Review of Lecture 4

- **Saving data in binary file format**

Save all variables located in the workplace

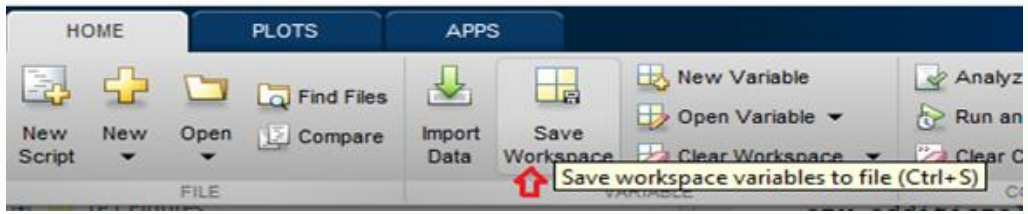
```
save ('file-name.mat') or
save file-name.mat
```

Save only the desired variables

```
save ('file-name', 'var1', 'var2') or
save file-name, var1, var2
```

Add variables to an existing file

```
save('file-name','var','-append')
```



- **Loading data from file in MATLAB**

Reads data from the indicated file `'file-name'`.

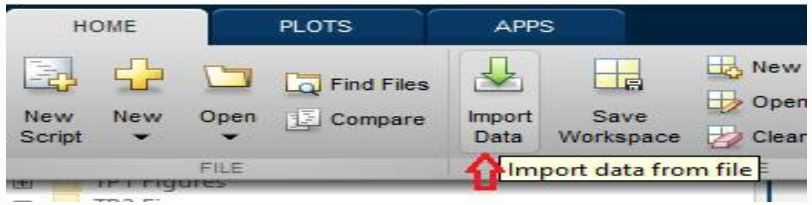
```
load('file-name')
```

Reads data and save it in variable Data.

```
Data = load('file-name')
```

Reads specified variables from a .mat file.

```
load('file-name', 'variables')
```



Info 3

Introduction to MATLAB

M. Bouzenita
2nd year Engineer - University of Jijel

Lecture 5

Vectors and matrices in MATLAB(1/2)

1. Vectors in MATLAB

In MATLAB, a vector represents a one-dimensional array of variables.

The orientation of these variables within the vector distinguishes between two key types:

Row vector

$$\begin{pmatrix} 2 & 1 & 5 \end{pmatrix}$$

Column vector

$$\begin{pmatrix} 2 \\ 1 \\ 5 \end{pmatrix}$$

2. Create a vector in

For creating a **row vector**, simply enclose the elements or variables within square brackets [ ] and separate them using either **commas** or **spaces.**

```
v = [1, 5, 8, 7, 10]
```

```
v = [1  5  8  7  10]
```

For creating a **column vector**, we write the elements of the vector in square brackets and separate them using **semi-colons (;).**

```
v= [1; 5; 8; 7; 10]
```

```
>> % Creating a row vector V
>> V = [1, 5, 8, 7, 10]
V =
     1     5     8     7    10
>> V = [1 5 8 7 10]
V =
     1     5     8     7    10

>> % Creating a column vector V
>> V= [1; 5; 8; 7; 10]
V =
     1
     5
     8
     7
    10
```

2. Create a vector in MATLAB

- To create a vector with elements **sequentially arranged**, we use

```
V = [first element : last element]
```

- If the variables of a vector `V` are **arranged with consecutive values but with a step** that differs from 1, we can generate it using the following expression

```
V = [first element : step : last element]
```

2. Create a vector in MA

- To create a vector with elements **sequentially arranged**, we use

```
V = [first element : last element]
```

- If the variables of a vector V are **arranged with consecutive values but with a step** that differs from 1, we can generate it using the following expression

```
V = [first element : step : last element]
```

```
>> V = [1:5]      % or V = 1:5

V =

     1     2     3     4     5

>> V = [1:2:10] % or V = 1:2:10

V =

     1     3     5     7     9
```

2. Create a vector in MATLAB

- To generate a sequence of values **evenly spaced between two defined boundaries**

  `linspace (S,E,n)` Where:

  `S`: The starting value;

  `E`: The ending value;

  `n`: The number of elements in the vector.

