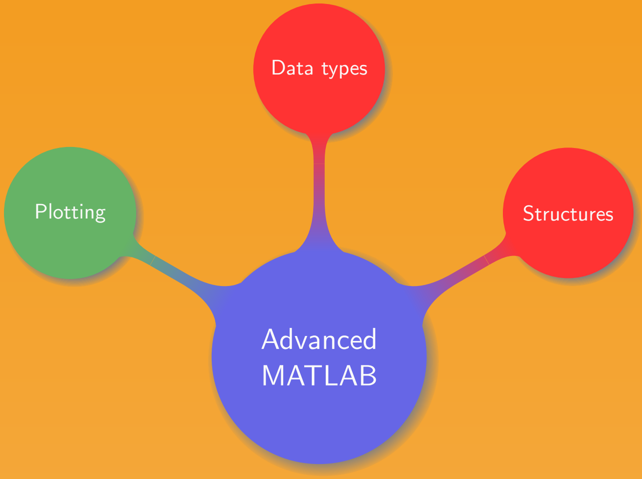
Introduction to Computer and Programming

4. Advanced MATLAB

Manuel – Summer 2019

Chapter organisation



General plotting process

Simple workflow:

1. Use plotting tools or functions to create a graph
2. Extract data info/perform data fitting
3. Edit components (axes, labels...)
4. Add labels, arrow
5. Export, save, print...

2D plotting

Basic plotting functions:

- Plot the columns of x, versus their index: `plot(x)`
- Plot the vector x, versus the vector y: `plot(x,y)`
- Plot function between limits: `fplot(f,lim)`
- More than one graph on the figure: `hold`

Plotting properties:

- Axis properties: `axis`
- Line properties: `linespec`
- Marker properties

Example

Explain each of the following commands:

```
y=exp(0:0.1:20);plot(y);
x=[0:0.1:20];y=exp(x);plot(x,y);
x=[-4:0.1:4];y=exp(-x.^2);plot(x,y,'-or');
hold on;
%fplot('2*exp(-x^2)',[-4 4]);
fplot(@(x)2.*exp(-x.^2))
hold off;
f=@(x) sin(1./x)
fplot(f,[0 .5])
hold;
fplot(f,[0 0.5],10000,'--r')
```

3D plotting

Study data in more than one dimension:

- Visualise functions of two variables
- Create a surface plot of a function
- Display the contour of a function

3D plotting

Study data in more than one dimension:

- Visualise functions of two variables
- Create a surface plot of a function
- Display the contour of a function

Example.
For $t \in [0, 2\pi]$ display the curve parametrised by

$$\begin{cases} x(t) = \sin(2t) + 1 \\ y(t) = \cos(t^2) \end{cases}$$

```
t=0:.01:2*pi;
x=sin(2.*t)+1;
y=cos(t.^2);
plot3(x,y,t);
```

Example

Process 3D plotting:

1. Define the function
2. Set up a mesh
3. Display the function

Display functions:

- Contour: `contour(x,y,z)`
- Color map: `pcolor(x,y,z)`
- 3D view: `surf(x,y,z)`

Example

Process 3D plotting:

1. Define the function
2. Set up a mesh
3. Display the function

Display functions:

- Contour: `contour(x,y,z)`
- Color map: `pcolor(x,y,z)`
- 3D view: `surf(x,y,z)`

Explain each of the following commands:

```
[x,y]=meshgrid(-4:0.1:4);
z=(x.^2-y.^2).*exp(-(x.^2+y.^2));
pcolor(x,y,z);
contour(x,y,z);
surf(x,y,z);
shading interp;
colormap gray;
```

More plotting

2D plotting:

- Bar graph: `bar(x,y)`
- Horizontal bar graph: `barh(x,y)`
- Pie chart: `pie(x)`

3D plotting:

- 3D bar graph: `bar3(x,y)`
- 3D horizontal bar graph: `bar3h(x,y)`
- 3D pie chart: `pie3(x)`

Other useful functions:

- Polar graph: `polar(t,r)`
- More than one plot: `subplot(mnp)`

Interpolation

Goals of interpolation:

- Draw a smooth curve through known data points
- Use this curve to approximate unknown values in other points

