University of Nottingham

Computer Programming with MATLAB

MM1CPM - Lecture 6

Matricies and conditional execution of code

Dr. Roderick MacKenzie
roderick.mackenzie@nottingham.ac.uk
Autumn 2014

www.mm1cpm.com

Outline

- Recap of last lecture
- Matrices in MATALB
- Conditional execution of code
  - if statements
  - Nested if statements
- Summary

Recap: Loops in MATALB

Until the last lecture if we wanted to get the computer to repeat a command we had to copy and paste it many times:

```
%Script to print Hello!! x100
disp('Hello!!');
disp('Hello!!');
disp('Hello!!');
disp('Hello!!');
disp('Hello!!');
disp('Hello!!');
disp('Hello!!');
.... repeat 93 more times..
disp('Hello!!');
```

→

```
Hello!!
Hello!!
Hello!!
Hello!!
Hello!!
Hello!!
Hello!!
...............
Hello!!
```

Last lecture we learnt there is a better way to get a computer to repeat a command....

Recap: The **for** loop

Start the for loop

Count using the variable n

Count from 1 to 100

Loop

```
for n=1:100

        disp('Hello!!')          %print to the screen

end
```

End the for loop, if **'n'** has reached 100 stop counting, if not go to the start of the loop.

Recap: **while** loops example in MATLAB

- We also learnt about while loops, while loops run whilst something is true.
- They give you more control than for loops – more complex.

Loop

```
t=0                    %set 't' to zero
while (t<10)           %start of while loop
    t=t+0.5            %add one to 't'
    disp(t)            %print t to the screen
end                    %go to the top of while loop if 't'<10
```

