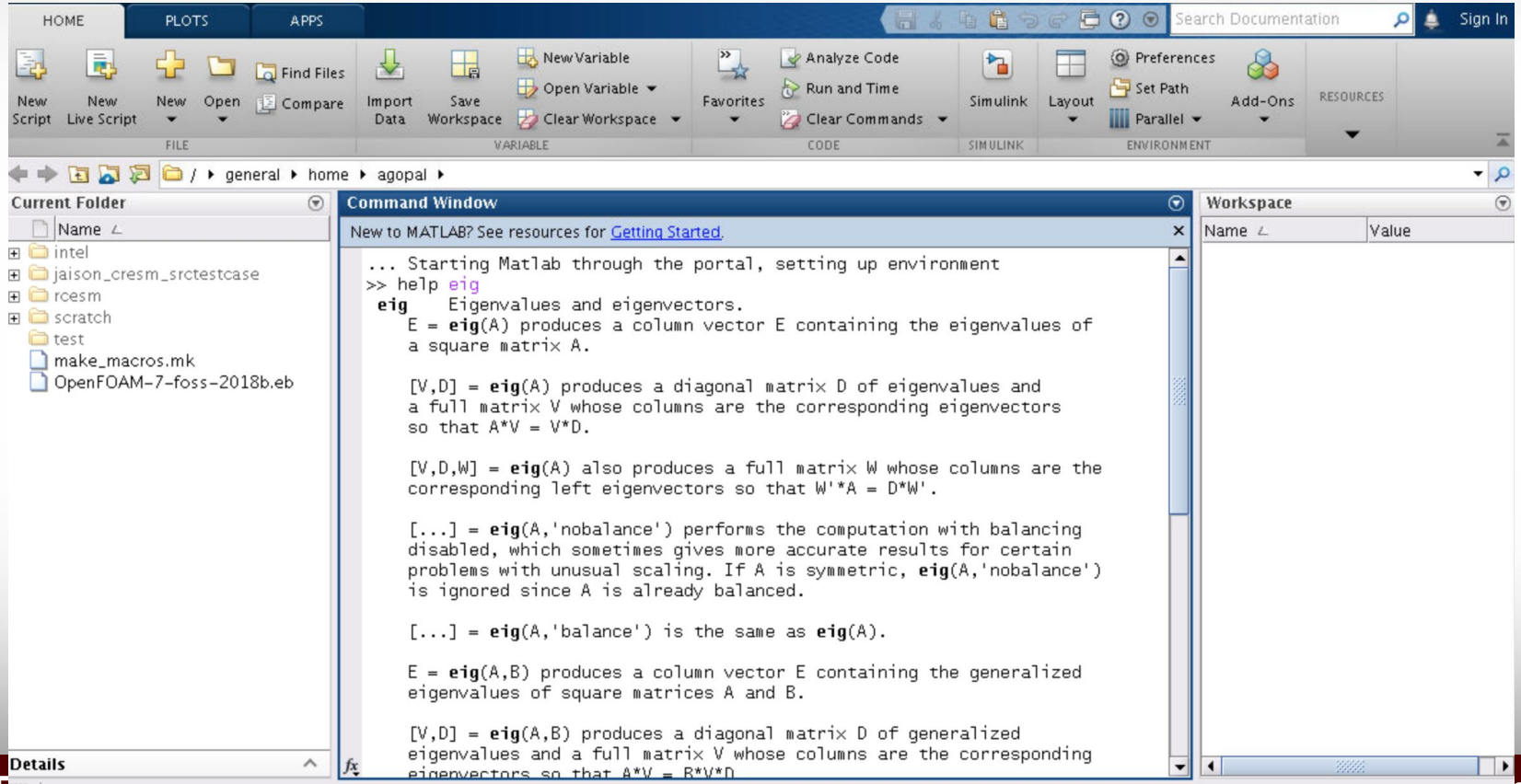
```
%Matlab
function[out]=myabs(number)
    if number > 0
        out = number
    else
        out = -number
    end
end
```

Functions

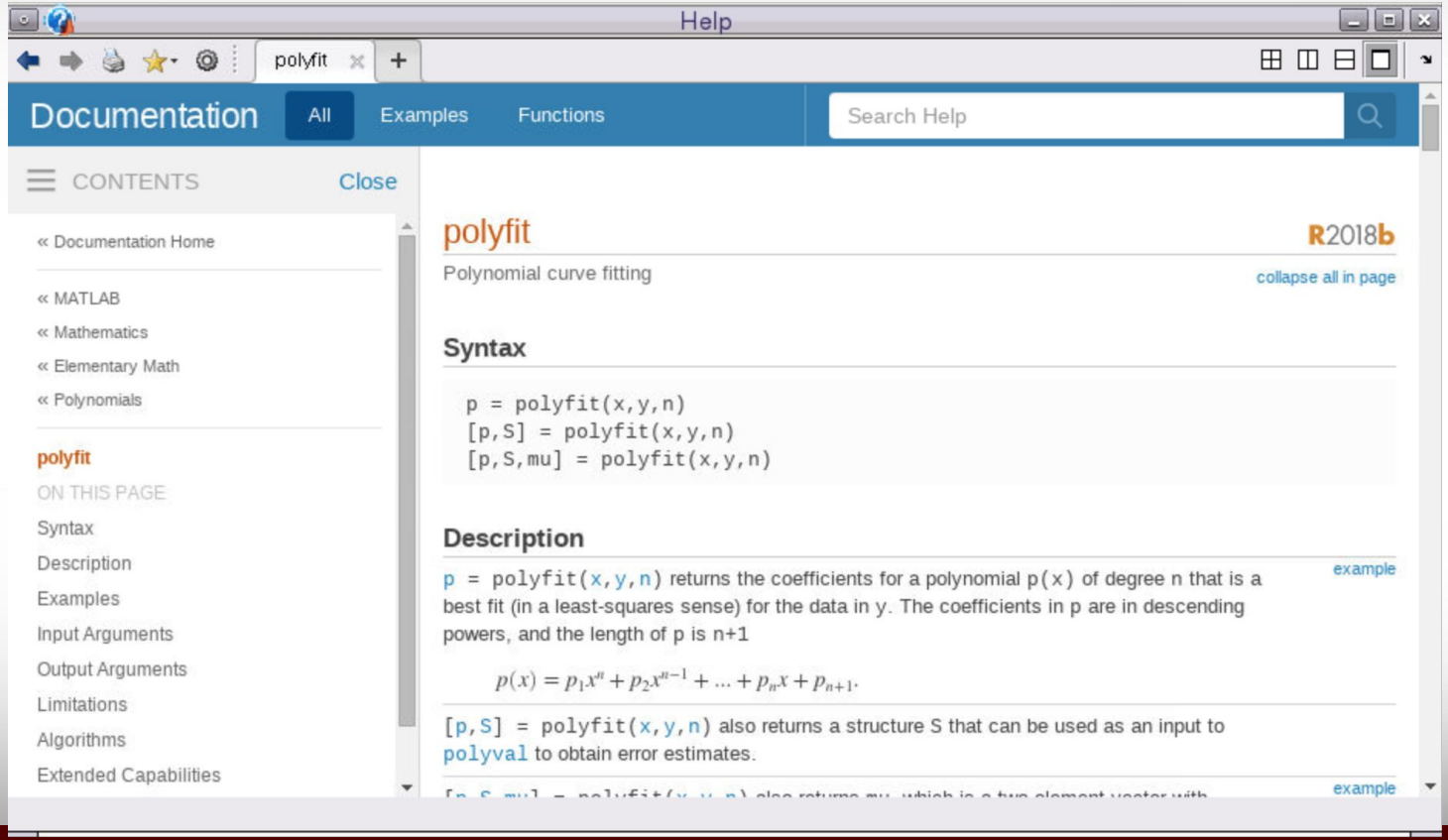
- In MATLAB, functions are defined in separate files.
- The name of the file and of the function should be the same.
- Files can include multiple local functions or nested functions

```
% myabs.m
function[out]=myabs(number)
    if number > 0
        out = number
    else
        out = -number
    end
end
```

Getting help in MATLAB



Getting help in MATLAB



MATLAB for Linear Algebra

Row and column vectors

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

x =  1   2   3

>> size(x)

ans = 1     3
```

```
y = [1 2 3]'

y =  1
     2
     3

>> size(y)
ans = 3     1
```

Vector multiplication

```
>> x*y
ans =  14
```

```
>> x.*y
ans =
 1     2     3
 2     4     6
 3     6     9
```

```
>> x.*y'
ans = 1   4   9
```

Matrix-vector multiplication

Try:

```
>> x = [1 2 3]
>> A = [1 2 3; 4 5 6 ; 7 8 9]
>> A*x
```

Now try:

```
>> A*x'
```

Matrix-matrix multiplication

```
>> A = [1 2; 3 4; 5 6];    % 3x2 matrix
>> B = [-1 0 1; 1 2 3];    % 2x3 matrix
>> A*B
ans =
     1     4     7
     1     8    15
     1    12    23
```

Solution of linear system

$$A\,x = b$$

$$x = A^{-1}\,b$$

```
>> b = [3 5 7]
>> A = magic(3);
```

```
>> x = inv(A)*b
```

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

Sparse matrices

```
>> A = sparse([0 2 0 1 0; 1 0 -1 0 0; 0 0
0 3 1; -2 0 0 0 2; 0 0 2 2 0]);
>> b = sparse([1; 0; 8; 4; 2]);
>> x = A\b
x =
    (1,1)       -1.5000
    (2,1)       -0.7500
    (3,1)       -1.5000
    (4,1)        2.5000
    (5,1)        0.5000
```

Eigenvalues/eigenvectors

$$A x = d x$$

```
>> A = magic(4)

>> [V,D]=eig(A)

V = -0.5000   -0.8236    0.3764   -0.2236
    -0.5000    0.4236    0.0236   -0.6708
    -0.5000    0.0236    0.4236    0.6708
    -0.5000    0.3764   -0.8236    0.2236

D =34.0000         0         0         0
         0    8.9443         0         0
         0         0   -8.9443         0
         0         0         0   -0.0000
```

Vector/Matrix functions

Matlab Function	Description
dot(a,b)	Dot product of two matrices/vectors
cross(a,b)	Cross product
norm(a)	Vector/Matrix norm
trace	Sum of diagonal elements
det	Determinant of matrix
cond	Matrix condition number
svd	Singular value decomposition

Polynomials in Matlab

Polynomial Functions

Matlab Function	Description
polyval(p,x)	Evaluate polynomial whose coefficients are by array p, at scalar value x
roots(p)	Evaluate roots of polynomial p
polyint(p)	Polynomial integration
polyder(p)	Polynomial derivative
polyfit	Determine least-squares polynomial fit of data
conv()/deconv()	Polynomial multiplication/division

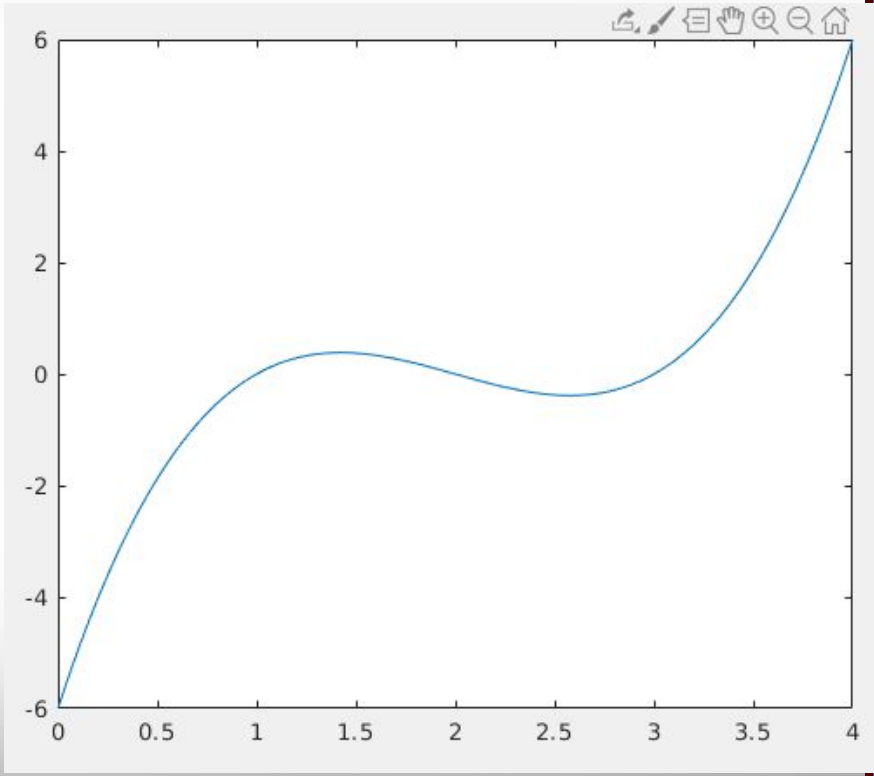
Polynomial evaluation

$$p(x) = x^3 - 6x^2 + 11x - 6 = 0$$

```
>> p = [1 -6 11 -6];
>> polyval(p, 0)      % evaluate p(x) @ x=0
ans = -6

>> x=[0:0.01:4];
>> y=polyval(p, x);
>> plot(x,y)
```

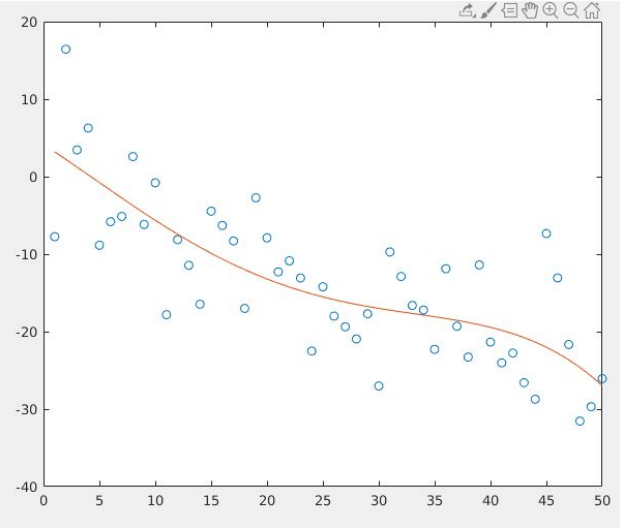
Polynomial roots

```
>> roots(p)

ans =

    3.0000
    2.0000
    1.0000
```



Polynomial fit

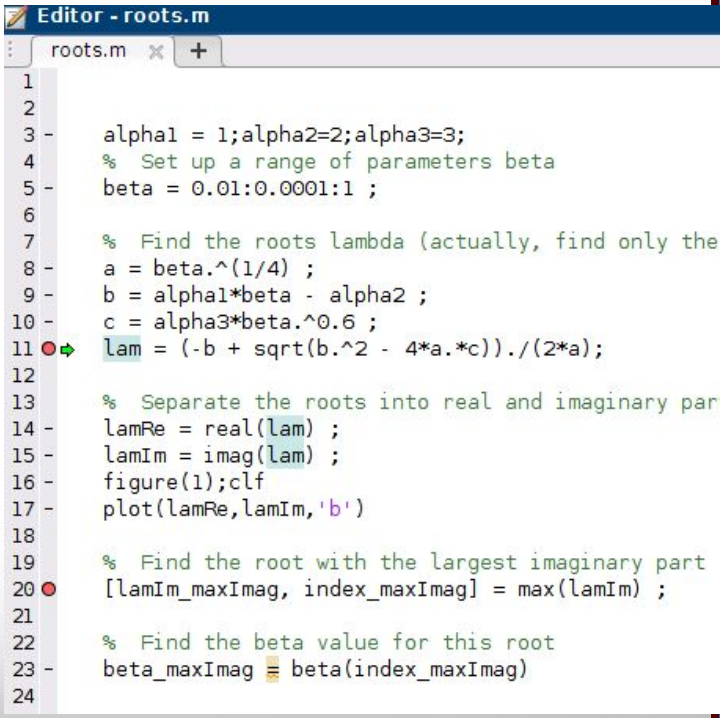
```
>> x = 1:50;
>> y = -0.5*x + 6*randn(1,50);
>> p = polyfit(x,y,4);
>> f = polyval(p,x);
>> plot(x,y,'o')
>> hold on;
>> plot(x,f,'-')
```



Debugging in Matlab

Using breakpoints

1. Click on the dash (-) in the breakpoint alley next to line number or press F12
2. Run script or call function
3. Control (green arrow) pauses at breakpoint
4. Investigate values and continue running script with optionally more breakpoints



Debugging toolbar

Toolbar Button
 Run to Cursor
 Step
 Step In
 Continue
 Step Out
 Pause
 Quit Debugging

Matlab File I/O

Matlab format files .mat