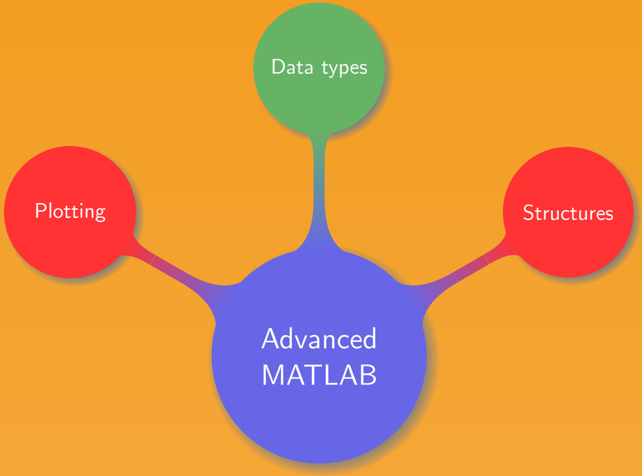
Interpolation in MATLAB:

- 2D: `interp1(X,Y,xi,m)`
- 3D: `interp2(X,Y,Z,xi,yi,m)`

Example.

```
X=[0:3:20]; Y=[12 15 14 16 19 23 24];
interp1(X,Y,4.1)
plot(X,Y,'*')
hold;
xi=[4.1 5.3 8.2 12.6];
yi=interp1(X,Y,xi);
plot(xi,yi,'or');
```

Chapter organisation



Main problematic

So far in MATLAB we:

- Focused on high level problems
- Did not address the internal mechanisms of the program

Not all the data is the same:

- How information is represented in the computer
- Determine the amount of storage allocated to a type of data
- Methods to encode the data
- Available operations on that data

Why data types?

From mathematics to computer science:

- Different numbers (integer, real, complex, etc.)
- Different ranges (short, long, etc.)
- Different precisions (single, double, etc.)

Why data types?

From mathematics to computer science:

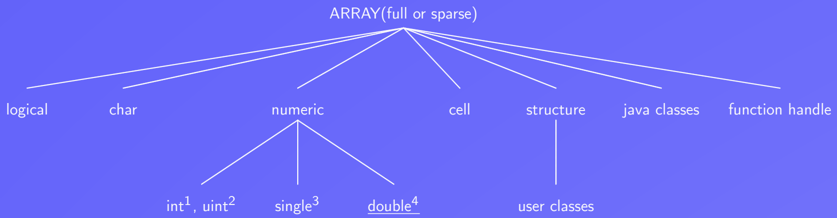
- Different numbers (integer, real, complex, etc.)
- Different ranges (short, long, etc.)
- Different precisions (single, double, etc.)

Example.

Representing signed integers over 8 bits:

1. Signed magnitude: 7 bits for the numbers, 1 bit for the sign
2. Two's complement: invert all the bits of a, add 1 to get $-a$
   e.g. $00101010 \rightarrow 11010101 + 1 = 11010110$
   $00101010 = -0 \cdot 2^7 + 2^5 + 2^3 + 2 = 42$
   $11010110 = -1 \cdot 2^7 + 2^6 + 2^4 + 2^2 + 2 = 86 - 128 = -42$

Data types in MATLAB



1. int: int8, int16, int32 and int64
2. uint: unit8, uint16, uint32 and uint64
3. 32bits; `realmax('single')`, `realmin('single')`
4. 64 bits; `realmax`, `realmin`

Type related functions

Type of a variable:

- `whos`
- `isnumeric`
- `isreal`
- `isnan`
- `isinf`
- `isfinite`

Type related functions

Type of a variable:

- `whos`
- `isnumeric`
- `isreal`
- `isnan`
- `isinf`
- `isfinite`

Numeric conversions:

- `cast(a,'type')`
- `uint8(a)`

Type related functions

Type of a variable:

- `whos`
- `isnumeric`
- `isreal`
- `isnan`
- `isinf`
- `isfinite`

Numeric conversions:

- `cast(a,'type')`
- `uint8(a)`

Useful string functions:

- `isletter`
- `isspace`
- `strcmp(s1,s2)`
- `strcmpi(s1,s2)`
- `strncmp(s1,s2,n)`
- `strncmpi(s1,s2,n)`
- `strrep(s1,s2,s3)`
- `strfind(s1,s2)`
- `findstr(s1,s2)`
- `num2str(a,'format')`
- `str2num(s)`

String parsing

Exercise.

Input two numbers as strings and calculate their sum

String parsing

Exercise.
Input two numbers as strings and calculate their sum

```
clear all, clc;
numbers=input('Input two numbers: ', 's');
space=strfind(numbers,' ');
number1=str2num(numbers(1:space-1));
number2=str2num(numbers(space+1:end));
number1+number2
```

Understanding the code:

- What is this code doing?
- How are `strfind`, and `str2num` used?
- What is `space` containing, and how is it used?