- The result would be:

```
0.5
1.0
1.5
.....
10
```

the program counts to ten in steps of 0.5

Recap: Nested loops

- Often in engineering you will need to put one loop inside another loop

Outer loop

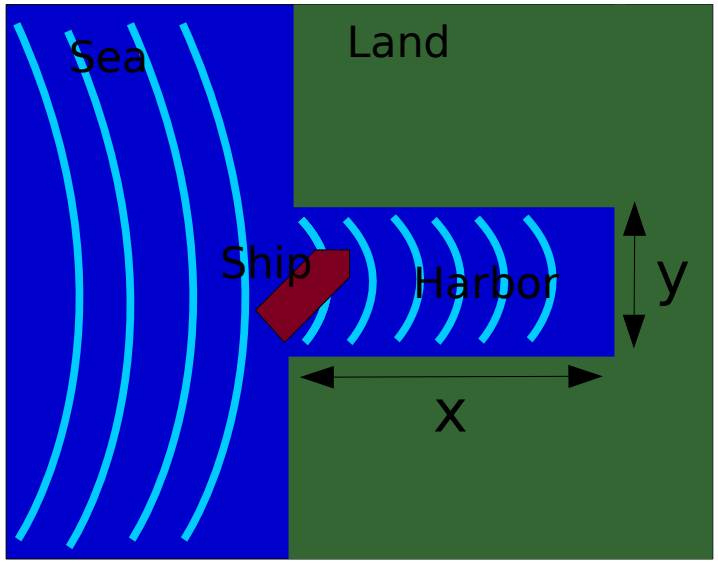
Inner loop

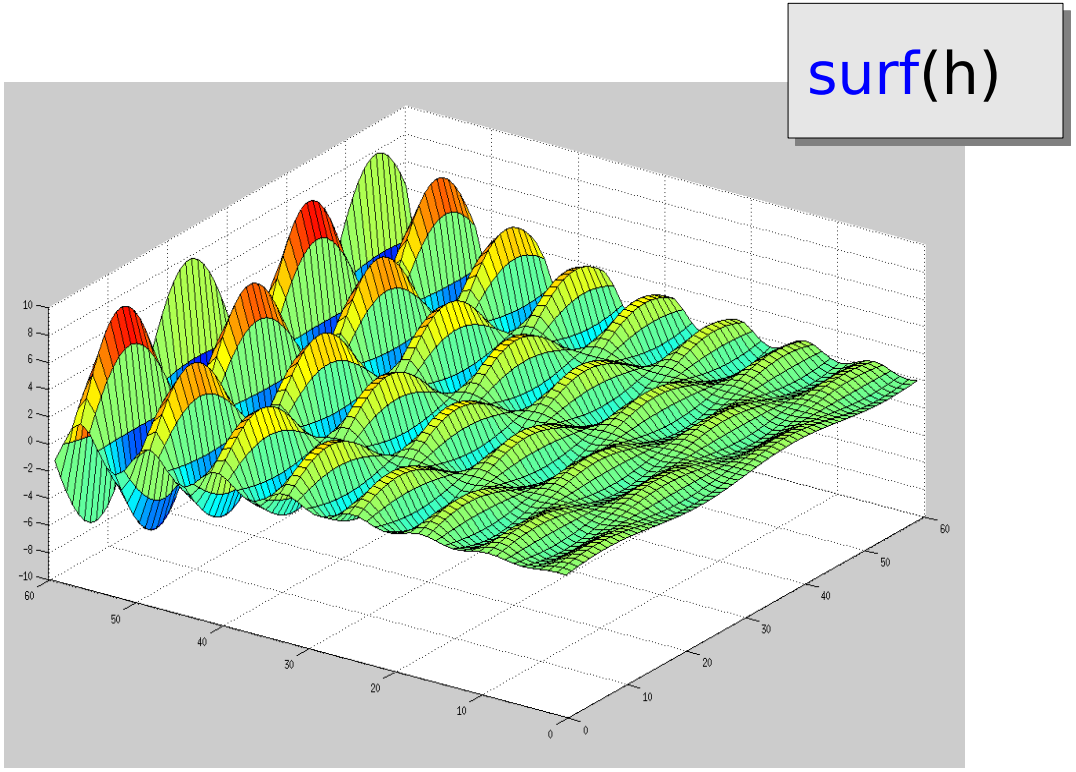
```
for x=1:5  %count using x from 1 to 5

    for y=1:5  %count using y from 1 to 5
    a=sprintf('x=%d y=%d',x,y)
    disp(a)
    end

end
```

x=1 y=1	x=2 y=1	x=3 y=1	x=4 y=1	x=5 y=1
x=1 y=2	x=2 y=2	x=3 y=2	x=4 y=2	x=5 y=2
x=1 y=3	x=2 y=3	x=3 y=3	x=4 y=3	x=5 y=3
x=1 y=4	x=2 y=4	x=3 y=4	x=4 y=4	x=5 y=4
x=1 y=5	x=2 y=5	x=3 y=5	x=4 y=5	x=5 y=5

Recap: Evaluating equations in 2D space

$$h(x,y)=10\sin(x\,0.8)\sin(y\,0.2)\exp((x-60)*0.05)$$





Example: Waves in a harbor.

Outline

- Recap of last lecture
- **Matrices in MATALB**
- Conditional execution of code
  - if statements
  - Nested if statements
- Summary

Making mathematics easy with MATLAB

I have been chatting to your maths lecturer ***Dr. Richard Tew***

He said that he has been teaching you matrices:

- **Adding matrices**
- **Subtracting matrices**
- **Determinants**
- **Inverting matrices**

- I will now teach you how to do all this in MATLAB in a **very easy way** – this should make **your life** as an engineer **much easier**.
- You will also be able to **solve much bigger** problems than you could with pen and paper

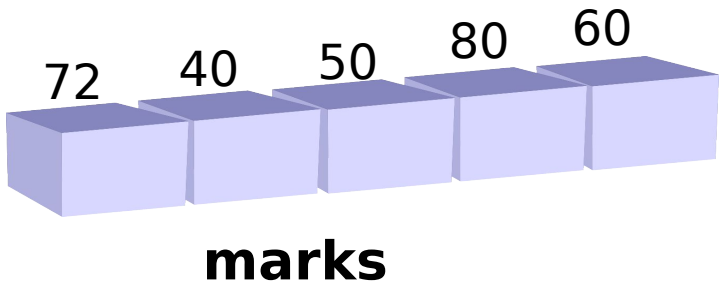
Matrices

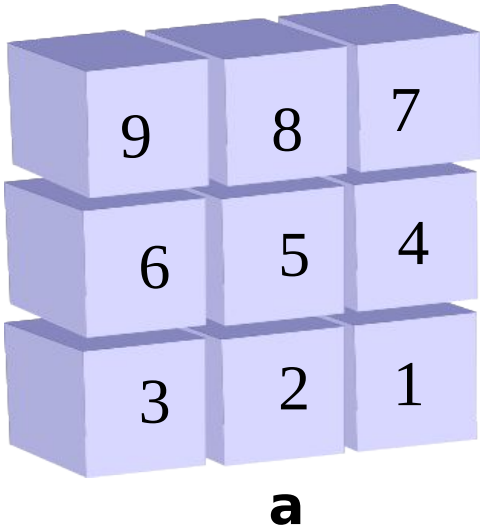
- In mathematics Dr Tew has taught you about **matrices** which look like this:

$$marks = \begin{bmatrix} 72 & 40 & 50 & 80 & 60 \end{bmatrix} \qquad a = \begin{bmatrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}$$