- We can also create a vector based on existing vectors or variables

```
V = [V1 V2]
```

2. Create a vector in M

- To generate a sequence of values **evenly spaced between two defined boundaries**

  `linspace (S,E,n)` Where:

  S: The starting value;

  E: The ending value;

  n: The number of elements in the vector.

- We can also create a vector based on existing vectors or variables

  `V = [V1 V2]`

```
>> V = linspace (2,5,3)

V =

    2.0000    3.5000    5.0000

>> V1 = [1 2 3];
>> V2 = [4 5 6];
>> V12 = [V1 V2]

V12 =

     1     2     3     4     5     6
```

2. Create a vector in MATLAB

In addition to creating vectors, we have the flexibility to perform a range of **operations** during their constructions. This includes the ability to **modify**, **add**, or **remove** elements within the vectors.

`V(x)`: returns the values located at position `'x'` within the vector.

x can be integer, **end** index or **vector** to indicate respectively **position**, **last element** or **set of elements** located at the indicated vector position **x**.

`V(x) = []:` This command removes the elements located in the position **x**

`V(x) = a` : This command replaces the element located in the position **x** with the element **a**

`V(X) = A` : This command replaces the elements located in the vector of positions **X** (Index vector) with the elements in the vector **A**

`V = [V, e]` : This command adds the element **e** to the vector **V**

2. Create a vector in

In addition to creating vectors, we have the fle
their constructions. This includes the ability t
vectors.

V(x): returns the values located at position **'x'**

x can be integer, **end** index or **vector** to indi
elements located at the indicated vector positio

V(x) = []: This command removes the elem

V(x) = a : This command replaces the elemen

V(X) = A : This command replaces the elemen
vector) with the elements in the vector **A**

V = [V, e] : This command adds the element

```
>> V = [1, 5, 8, 7, 10];
>> V (3)
ans =
     8

>> V (end)
ans =
    10

>> V ([1 3 5])
ans =
     1     8    10
```

2. Create a vector in

In addition to creating vectors, we have the fle
their constructions. This includes the ability t
vectors.

V(x): returns the values located at position **'x'**

x can be integer, **end** index or **vector** to indi
elements located at the indicated vector positio

V(x) = []: This command removes the elem

V(x) = a : This command replaces the elemen

V(X) = A : This command replaces the elemen
vector) with the elements in the vector **A**

V = [V, e] : This command adds the element

```
>> V = [1, 5, 8, 7, 10]

V =

     1     5     8     7    10

>> V(3) = []

V =

     1     5     7    10
```

2. Create a vector in

In addition to creating vectors, we have the fle
their constructions. This includes the ability t
vectors.

`V(x)`: returns the values located at position `'x'`

`x` can be integer, **`end`** index or **vector** to indi
elements located at the indicated vector positio

`V(x) = []:` This command removes the elem

`V(x) = a` : This command replaces the elemen

`V(X) = A` : This command replaces the elemen
vector) with the elements in the vector **`A`**

`V = [V, e]` : This command adds the element

```
>> V = [1, 5, 8, 7, 10]

V =

     1     5     8     7    10

>> V(3) = 6

V =

     1     5     6     7    10
```

2. Create a vector in

In addition to creating vectors, we have the fle
their constructions. This includes the ability t
vectors.

V(x): returns the values located at position **'x'**

x can be integer, **end** index or **vector** to indi
elements located at the indicated vector positio

V(x) = []: This command removes the elem

V(x) = a : This command replaces the elemen

V(X) = A : This command replaces the elemen
vector) with the elements in the vector **A**

V = [V, e] : This command adds the element

```
>> V = [1, 5, 8, 7, 10]

V =

     1     5     8     7    10
```

```
>> V ([1 3 5]) = [2 4 6]

V =

     2     5     4     7     6
```

2. Create a vector in

In addition to creating vectors, we have the fle
their constructions. This includes the ability t
vectors.