```
>> save('data.mat')
>> save('data.mat','var1','var2')


>> load('data.mat')

>> whos('-file','data.mat')
```

Readmatrix/Writematrix

Supported Formats

-- .txt, .dat, or .csv for delimited text files

-- .xls, .xlsb, .xlsm, .xlsx, .xltm, .xltx, or .ods for spreadsheet files

```
>> M = magic(4)

M =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
>>writematrix(M,'mat_M.txt','Delimiter',
'tab')
>> A = readmatrix('mat_M.txt')
```

Low-level text file I/O

Matlab Function	Description
fopen	Open file, or obtain information about open files
fprintf	Read data from binary file
fscanf	Write data to binary file
fgets/fgetl	Read line from file, keeping/removing newline characters
feof	Check for End of File
fclose	Close file

Text file write - fprintf

```
>> fileID = fopen('test.txt','w')

>> fprintf(fileID,'Hello world\n')

>> fprintf(fileID,'%d %f %s\n',a(1),b(1),c{1})

>> fprintf(fileID,'%d %f %s\n',a(2),b(2),c{2})

>> fclose(fileID);
```

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

>> b = [0.5:0.25:1.0]

>> c ={'One','Two','Three'}
```

Text file write - fprintf

```
>> fileID = fopen('test.txt','r')

>> fscanf(fileID,'%d %f %s\n')

>> fclose(fileID);
```

Binary file I/O

Matlab Function	Description
fopen	Open file, or obtain information about open files
fread	Read data from binary file
fwrite	Write data to binary file
fseek	Move to position in file
feof	Check for End of File
fclose	Close file

Binary file write

```
>> fileID = fopen('myfile.dat','w');

>> fwrite(fileID,a,'int');

>> fwrite(fileID,b,'double');

>> fclose(fileID)

>> dir('myfile.dat').bytes

ans = 36
```