Arrays

- I have been teaching you about **arrays** which look like this:






```
>marks = [ 72 40 50 80 60]
```

```
>a = [ 9 8 7 ; 6 5 4 ; 3 2 1 ]
```

; ;

- Do you spot any similarity?

Arrays=Matrices

- **Arrays** are the same thing as **Matrices** – there is no difference at all!
- **Array** is the word used in computing. **Matrix** is the word used in Mathematics
- In fact the name **MATLAB** comes from the two words **MAT**rix **LAB**oratory.
- This suggests **MATLAB** may be very useful for working with Matrices.
- Let's have a look at matrix algebra in MATLAB – this will speed up your mathematics...

Arrays and Matrices

1) Matrix multiplication

2) Calculating matrix inverse

3) Adding and subtracting matrices

4) Calculating matrix determinant

5) Calculating matrix transpose

- You really know how to do the mathematics
- You already know about handling arrays.
- I'm just going to joint the concepts together.

Mathematical operations in MATLAB

- Let's remind our selves how we do mathematics with normal numbers in MATLAB (you should be good at this).

```
>8*3        <enter>        %multiplying
>7/10       <enter>        %dividing
>7^3        <enter>        %raise to the power
>3+7        <enter>        %adding
> 3-7       <enter>        %subtracting
>(3+7)/4    <enter>        %brackets
```

- The good news is that all these operations work on matrices

Matrix multiplication in MATLAB

Imagine you wanted to multiply matrix **x** by matrix **y**:

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

We could do it like this.....

Matrix multiplication by hand

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

We ***could*** do it by hand like this:

xy=z

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

$$= \begin{bmatrix} 14 & 32 \\ 32 & 77 \\ 50 & 122 \end{bmatrix}$$

But that looks like a lot of work.....

Matrix multiplication by hand

- How would you write **7 multiplied by 3** in MATLAB?

- How would you write **variable x multiplied by variable y** in MATLAB?

- Can you guess how you would write **matrix x multiplied** by **matrix y** in MATLAB?

Matrix multiplication in MATLAB

- Let's get MATLAB to do the hard work for us
- Define x,y and then multiply them with the * **operator**

```
>x = [ 1 2 3 ; 4 5 6 ; 7 8 9 ]          %define matrix x

>y = [1 4; 2 5 ; 3 6]                   %define matrix y

>z=x*y                                  %do the multiplication
 z =
   14    32
   32    77
   50   122
```

- That's it. This will work for arrays of any size.....

Multiplying BIG matricies is just as easy.....

```
> a=rand(6,6)
a =
  0.020765  0.271189  0.632376  0.729180  0.566901  0.819066
  0.368701  0.912497  0.376819  0.449630  0.905317  0.719376
  0.848678  0.547864  0.290577  0.252527  0.695072  0.726722
  0.279395  0.974452  0.654979  0.162743  0.383367  0.884372
  0.919076  0.720000  0.187671  0.771521  0.256806  0.944307
  0.069884  0.622976  0.056963  0.686076  0.987363  0.014216

> b =rand(6,1)
  0.7765851
  0.9124409
  0.9125331
  0.8954564
  0.4884701
  0.9482281
  0.4786666
>c=a*b
```

Multiplying BIG matricies is just as easy.....

```
> a=rand(6,6)
a =
  0.020765  0.271189  0.632376  0.729180  0.566901  0.819066
  0.368701  0.912497  0.376819  0.449630  0.905317  0.719376
  0.848678  0.547864  0.290577  0.252527  0.695072  0.726722
  0.279395  0.974452  0.654979  0.162743  0.383367  0.884372
  0.919076  0.720000  0.187671  0.771521  0.256806  0.944307
  0.069884  0.622976  0.056963  0.686076  0.987363  0.014216

> b =rand(6,1)
  0.7765851
  0.9124409
  0.9125331
  0.8954564
  0.4884701
  0.9482281
  0.4786666
>c=a*b
```

```
c =
  1.5222
  2.0624
  1.7963
  1.9112
  2.0115
  1.3289
```

All you have to do is be able to type it in and MATLAB will do the hard work.

Arrays and Matrices

1) Matrix multiplication
2) **Calculating matrix inverse**
3) Adding and subtracting matrices
4) Calculating matrix determinant
5) Calculating matrix transpose

Calculating the inverse of a matrix

- In mathematics you were taught to calculate the inverse of a matrix like this:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$A^{-1} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^{-1} = \frac{-1}{\det(A)} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix} = \frac{-1}{1 \cdot 4 - 2 \cdot 3} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix}$$

- Only works for 2x2 matrices
- Again impractical to do by hand once the problem gets big.

Calculating the inverse of a matrix

- How would you write the 7^{-1} in MATLAB? i.e. inverse of 7.

- How would you write x^{-1} in MATLAB? i.e. the inverse of x.

- Can you guess how you would write x^{-1} where x is an **array** i.e. the inverse of matrix x.

Calculating the inverse of a matrix **(^-1)**

- In MATLAB you would just type

```
>x = [ 1 2 ; 3 4 ]          %define matrix x
    x=1 2
       3 4

>y=x^-1                     %take the inverse

y=
 -2.0000    1.0000
  1.5000   -0.5000
```

Again this will work on any size matrix.

Arrays and Matrices

1) Matrix multiplication

2) Calculating matrix inverse

3) Adding and subtracting matrices

4) Calculating matrix determinant

5) Calculating matrix transpose

Adding (+) and subtracting (-) matrices

In Mathematics you learnt:

Adding:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 1 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

Subtracting:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

In MATLAB it's the same....

Adding (+) and subtracting (-) matrices

The add (+) and subtract (-) operators also work on matrices:

```
>a= [ 1 0 1 0 1 0 1 0 ]

>b= [ 2 2 2 2 2 2 2 2 ]

>c=a+b
  c= [ 3 2 3 2 3 2 3 2 ]

>c=a-b
  c=[-1 -2 -1 -2 -1 -2 -1 -2]
```

This will also work on 2D and 3D arrays.

Arrays and Matrices

1) Matrix multiplication

2) Calculating matrix inverse

3) Adding and subtracting matrices

4) Calculating matrix determinant

5) Calculating matrix transpose

Calculating the determinant of a matrix

In mathematics you have been taught, to calculate the determinant of a matrix in the following way:

2x2 determinant

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

3x3 determinant

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = a\begin{vmatrix} e & f \\ h & i \end{vmatrix} - b\begin{vmatrix} d & f \\ g & i \end{vmatrix} + c\begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

$$= a(ei - fg) - b(di - fg) + c(dh - eg)$$

What a lot of work!.....

determinant using MATLAB (the det command)

- Again, let's get MATLAB to do the hard work for us.
- Just define the array (matrix) and use the determinant command to calculate the determinant:

2x2 determinant

```
>a=[ 1 2 ; 3 4]

    a= 1 2
       3 4

>det(a)
   ans = -2
```

3x3 determinant

```
>a=[ 1 2 3 ; 4 5 6 ; 7 8 9 ]

    a= 1 2 3
       4 5 6
       7 8 9

>det(a)
   ans = 6.6613e-16
```

- Again this works on any size matrix.

Calculating the determinant of a BIG matrix

- Is now effortless...

```
>a=rand(9,9)
0.871155  0.466772  0.367611  0.670919  0.749830  0.225811  0.385159  0.431584  0.129268
0.784839  0.505594  0.377279  0.343956  0.354255  0.083763  0.339341  0.118705  0.941804
0.101405  0.415996  0.401407  0.971288  0.988494  0.596689  0.173983  0.614268  0.743980
0.932109  0.296903  0.368229  0.494855  0.657463  0.446307  0.201825  0.789940  0.902783
0.976103  0.319949  0.487244  0.665471  0.381459  0.744140  0.765919  0.105289  0.758121
0.500984  0.216474  0.076094  0.769049  0.418538  0.139015  0.066101  0.641233  0.112989
0.553294  0.995255  0.290148  0.050998  0.980303  0.215171  0.111843  0.367035  0.601902
0.964871  0.563615  0.035777  0.572351  0.462943  0.420246  0.933567  0.973604  0.608682
0.051684  0.243024  0.517364  0.611405  0.771370  0.309329  0.606791  0.431090  0.267379

>det(a)
   ans = -0.014988
```

- This would have been very difficult to do by hand.

Arrays and Matrices

1) Matrix multiplication

2) Calculating matrix inverse

3) Adding and subtracting matrices

4) Calculating matrix determinant

5) Calculating matrix transpose

Transposing' an array or matrix

In mathematics you have learnt how to perform a matrix transpose:

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

```
>a= [ 1 2 3 ; 4 5 6 ; 7 8 9 ]

> b=a'

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

a' ← transpose operator

Summary of matrix operations

- Matrices are Arrays

- You've used most of these operations before, I have just told you that they also work on Matrices/arrays.

Operation	Sign	Example
Multiplying	*	c=a*b
Determinant	det	c=det(a)
Inverse	^	c=a^-1
Transpose	'	c=a'
Subtraction	-	c=a-b
Adding	+	c=a+b

Outline

- Recap of last lecture
- Matrices in MATALB
- **Conditional execution of code**
  - if statements
  - Nested if statements
- Summary

Conditional execution of code

- Think about these statements:
  - **If** the car crashes inflate the airbag
  - **If** a fire is detected in the engine turn off the fuel.
  - **If** the aircraft is on a collision course with the ground sound an alarm.

- These are all called **if** statements, **if** **something is true**, **then do something**.
- These are the statements that give computers intelligence and enable them to make decisions.

A real world example:

- Let's revisit our program in our reentry capsule from lecture 5:

Loop

```
Line 1: answer=am_I_going_slow_enough_to_open_the_parachute
Line 2: if (answer==yes) open_parachute
Line 3: if (answer==no) do_nothing
Line 4: go back to line 1
```

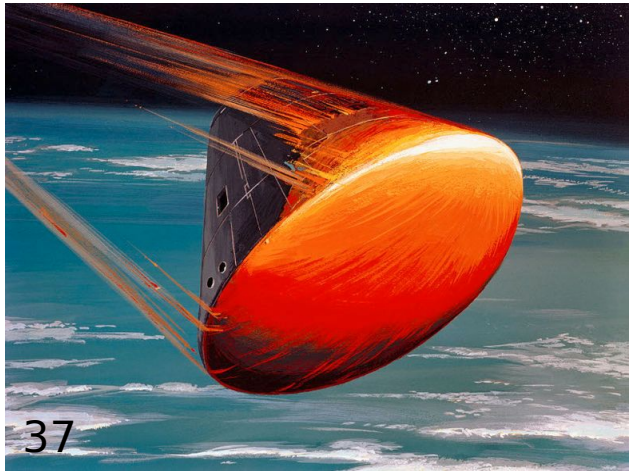
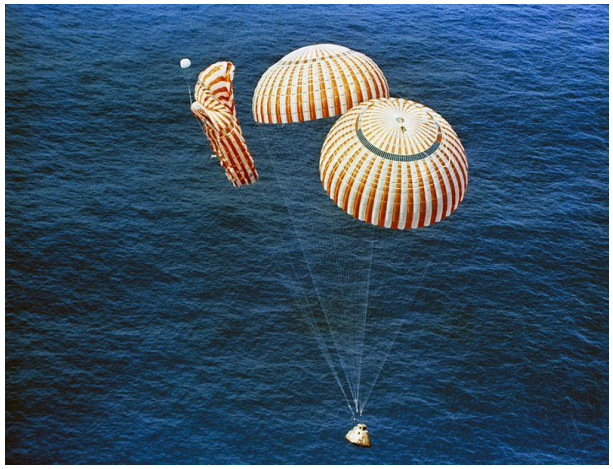


Image from NASA



A second real world example:

Loop

Line 1: answer= **has_the_car_hit_an_object**
Line 2: **if (answer==yes) open_airbag**
Line 3: **if (answer==no) do_nothing**
Line 4: go back to line 1

DaimlerChrysler AG

FAPE

•***All*** decision making in computers is done with these ***if*** statements.

Question:How important are ***if*** statements?

How important do you think your ability to program ***if*** statements are to you future career?

A: Not important

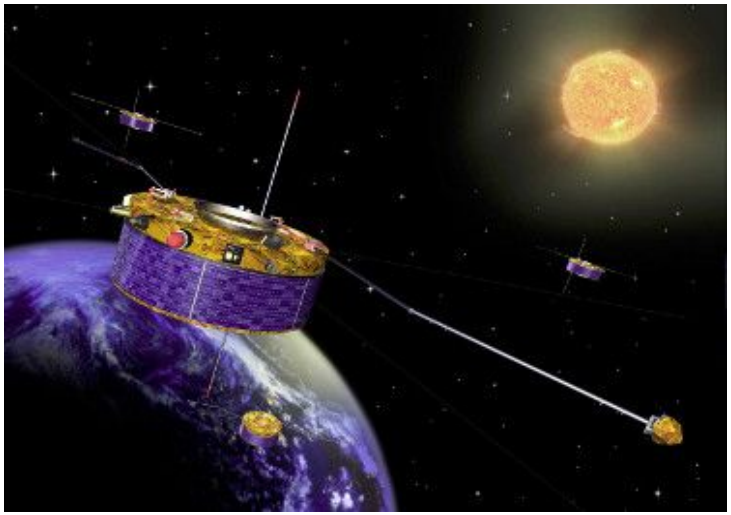
B: Quite important

C: Very important

D: Extremely important – this is the most important lecture I ever going to attend!

Let's find out.....

Getting ***if*** statements right



- ***Cluster*** was a joint **Euopean Space Agency/NASA** satellite launched in 1996 on an Ariane 5 rocket at a cost of **$370** million to study the Earth's magnetic field.

- An engineer made a ***single*** mistake in a single ***if*** statement on the rocket's guidance computer

Youtube

- Let's see what happened.....

What happened?

- The device in the rocket measuring acceleration (**accelerometer**) gave the computer an **unrealistic value of acceleration** – this happens sometimes with sensitive instruments

- However, the engineer ***forgot*** to put this ***if*** statement in the code to check the acceleration was realistic:

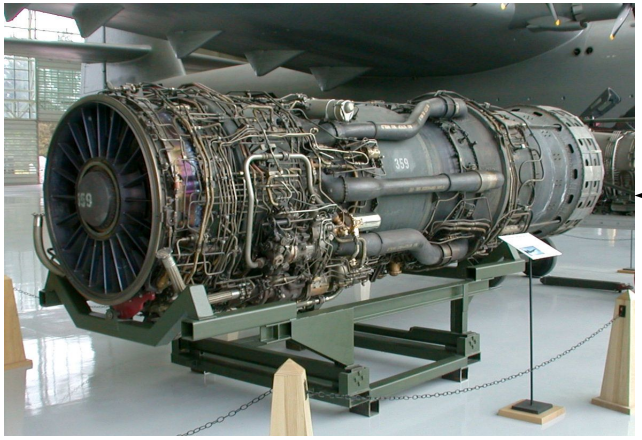
```
if acceleration> 32767 ignore_the_value
```

- The rocket thought it was 90 degrees off course and then tried to suddenly correct its course when it was traveling faster than the speed of sound..and the air flow ripped the rocket to bits....

http://en.wikipedia.org/wiki/Cluster_(spacecraft)

Airbus A320

- Fuel flow on modern airliners is controlled by computer, the pilot just suggests to the computer how much fuel he wants – the computer makes the final decision



Computer

```
if (new_position_of_throttle > old_position_of_throttle)
increase_fuel_flow()
```

How important are **if** statements?

- **If** statements are quite easy to understand and write.

- But if you make a mistake the consequences can be very serious and potentially kill people.

- There have been cases of errors like this in aircraft fly-by-wire systems..




Summary

- Recap of last lecture
- Matrices in MATALB
- Conditional execution of code
  - **if statements**
  - Nested if statements
- Summary

A simple example of an *if* statement in MATLAB

If block

```
speed=80                    %speed in mph
if (speed>70)               %check if speed bigger than 70

 disp('Too fast.')          %print 'Too fast'
 disp('Slow down!')

end                         %end of if statement
```

- **If** the condition is true, all the code between the **if** statement and the **end** will be executed.

FAPE

Youtube example

Conditions

> is called a **conditional test**, this one is called '**bigger than**'

If block

```
speed=80                    %speed in mph
if (speed>70)               %check if speed bigger than 70

 disp('Too fast.')          %print 'Too fast'
 disp('Slow down!')

end                         %end of if statement
```

There are other conditional tests which we can use.....

Other conditional tests

Test	Description	Example
>	Bigger than	if (speed>70)
<	Less than	if (speed<70)
<=	Less or equal to	if (speed<=70)
>=	Greater or equal to	if (speed>=70)
==	Equal to	if (speed==70)
~=	Not equal to	if (speed~=70)

- Where have you seen these conditions before?

The if-else statement

•Often in computing (and life) you will have to decide if you want to do one thing or another:

•**If** I have a coursework deadline go to the library, **else** go to the party.

•**If** I have more than £50,000 buy a Ferrari **else** buy a used Fiat punto.

•These are examples of **if-else** statements, let's have a look at **if-else** statements in MATALB.

The if-else statement in MATLAB

Here is an example of an **if-else** statement in MATLAB:

```
money=100000
if (money>50000)
 disp('Buy a Ferrari ')
else
 disp('Buy a Fiat Punto')
end
```

- **If** the 'money' is over 50000, it prints 'Buy a Ferrari' **else** it prints 'Buy a Fiat Punto'

Youtube example

The **if-elseif-else** statement

You can also join if statements together using the **elseif** statement:

```
speed=60                    %speed in mph
if (speed>70)               % if speed bigger than 70
        disp('Too fast!!')
elseif (speed<30)           %if speed below 30
        disp('Too slow!')
else                        %if neither condition is met
        disp('Speed OK')
end
```

In English: if the speed is **bigger than 70** print 'Too fast', else if the speed is below 30 print 'Too slow!', else neither of these conditions have been met so print 'Speed OK'

The if-elseif-elseif-else statement

You can join as many elseif statements as you like together:

Start of if statement (first logical test)

Next logical test

Another elseif statement

If neither of these conditions are met

End of if block

```
speed=65                    %speed in mph
if (speed>70)               %if speed bigger than 70
        disp('Too fast!!')
elseif (speed<30)           %if speed smaller than 30
        disp('Too slow!')
elseif (speed==65)          %if speed is equal 65
        disp('Just right')
else                        %if it's none of the above
        disp('Speed OK')
end
```

Your go!

- The weight of a muffin on a production line is stored in the variable 'x'.
- If the muffin weighs **more** than **40 grams** it is too heavy
- If the muffin weighs **less** than **30 grams** it is too light
- Otherwise the weight of the **muffin is perfect**.

- Write a program to print '**muffin too heavy**', '**muffin too light**' or '**muffin perfect**' depending upon the content of the variable 'x'.

The Muffin example!!

```
weight=80                    %set weight of muffin
if (weight>40)               %if weight bigger than 40
    disp('Too heavy!')
elseif (weight<30)           %if weight bigger than 30
    disp('Too light')
else                         %if none of the above
    disp('Perfect');
end
```

Summary

- Recap of last lecture
- Matrices in MATALB
- **Conditional execution of code**
  - if statements
  - Nested if statements
- Summary

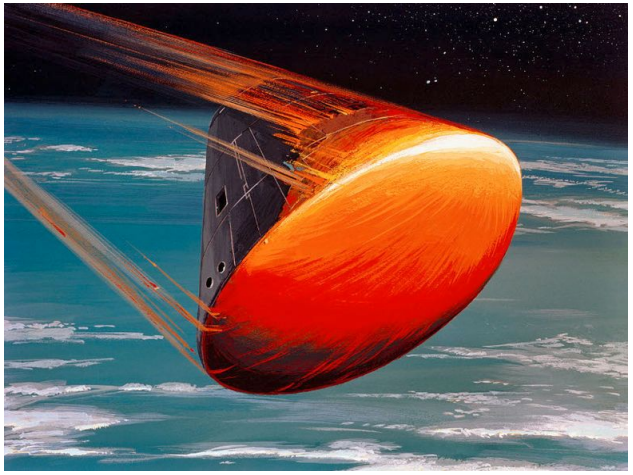
Nested if statements

Just as you can have nested loops you can also have nested **if** statements:

```
speed=100                %(km/h)
altitude=10              %km
if (speed<200)

    if (altitude<20)
        disp('Open parachute')
    end

end
```

If block

If block



The inner **if** statement will only be executed if the outer condition is met. i.e. both conditions have to be true.

Youtube example

Summary

- Recap of last lecture
- Matrices in MATALB
- Conditional execution of code
  - if statements
  - Nested if statements
- **Summary**