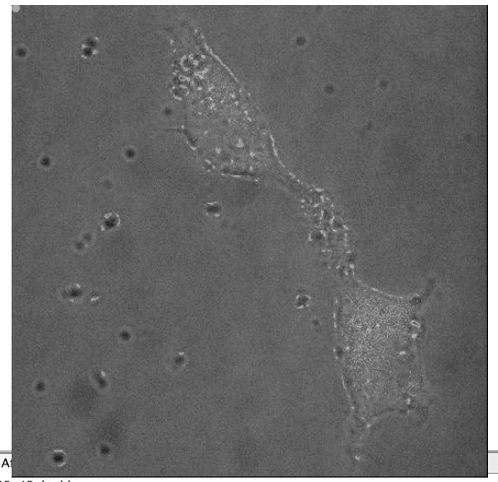
Introduction to Scientific Computation

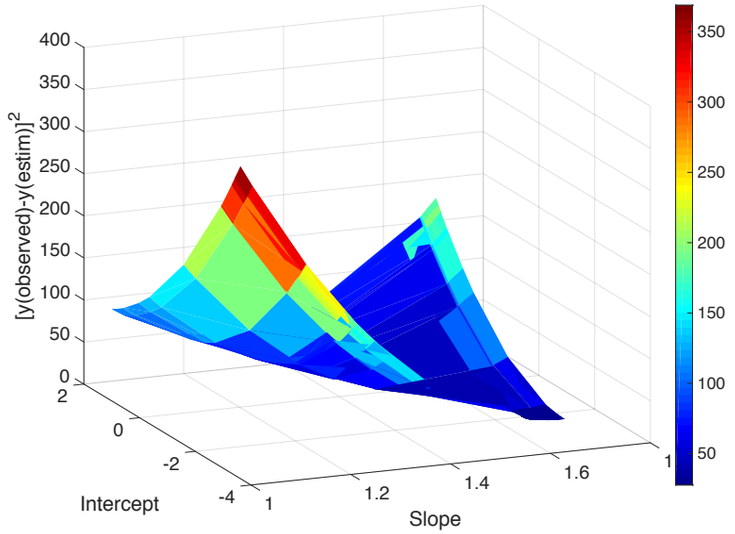


45x45 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	
2	1.8719	2.1662	1.9119	-6.1040	0.9919	0.7463	3.1476	-6.04
3	2.8526	0.2256	3.8202	-4.0920	0.1165	5.7883	12.9021	-8.94
4	2.0403	0.2782	5.8758	-6.0550	1.9590	8.7028	12.9628	-5.26
5	5.0333	-0.8429	7.9372	-4.1242	6.9566	9.8591	19.0487	-2.14
6	6.8161	-0.8040	8.8785	-6.2116	3.0318	12.7550	27.0367	-6.09
7	8.0695	-1.8806	-1.9477	-5.1795	-1.9021	15.9154	26.0704	-11.15
8	6.7938	-1.9829	-5.9837	-9.1850	3.1045	16.0414	30.0233	-16.23
9	6.0202	-4.9705	-13.4202	-7.1922	7.9476	19.9585	30.0196	-20.26
10	6.9395	-2.9546	-13.4397	-1.1495	13.1776	21.9063	27.9988	-19.10
11	7.9033	-1.7517	-13.4291	1.9137	15.1831	19.9469	33.0433	-21.17
12	12.8330	-2.9341	-13.4111	4.9086	16.1398	22.0347	35.8896	-19.14
13	11.8444	-6.8432	-13.3838	12.8698	21.1024	24.0305	38.0739	-20.12
14	13.9453	-14.6963	-13.4790	13.8271	20.1165	23.9439	43.9565	-20.09
15	19.6494	-13.9211	-13.5622	14.9737	22.1298	23.9872	43.1145	-15.14
16	19.7156	-12.7052	-13.8340	15.8757	22.0913	23.0084	45.0050	-14.12
17	22.7534	-16.8870	-13.7940	11.9530	25.1275	21.9375	45.9843	-7.23
18	22.9934	-18.9434	-13.7222	7.0032	25.1389	21.8102	44.0347	-4.17
19	23.8118	-19.9095	-13.7801	4.9072	24.2624	21.9088	40.1089	-0.18
20	20.6912	-17.8886	-14.0669	0.8217	28.0362	21.9693	38.1605	1.74
21	14.9876	-22.5683	-13.8957	-6.0632	28.0769	23.0827	39.0890	3.89
22	12.9788	-17.6657	-13.8064	-7.0523	28.9531	22.9662	39.1144	1.78
23	8.9605	-18.6915	-14.0975	-8.1147	29.2248	20.8487	37.0860	-0.11
24	9.7573	-18.6849	-13.9892	-8.0499	30.1370	18.9655	37.0715	-1.13
25	6.9114	-17.7213	-13.4500	-11.1345	31.2069	17.9289	34.1658	-3.16
26	7.8028	-7.7949	-12.0708	-9.0142	30.1414	17.8286	28.1213	2.87
27	5.7706	-4.8180	-13.0128	-4.9816	29.2002	15.8827	29.0591	3.68

```
fretmoallimgsS = cell(1,1);


for findx1 = 1:1;
    fnamefr1 = flistfrmovie1{findx1};
datA  = zeros(ysize, xsize, nframes);
for k = 1:nframes;
    if mod(k, 10) == 0
        k;
    end
    datA(:,:,k) = imread(fnamefr1, 't
```