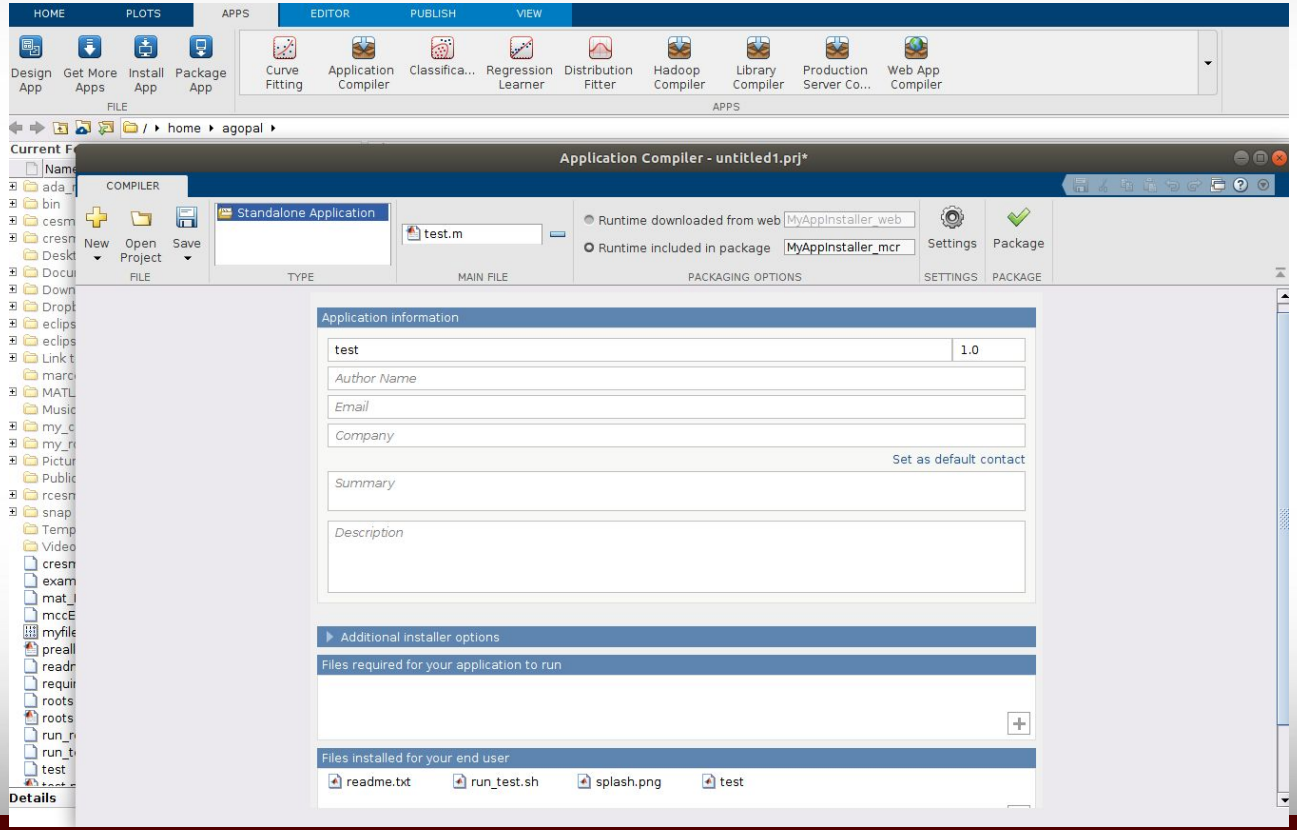
```
>> a = [1 2 3]

>> b = [0.5 0.75 1.0]
```

Binary file read

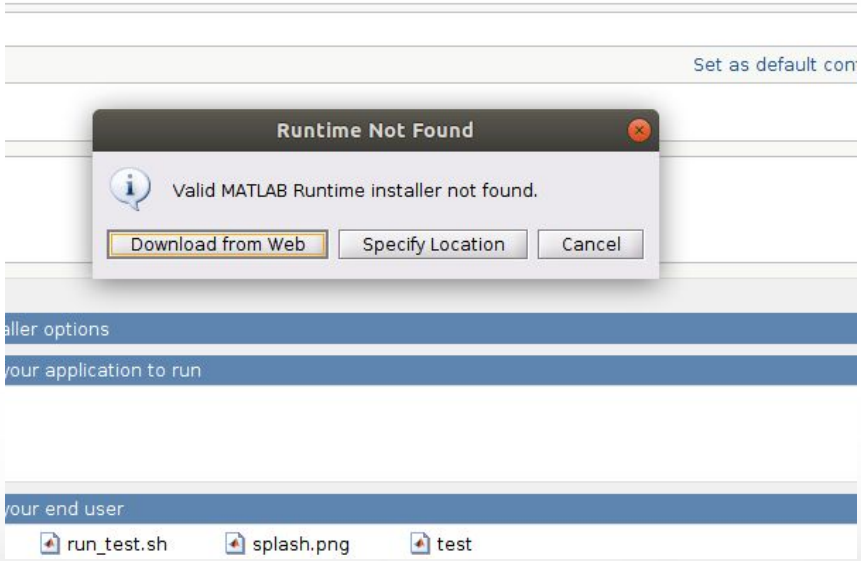
```
>> fileID = fopen('myfile.dat','r');

>> fread(fileID)     % try it


>> fread(fileID,3,'int')

>> fread(fileID,3,'double')
```

Matlab compiler

Create standalone matlab programs



Matlab runtime



From the command line - mcc

```
>> mcc -m script.m
```

On linux shell

```
    $LD_LIBRARY_PATH=/home/agopal/MATLAB/MATLAB_Runtime/v97/bin/gl
nxa64:/home/agopal/MATLAB/MATLAB_Runtime/v97/sys/os/glnxa64:LD_LIBR
ARY_PATH
    $export LD_LIBRARY_PATH

    $./script
```

Appendix