`V(x)`: returns the values located at position `'x'`

`x` can be integer, **`end`** index or **vector** to indi
elements located at the indicated vector positio

`V(x) = []:` This command removes the elem

`V(x) = a` : This command replaces the elemen

`V(X) = A` : This command replaces the elemen
vector) with the elements in the vector **`A`**

`V = [V, e]` : This command adds the element

```
>> V = [1, 5, 8, 7, 10]

V =

     1     5     8     7    10

>> V = [V, 2]

V =

     1     5     8     7    10     2
```

2. Create a vector in MATLAB

Important functions for vectors

MATLAB offers a wide range of functions for working with vectors. Here are some commonly used functions

`length (V)`: Returns the vector size

`sum (v)` : Calculates the sum of the elements within the vector

`prod (v)` : Calculates the product of the elements within the vector

`max (v)` : Returns the highest value of the vector

`min (v)` : Returns the minimum value of the vector

`mean (v)` : The average value of the elements within the vector

`sort (v)` : Arranges the elements of the array in ascending order

2. Create a vector in

Important functions for vec

MATLAB offers a wide range of functions fo
used functions

length (V): Returns the vector size

sum (v) : Calculates the sum of the elemen

prod (v) : Calculates the product of the ele

max (v) : Returns the highest value of the

min (v) : Returns the minimum value of th

mean (v) : The average value of the elemer

sort (v) : Arranges the elements of the ar

```
>> V = [1, 5, 7, 8, 9];
>> length (V)

ans =

     5

>> sum (V)

ans =

    30

>> prod (V)

ans =

        2520
```

2. Create a vector in

Important functions for vec

MATLAB offers a wide range of functions fo
used functions

`length (V)`: Returns the vector size

`sum (v)` : Calculates the sum of the elemen

`prod (v)` : Calculates the product of the ele

`max (v)` : Returns the highest value of the

`min (v)` : Returns the minimum value of th

`mean (v)` : The average value of the elemer

`sort (v)` : Arranges the elements of the arr

```
>> V = [1, 5, 7, 8, 9];
>> max (V)

ans =

     9

>> min (V)

ans =

     1

>> mean (V)

ans =

     6
```

2. Create a vector in

Important functions for vec

MATLAB offers a wide range of functions fo
used functions

`length (V)`: Returns the vector size

`sum (v)` : Calculates the sum of the elemen

`prod (v)` : Calculates the product of the ele

`max (v)` : Returns the highest value of the

`min (v)` : Returns the minimum value of th

`mean (v)` : The average value of the elemer

`sort (v)` : Arranges the elements of the ar

```
>> V = [3 2 8 0 5];
>> sort (V)

ans =

     0     2     3     5     8
```

2. Create a vector in MATLAB

Algebraic operations

MATLAB is a powerful programming environment for numerical computing that offers robust support for vector operations.

2. Create a vector in

Algebraic operations

MATLAB is a powerful programming environment for numerical computing that offers robust support for vector operations.

```
>> V = [1, 5, 7, 8, 9];
>> U = [6, 8, 4, 9, 7];

% addition
>> V + U

ans =

     7    13    11    17    16

% subtraction
>> V - U

ans =

    -5    -3     3    -1     2
```

2. Create a vector in

Algebraic operations

MATLAB is a powerful programming environment for numerical computing that offers robust support for vector operations.

```
>> V = [1, 5, 7, 8, 9];
>> U = [6, 8, 4, 9, 7];

% Multiplication element by element
>> V .* U

ans =

     6    40    28    72    63

% division element by element
>> V ./ U

ans =

    0.1667    0.6250    1.7500    0.8889    1.2857
```

2. Create a vector in

Algebraic operations

MATLAB is a powerful programming environment for numerical computing that offers robust support for vector operations.

```
% Transpose of a vector
>> V = [1, 5, 7, 8, 9]

V =

     1     5     7     8     9

>> V'

ans =

     1
     5
     7
     8
     9
```

Practice