Halil Bayraktar

Lecture 8 - Function sorting

Today’s lecture

- Functions review
- Data sorting (bubble and quick sorting)

How to write a function in Matlab

function [output value 1, output value 2, ...]
=name(input1, input2,input3…])

body of the function

end

```
function [result]=name(x,y,z,.....)
% function to compute the factorial of a number

body of the function



end
```

Any matlab function consists of

1. Each function in the matlab starts with a word of “function”, please do not use this word any other place in your code.

2. Find a name for your function
(note: The name should be the same as the name of the M-file in which this function is stored)

3. The input values are shown in parentheses. They are separated by commas if there is more than one input values.

4. Output values are shown in square brackets. If there are more than one inputs, they are separated by comma.

Example: Function to compute the factorial of a number

Solution 1

```
function [result]=factfun(n)
% function to compute the factorial of a number

    result=1;
    for i=1:n;
        result=result*i;

    end
    format short
      %result=fac;
      fprintf('%i factorial equals to %i \n',n,result)
end
```

Solution 2

```
function [result]=factfun(n)
% function to compute the factorial of a number

 result=prod(1:n)
   format short

     fprintf('%i factorial equals to %i \n',n,result)
end
```

Anonymous Functions

Anonymous functions are unnamed function objects defined in your program. to write a a function we use a symbol of @

@(input variable) expression

quad function tells us that it evaluates the integration of a function between two values.

```
a = @(x) sin(x).*cos(x);
      quad(a,0,3)
```

Ans = 0.01

$$I = \frac{2}{2}\int \sin x \cos x dx$$

$$I = \frac{1}{2}\int 2 \sin x \cos x dx$$

$$I = \frac{1}{2}\int \sin 2x dx$$

$$I = -\frac{1}{2}\frac{\cos 2x}{2}$$

$$\boxed{I = \frac{-\cos 2x}{4} + C}$$

Anonymous Functions

```
sqr = @(x) x.^2;
    a = sqr(5)
```

Example 2: How to sort numbers

When dealing with large data sets, we often sort them to assess the data. MATLAB itself has a very powerful function called “sort”. However, there are different ways of sorting numbers.

a. Bubble sorting

b. Quick sorting etc.

Here is the problem:

Write a function to sort the following numbers from smallest to largest value: 7 3 2

X(1) > X(2) >X(3)

Return 2 > 3> 7

Bubble sorting

	7	3	2
Step 1	3	7	2
Step 2	3	2	7
Step 3	2	3	7

Bubble sorting

The bubble sort is a simple sorting algorithm. Although it is very inefficient compared to quick sorting, it is very easy to understand.

```
function [x] = bubblesort2(x)
tic
n=length(x); %n=3
for j = n-1:-1:1; % j=2,1
    for sortnum = 1:j; %j 1,2  then 1
        if x(sortnum) > x(sortnum+1);
            temporary = x(sortnum+1);
            x(sortnum+1) = x(sortnum)
            x(sortnum) = temporary;
        end
        disp(x)
    end
end
toc
end
```