Binary file functions

Working with a binary file:

- Read: `fread(fd,count,'type')`, read `count` elements as `type`
- Write: `fwrite(fd, A, 'type')`, write `A` as `type`
- Position in a file: `ftell(fd)`
- Jump in a file: `fseek(fd,offset,'origin')`, move by `offset` bytes, starting at `origin`

Binary file functions

Working with a binary file:

- Read: `fread(fd,count,'type')`, read `count` elements as `type`
- Write: `fwrite(fd, A, 'type')`, write `A` as `type`
- Position in a file: `ftell(fd)`
- Jump in a file: `fseek(fd,offset,'origin')`, move by `offset` bytes, starting at `origin`

Example.

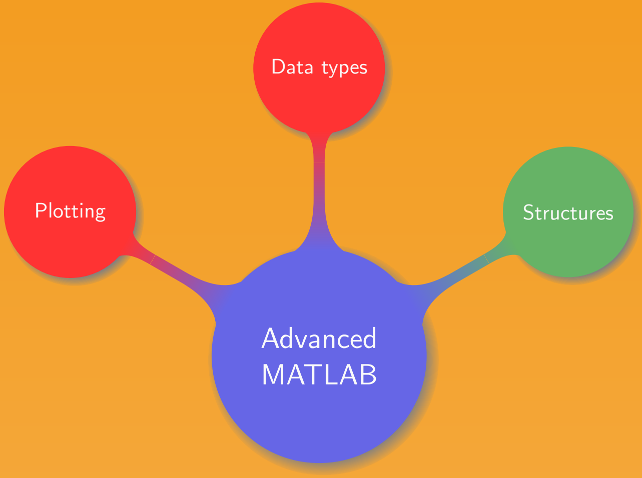
```
A=3:10;
fd=fopen('test','w'); fwrite(fd,A,'int32');
fclose(fd);
fd=fopen('test','r'); fseek(fd,4*4,'bof');
fread(fd,4,'int32'), ftell(fd)
fseek(fd,-8,'cof');fread(fd,4,'int32')
fclose(fd);
```

Binary file functions

Alter the previous sample code and explain its behaviour:

- Define a different `A`
- Display the type of `A`
- Read the numbers as `int64`
- Write the numbers as `double` and read them as `int8`
- Consecutively display the first and fourth elements

Chapter organisation



What is a structure?

Structure:

- Array with “named data containers” called fields
- A fields can contain data of any type

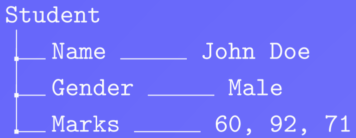
What is a structure?

Structure:

- Array with “named data containers” called fields
- A fields can contain data of any type

Example.

A student is defined by a `name`, a `gender`, and some `grades`. We can represent a student in the form of a “tree” or organise many students in an array.



Student

Name John Doe

Gender Male

Marks 60, 92, 71

Name	Gender	Marks
Iris Num	F	30 65 42
Jessica Wen	F	98 87 73
Paul Wallace	M	65 73 68

Structures in MATLAB

Exploiting the power of structures:

1. Initializing the structure

```matlab
student(1)= struct('name','iris num', 'gender',...
 'female', 'marks', [30 65 42]);
student(2)= struct('name','jessica wen',...
'gender', 'female', 'marks', [98 87 73]);
student(3)= struct('name','paul wallace',...
'gender', 'male','marks', [65 72 68]);
```

Structures in MATLAB

Exploiting the power of structures:

1. Initializing the structure

```
student(1)= struct('name','iris num', 'gender',...
 'female', 'marks', [30 65 42]);
student(2)= struct('name','jessica wen',...
'gender', 'female', 'marks', [98 87 73]);
student(3)= struct('name','paul wallace',...
'gender', 'male','marks', [65 72 68]);
```

2. Using the structure

```
student(3).gender
mean([student(1:3).marks])
```

3. Who got the best mark?

Structures in MATLAB

Exploiting the power of structures:

1. Initializing the structure

```
student(1)= struct('name','iris num', 'gender',...
 'female', 'marks', [30 65 42]);
student(2)= struct('name','jessica wen',...
'gender', 'female', 'marks', [98 87 73]);
student(3)= struct('name','paul wallace',...
'gender', 'male','marks', [65 72 68]);
```

2. Using the structure

```
student(3).gender
mean([student(1:3).marks])
```

3. Who got the best mark?

```
[m,i]=max([student(1:3).marks]);
student(ceil(i/3)).name
```

Key points

- Using `plot` draw simple geometrical shapes
- How to keep or erase previous graphs?
- How to measure the quality of a fit?
- Cite the most common data types and their size in bytes
- What is a data structure?

Thank you!