7 3 2

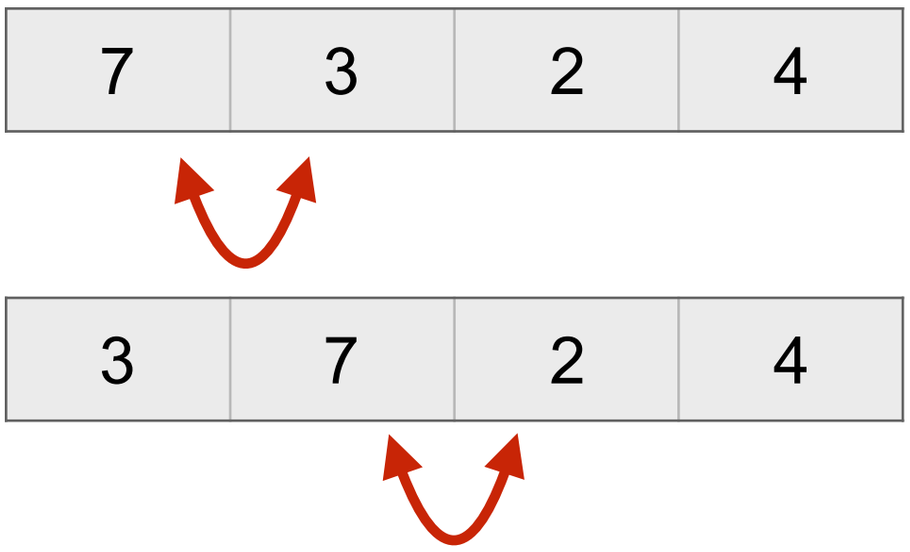
3 7 2

3 2 7

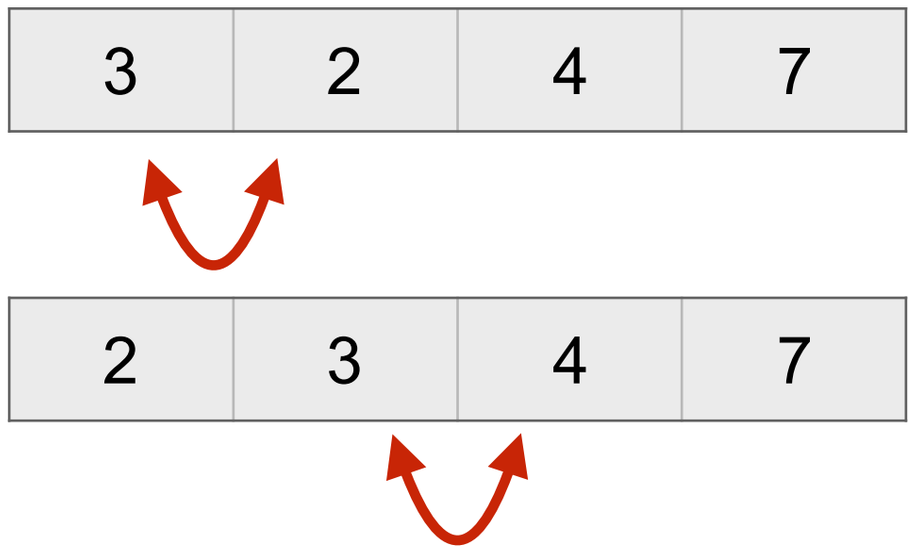
2 3 7

Bubble sorting

step 1



step 2



2. Quick Sorting

This sorting technique is faster and more useful than bubble sorting. It is also known as divide and conquer algorithm.

The following steps are applied for quick sorting:

1. Pick an element and usually called a “pivot number” from an array.

2. Next step is “Partitioning section” means that reorder the array so that all elements with values less than the pivot come before the pivot number , while all elements with values greater than the pivot number come after it. If they are equal, the values can go either way.

3. After this partitioning, the pivot is now in its final position.

4. Then successively apply all the steps shown above to the sub-array of elements with smaller values and separately to the sub-array of elements with greater values.

Quick Sorting

19	32	22	24	17	12	14	15	6

Pivot number = 19

19	32	22	24	17	12	14	15	6

19	32	22	24	17	12	14	15	6

32>19
6<19

19	6	22	24	17	12	14	15	32

22>19
15<19

19	6	15	24	17	12	14	22	32

24>19
14<19

19	6	15	14	17	12	24	22	32

17<19
12<19

STOP

Both numbers are smaller than 19

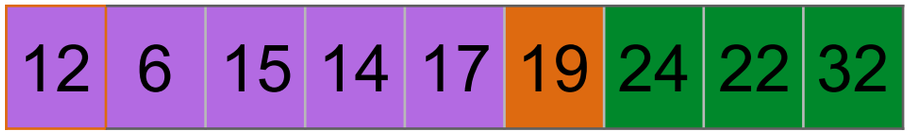
19	6	15	14	17	12	24	22	32

12<19
24>19

NOW splitting 1D array

12	6	15	14	17	19	24	22	32

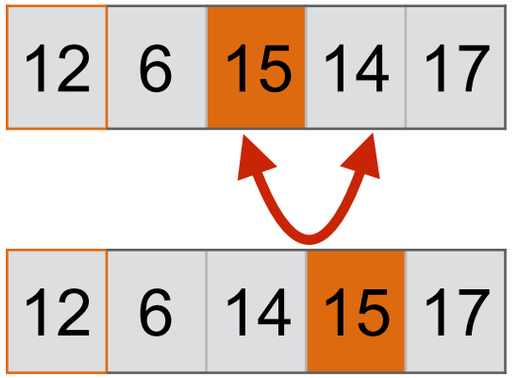
All numbers<19 All numbers>19



All numbers<19 All numbers>19

Choose another pivot number

12<15
17>15

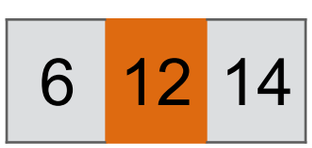


6<15
14<15

Both numbers are smaller than 15

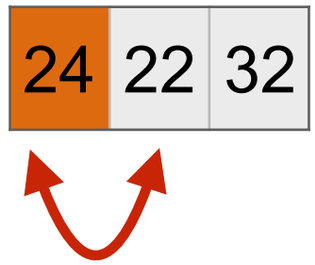
6<12
14>12

Splitting numbers



SORTED

Choose another pivot number



32>24
22<24

SORTED

Result: