Computational Methods in Physics (PHY 365)

FA23

Dr. Muhammad Kamran

Department of Physics

COMSATS University Islamabad

Lab 1

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.
- Applications

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.
- Applications
  - ◇ Root finding.

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.
- Applications
  - ◇ Root finding.
  - ◇ Integration.

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.
- Applications
  - ◇ Root finding.
  - ◇ Integration.
  - ◇ Ordinary differential equations.

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.
- Applications
  - ◇ Root finding.
  - ◇ Integration.
  - ◇ Ordinary differential equations.
  - ◇ System of linear equations.

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.
- Applications
  - ◇ Root finding.
  - ◇ Integration.
  - ◇ Ordinary differential equations.
  - ◇ System of linear equations.
  - ◇ Partial differential equations.

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.
- Applications
  - ◇ Root finding.
  - ◇ Integration.
  - ◇ Ordinary differential equations.
  - ◇ System of linear equations.
  - ◇ Partial differential equations.
  - ◇ Matrix eigenvalue problem.

Introduction to Computational Physics

- The field of Computational Physics combines physics, computer science and mathematics to provide scientific solutions to realistic and often complex problems.
- A computational physicist understands
  - → the workings of computers and the relevant science and mathematics, and
  - → how computer algorithms and simulations connect the two.
- Applications
  - ◇ Root finding.
  - ◇ Integration.
  - ◇ Ordinary differential equations.
  - ◇ System of linear equations.
  - ◇ Partial differential equations.
  - ◇ Matrix eigenvalue problem.
  - ◇ Other applications.

Introduction to Computational Physics

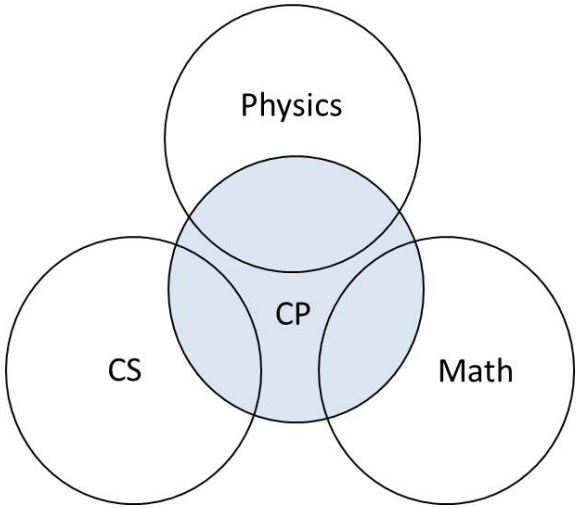
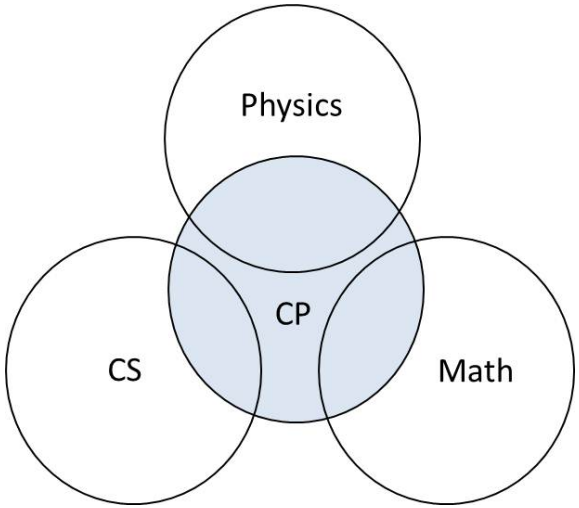


Figure: A representation of the multidisciplinary nature of Computational Physics

Introduction to Computational Physics



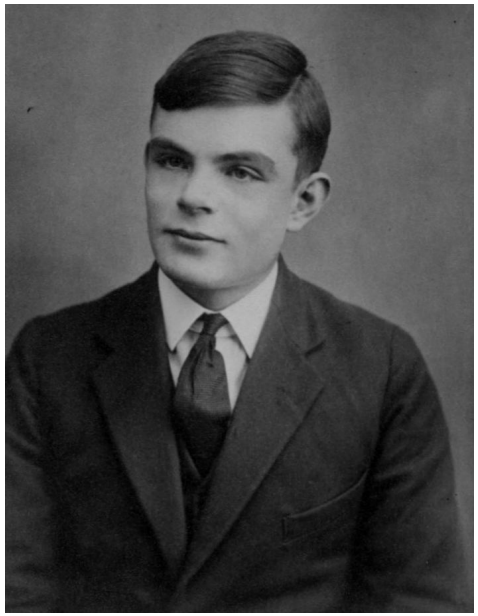


Figure: A representation of the multidisciplinary nature of Computational Physics

Introduction to Computational Physics

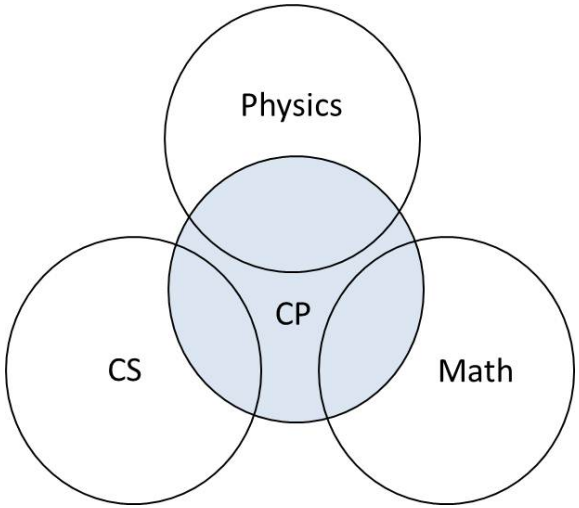


Figure: A representation of the multidisciplinary nature of Computational Physics

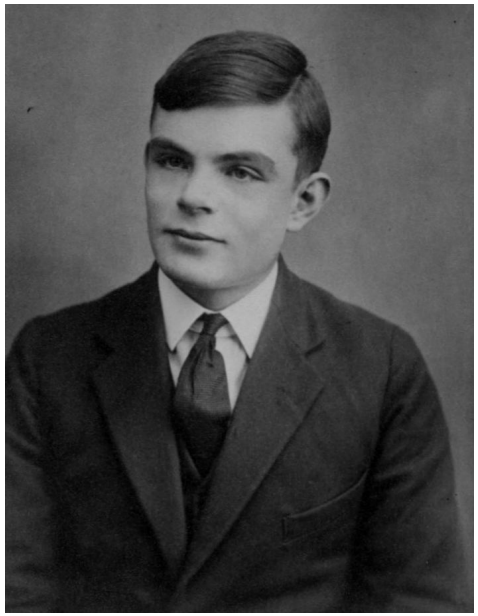


Figure: Alan Turing

Introduction to Computational Physics

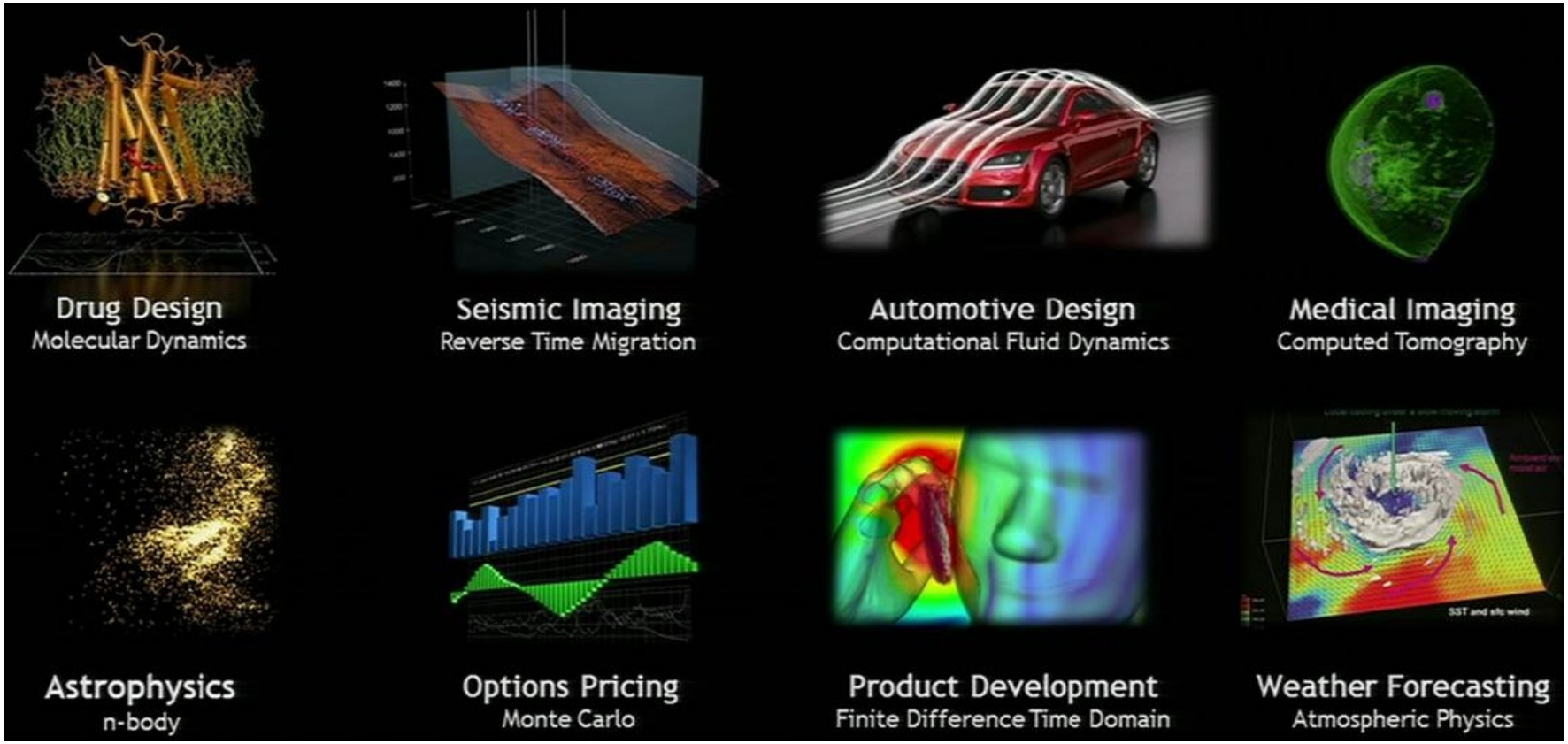


Figure: Some applications of CP

Introduction to MATLAB

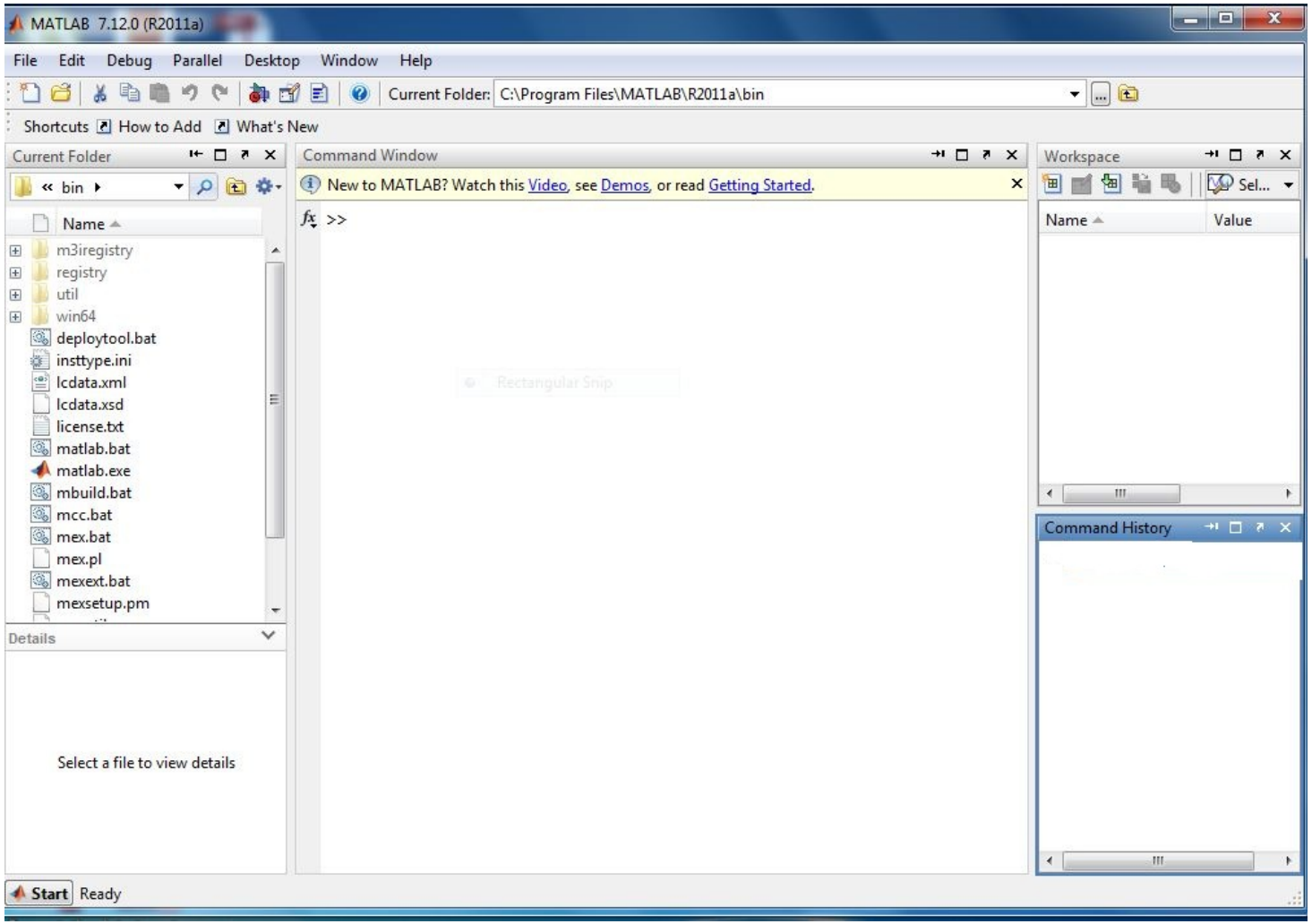


Figure: MATLAB

Key features

- High-level language for scientific and engineering computing.

Key features

- High-level language for scientific and engineering computing.
- Graphics for visualizing data and tools for creating custom plots.

Key features

- High-level language for scientific and engineering computing.
- Graphics for visualizing data and tools for creating custom plots.
- Apps for curve fitting, data classification, signal analysis, and many other domain-specific tasks.

Key features

- High-level language for scientific and engineering computing.
- Graphics for visualizing data and tools for creating custom plots.
- Apps for curve fitting, data classification, signal analysis, and many other domain-specific tasks.
- Add-on toolboxes for a wide range of engineering and scientific applications.

Key features

- High-level language for scientific and engineering computing.
- Graphics for visualizing data and tools for creating custom plots.
- Apps for curve fitting, data classification, signal analysis, and many other domain-specific tasks.
- Add-on toolboxes for a wide range of engineering and scientific applications.
- Tools for building applications with custom user interfaces.

Some built-in functions

- zeros (r, c).
- ones (r, c).
- eye (r, c).
- magic (m).
- rand (r, c).
- diag (x).
- linspace (x_i, x_f, p).

Matrix operations

- ^ exponentiation.

Matrix operations

- ^ exponentiation.
- + addition.

Matrix operations

- ^ exponentiation.
- + addition.
- - subtraction.

Matrix operations

- `^` exponentiation.
- `+` addition.
- `-` subtraction.
- `*` (matrix) multiplication.

Matrix operations

- `^` exponentiation.
- `+` addition.
- `-` subtraction.
- `*` (matrix) multiplication.
- `.` `*` (element-wise) multiplication.

Matrix operations

- `^` exponentiation.
- `+` addition.
- `-` subtraction.
- `*` (matrix) multiplication.
- `.` `*` (element-wise) multiplication.
- `.` `/` (element-wise) division.

Matrix operations

- `^` exponentiation.
- `+` addition.
- `-` subtraction.
- `*` (matrix) multiplication.
- `.` `*` (element-wise) multiplication.
- `.` `/` (element-wise) division.
- `\` left division.

Matrix operations

- `^` exponentiation.
- `+` addition.
- `-` subtraction.
- `*` (matrix) multiplication.
- `.` `*` (element-wise) multiplication.
- `.` `/` (element-wise) division.
- `\` left division.

→ A \B = inv (A) ∗ B.

Relational Operators

- `<` less than.

Relational Operators

- $<$ less than.
- $>$ greater than.

Relational Operators

- $<$ less than.
- $>$ greater than.
- $<$ $=$ less than or equal to.

Relational Operators

- $<$ less than.
- $>$ greater than.
- $<=$ less than or equal to.
- $>=$ greater than or equal to.

Relational Operators

- `<` less than.
- `>` greater than.
- `<` `=` less than or equal to.
- `>` `=` greater than or equal to.
- `=` `=` equal to.

Relational Operators

- `<` less than.
- `>` greater than.
- `<` `=` less than or equal to.
- `>` `=` greater than or equal to.
- `=` `=` equal to.
- `~` `=` not equal to.

Logical Operators

- & and.

Logical Operators

- `&` and.
- `~` not.

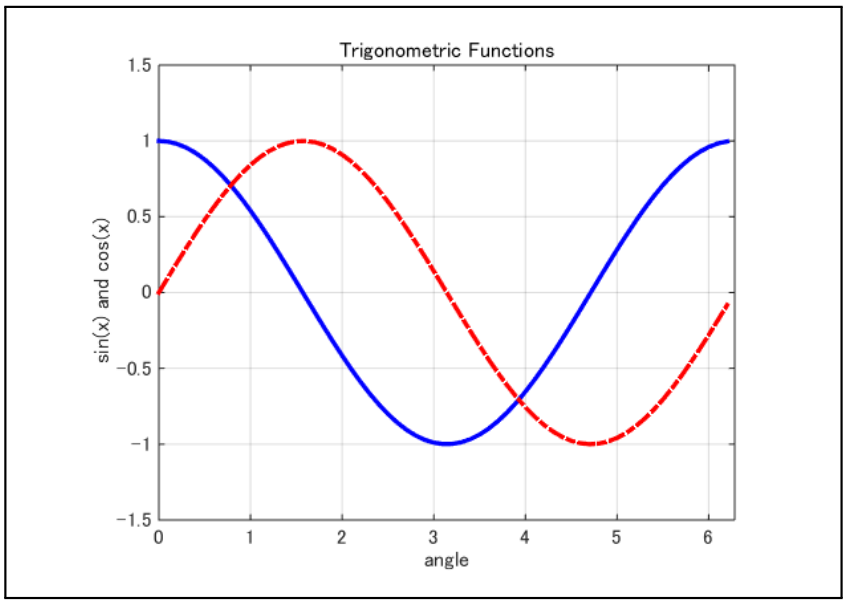
Logical Operators

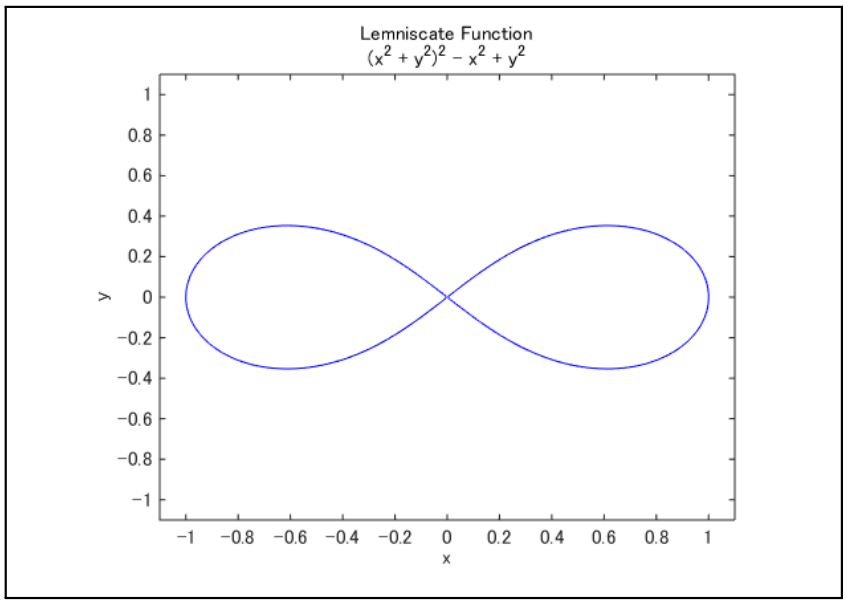
- `&` and.
- `~` not.
- `|` or.

Logical Operators

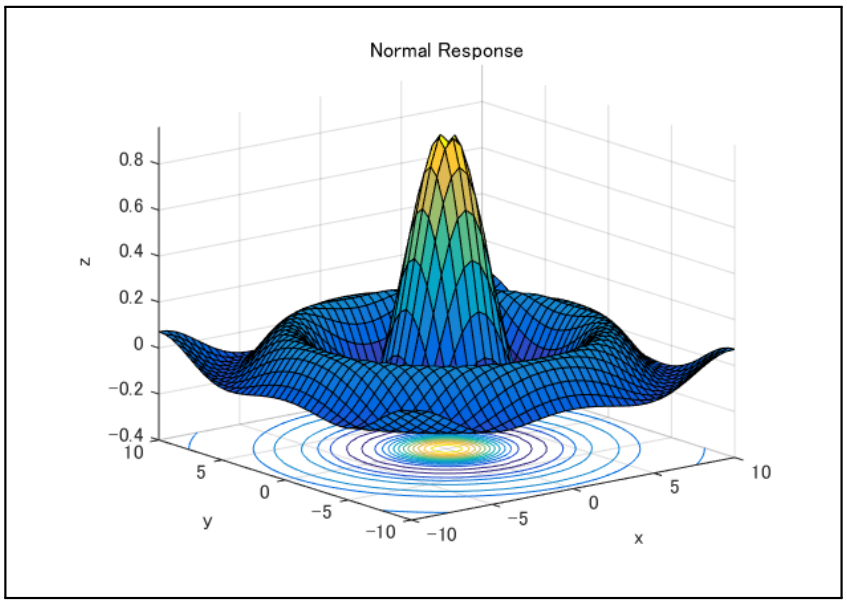
- `&` and.
- `~` not.
- `|` or.
- xor exclusive-OR.

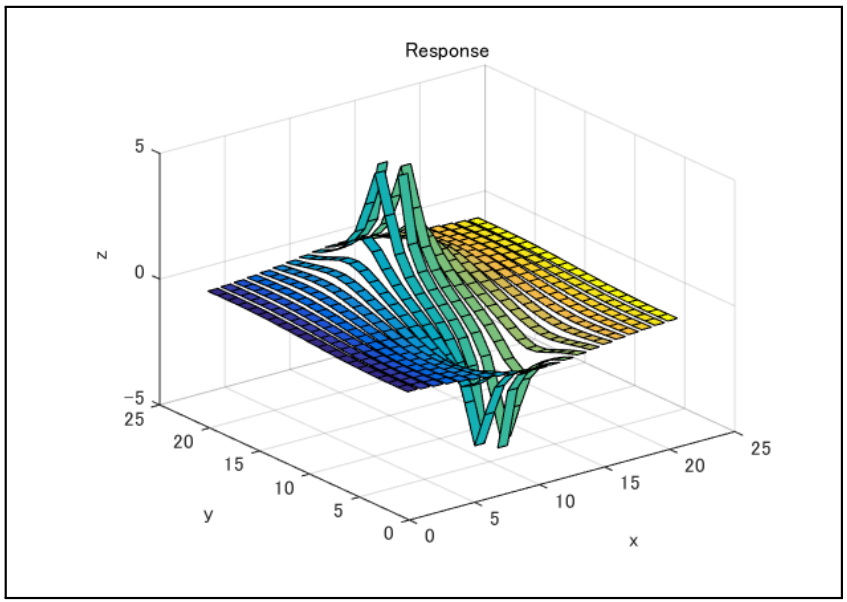
Plots


Trigonometric Functions
sin(x) and cos(x)
angle
1.5
1
0.5
0
−0.5
−1
−1.5
0
1
2
3
4
5
6

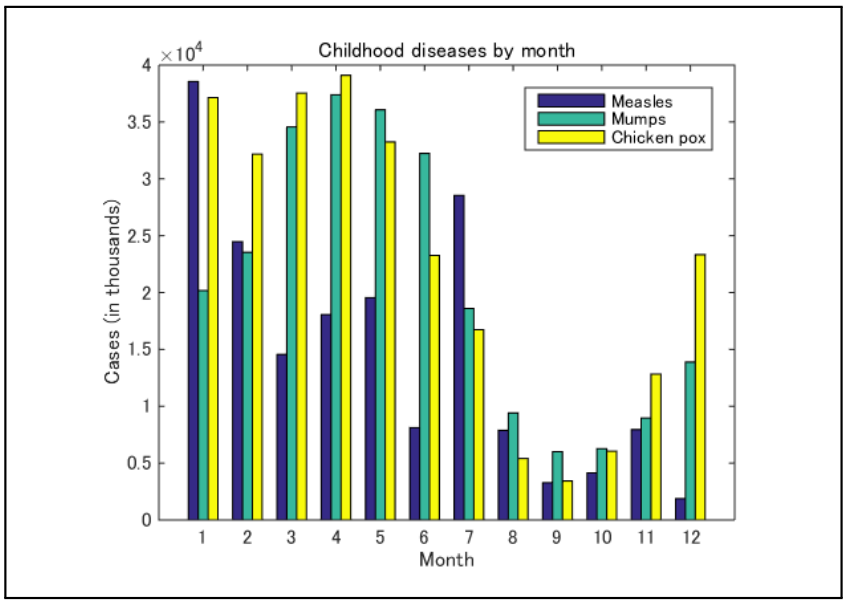

Lemniscate Function
$(x^2 + y^2)^2 - x^2 + y^2$
y
x
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
−1
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
1

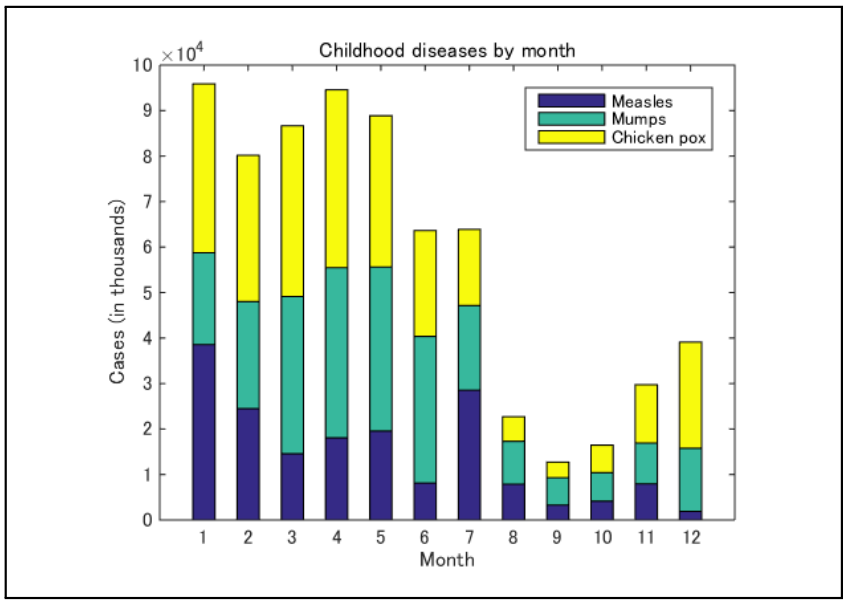
Plots


Normal Response
z
0.8
0.6
0.4
0.2
0
−0.2
−0.4
10
5
0
−5
−10
y
−10
−5
0
5
10
x

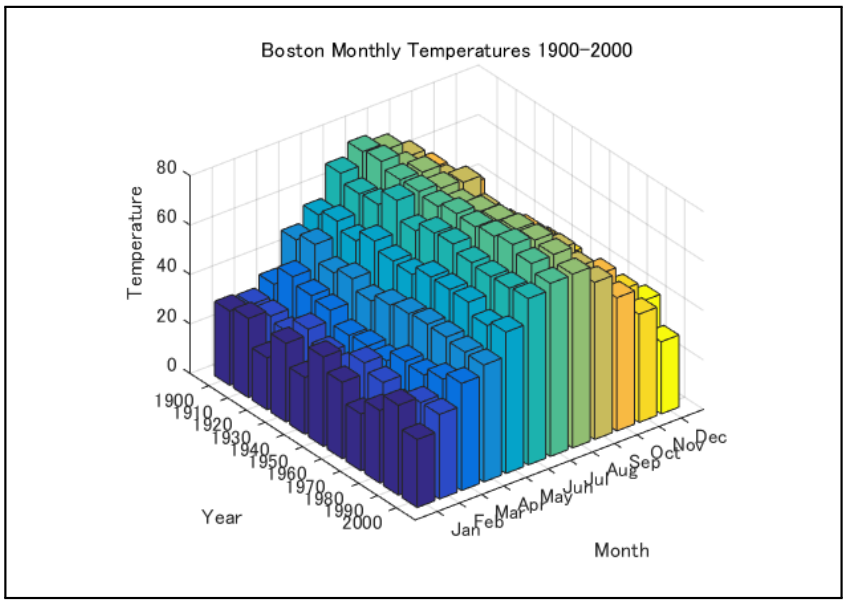

Response
z
5
0
−5
25
20
15
10
5
0
y
0
5
10
15
20
25
x

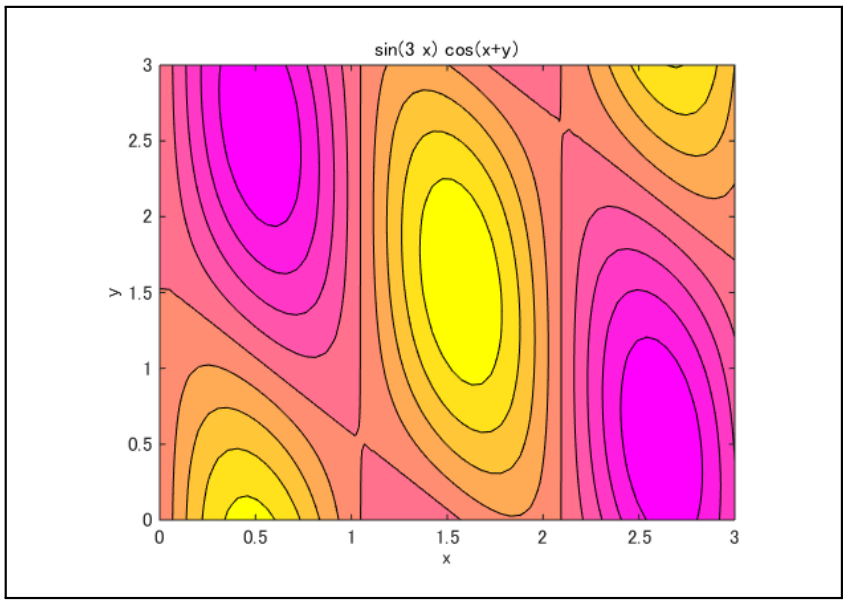
Plots


Childhood diseases by month
×10^4
Cases (in thousands)
Month
Measles
Mumps
Chicken pox
0
0.5
1
1.5
2
2.5
3
3.5
4
1 2 3 4 5 6 7 8 9 10 11 12

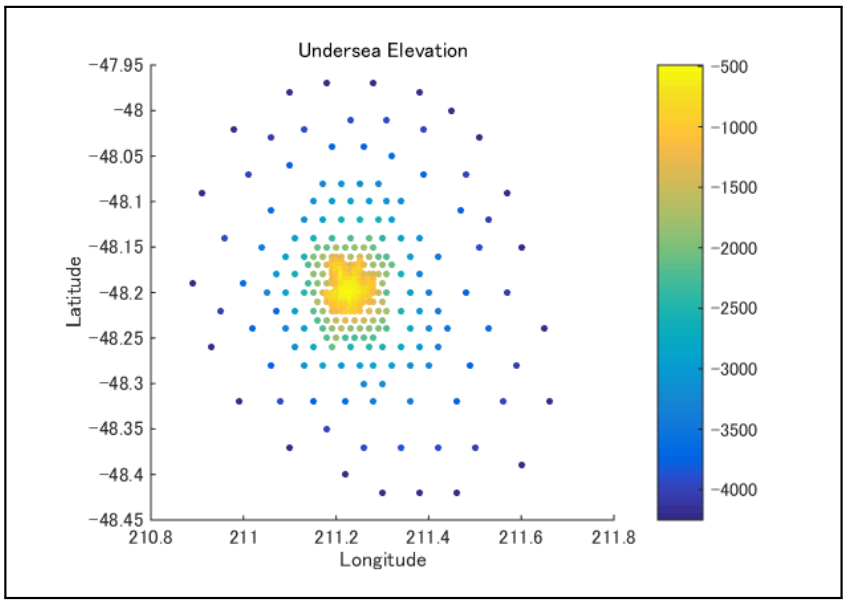

Childhood diseases by month
×10^4
Cases (in thousands)
Month
Measles
Mumps
Chicken pox
0
1
2
3
4
5
6
7
8
9
10
1 2 3 4 5 6 7 8 9 10 11 12

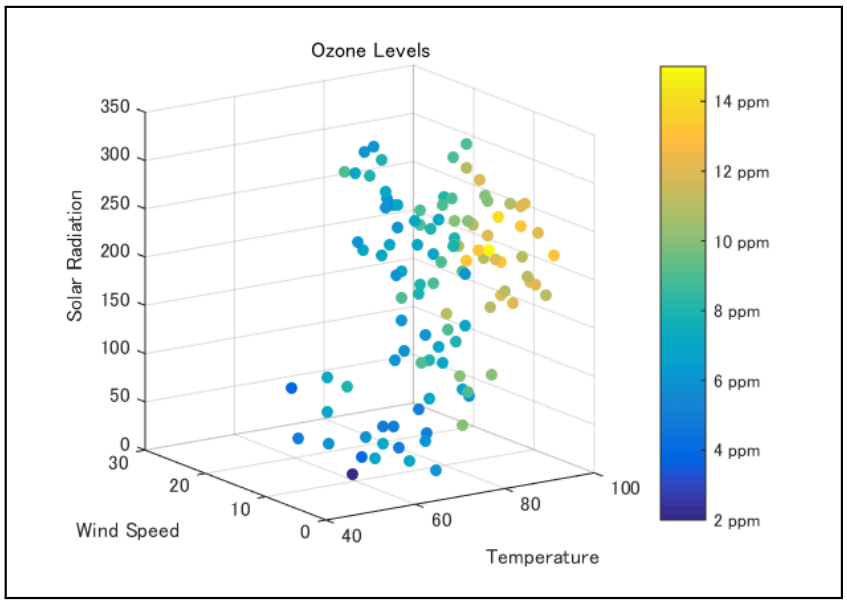
Plots


Boston Monthly Temperatures 1900–2000
Temperature
80
60
40
20
0
1900
1910
1920
1930
1940
1950
1960
1970
1980
1990
2000
Year
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month

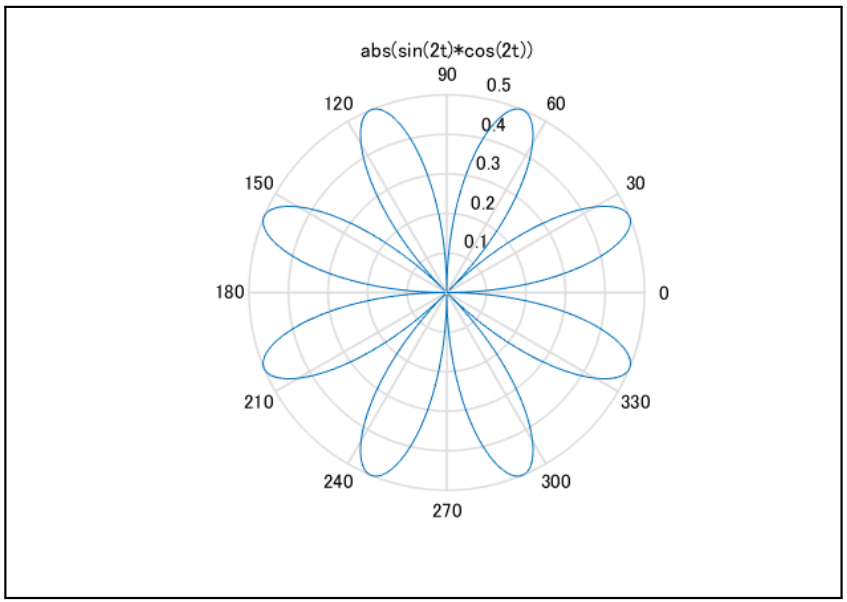

sin(3 x) cos(x+y)
y
3
2.5
2
1.5
1
0.5
0
0
0.5
1
1.5
2
2.5
3
x

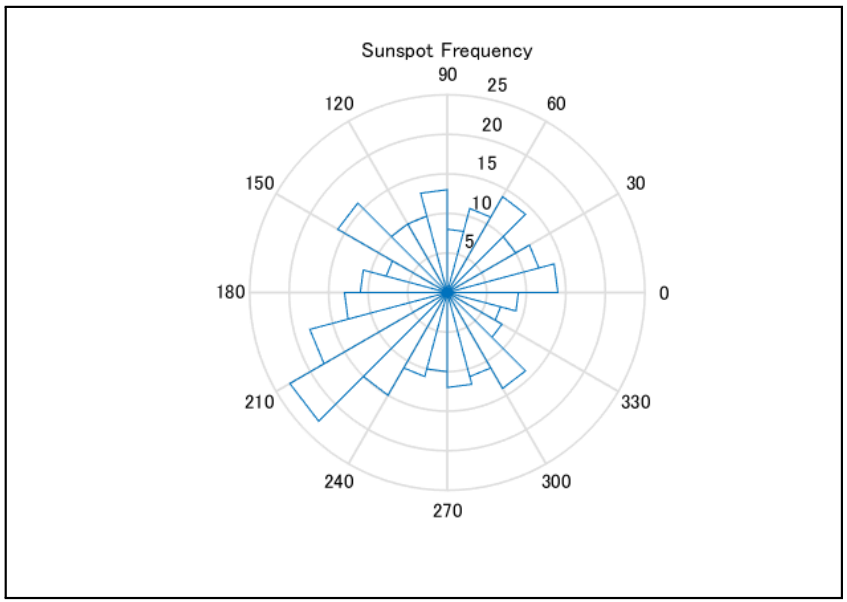
Plots


Undersea Elevation
Latitude
Longitude
−47.95
−48
−48.05
−48.1
−48.15
−48.2
−48.25
−48.3
−48.35
−48.4
−48.45
210.8
211
211.2
211.4
211.6
211.8
−500
−1000
−1500
−2000
−2500
−3000
−3500
−4000

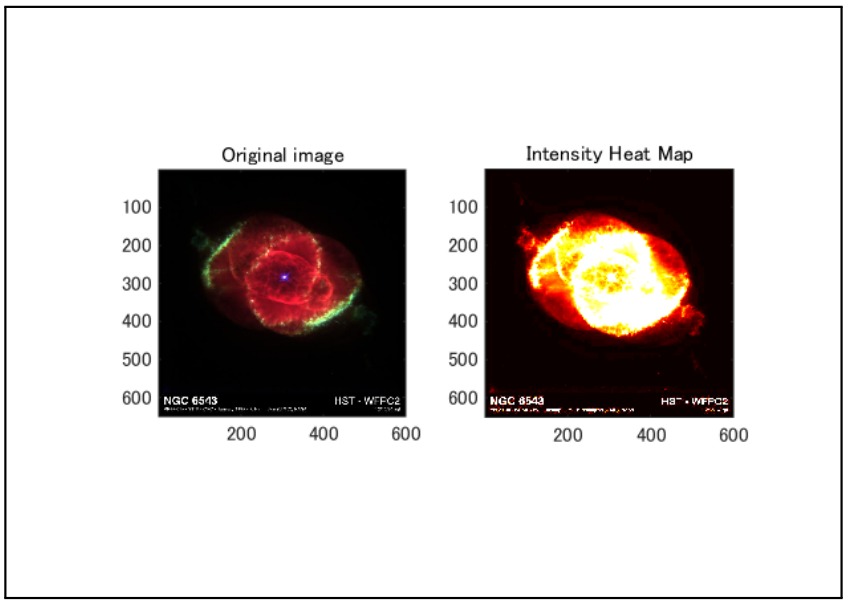

Ozone Levels
Solar Radiation
Wind Speed
Temperature
350
300
250
200
150
100
50
0
30
20
10
0
40
60
80
100
14 ppm
12 ppm
10 ppm
8 ppm
6 ppm
4 ppm
2 ppm

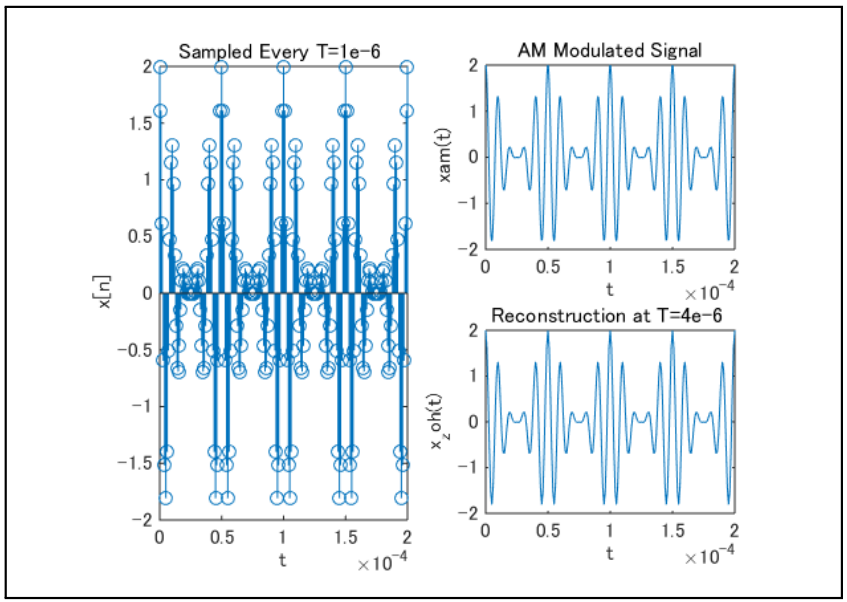
Plots


abs(sin(2t)*cos(2t))
90
0.5
120
60
0.4
0.3
150
30
0.2
0.1
180
0
210
330
240
300
270

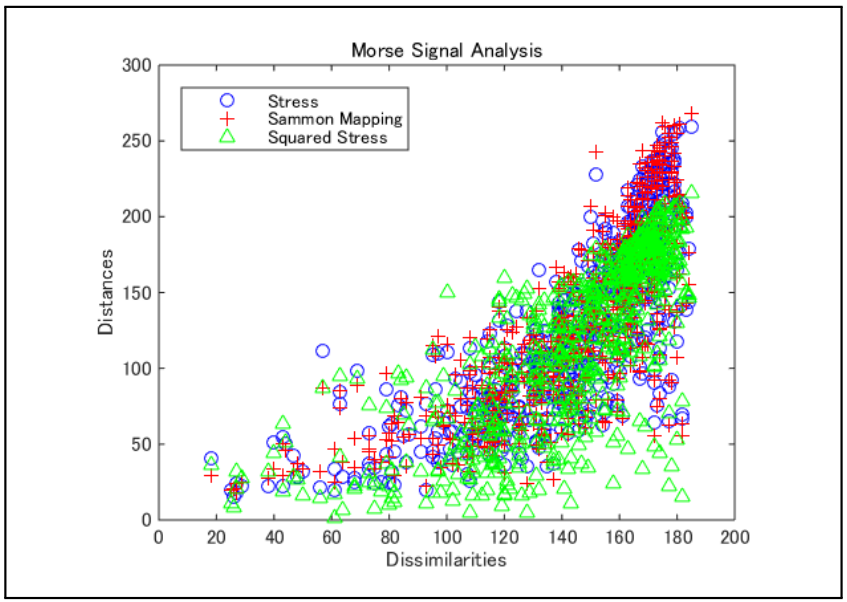

Sunspot Frequency
90
25
120
60
20
15
150
30
10
5
180
0
210
330
240
300
270

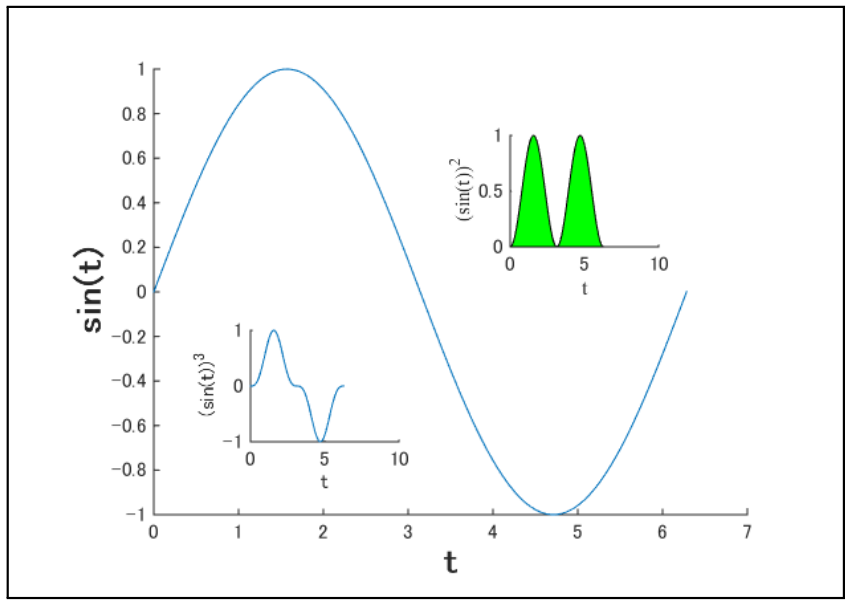
Plots


Original image
Intensity Heat Map
100
200
300
400
500
600
200
400
600
NGC 6543
HST - WFPC2

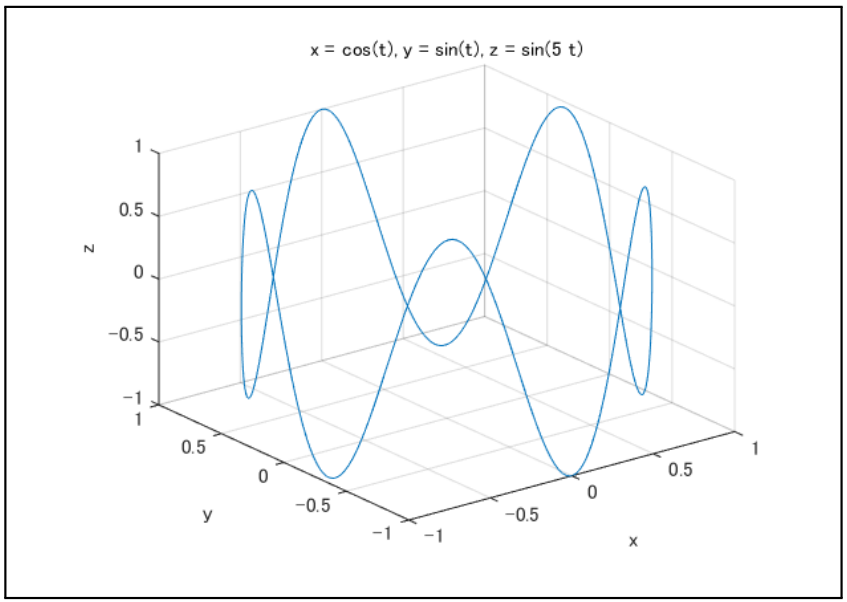

Sampled Every T=1e-6
x[n]
t
×10^-4
AM Modulated Signal
xam(t)
Reconstruction at T=4e-6
x_oh(t)

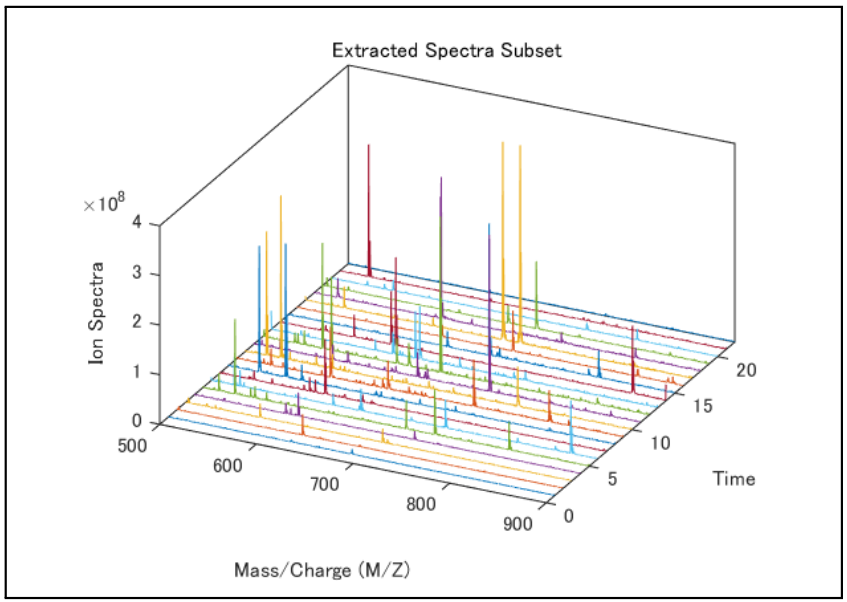
Plots


Morse Signal Analysis
Stress
Sammon Mapping
Squared Stress
Distances
Dissimilarities
0
50
100
150
200
250
300
0
20
40
60
80
100
120
140
160
180
200

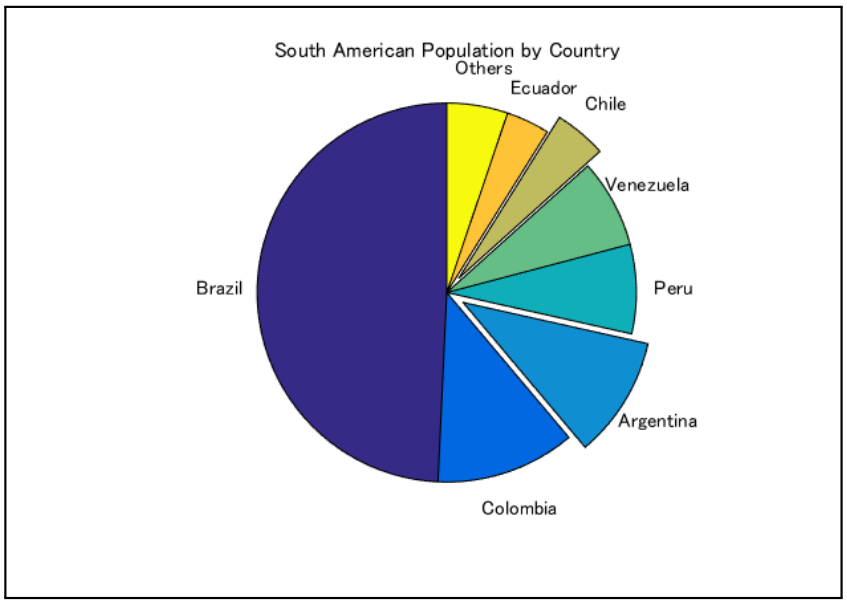

sin(t)
t
(sin(t))^2
(sin(t))^3
−1
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
1
0
1
2
3
4
5
6
7
0.5
10

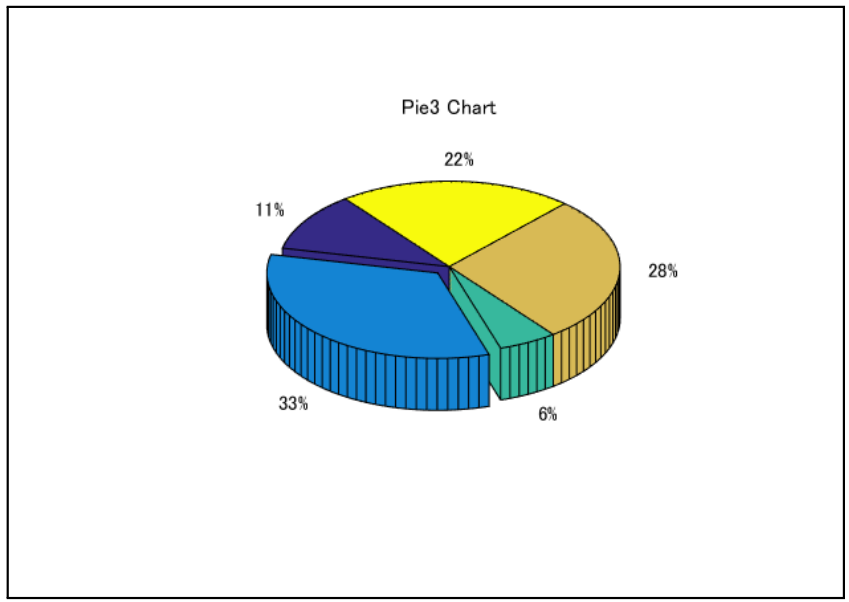
Plots


x = cos(t), y = sin(t), z = sin(5 t)
z
y
x

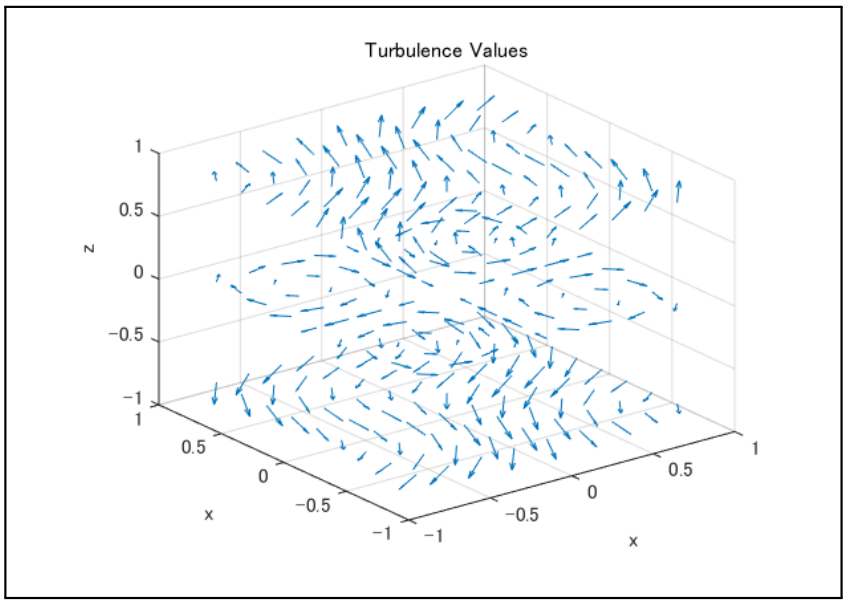

Extracted Spectra Subset
Ion Spectra
Mass/Charge (M/Z)
Time

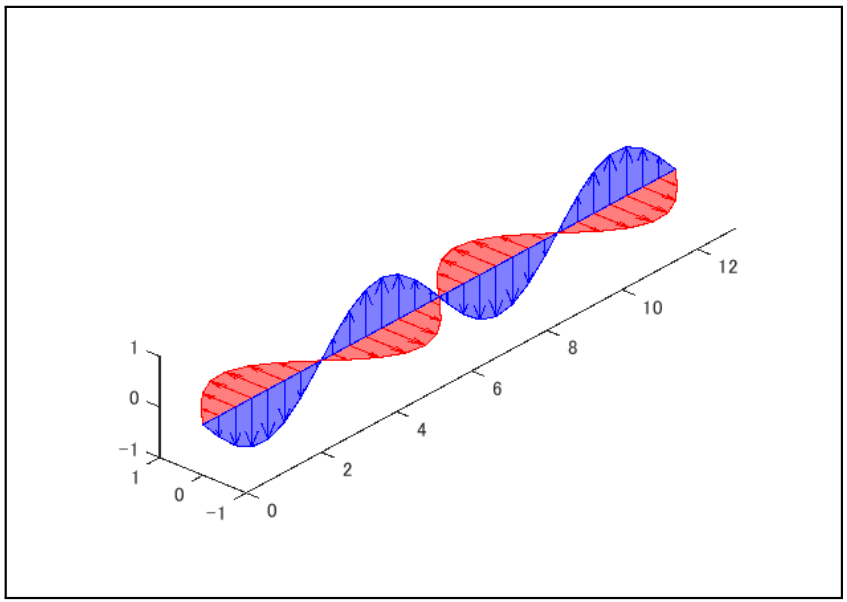
Plots


South American Population by Country
Others
Ecuador
Chile
Venezuela
Peru
Argentina
Colombia
Brazil

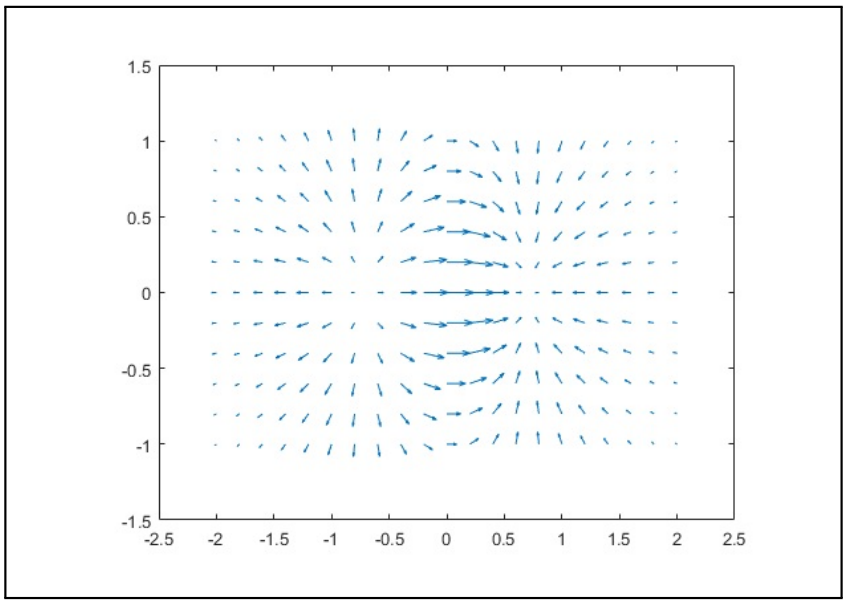

Pie3 Chart
22%
28%
6%
33%
11%

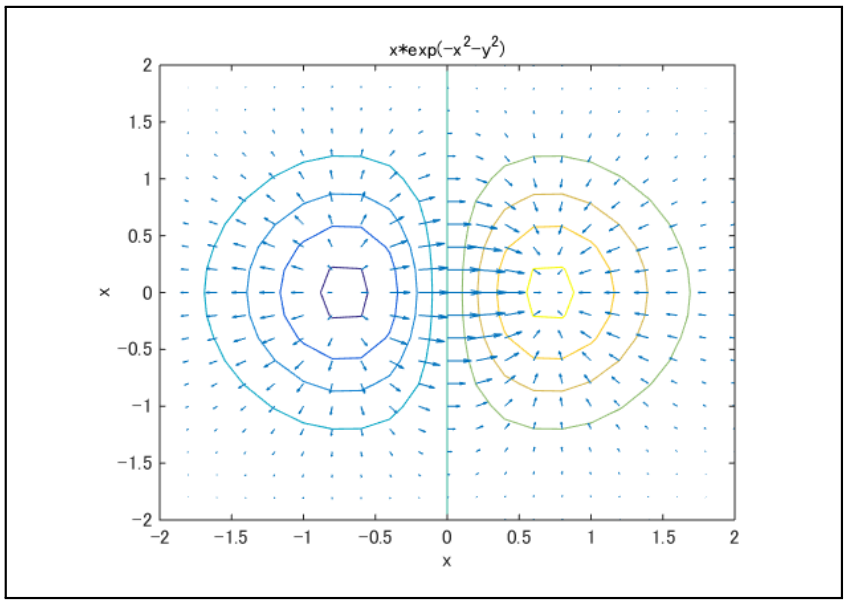
Plots



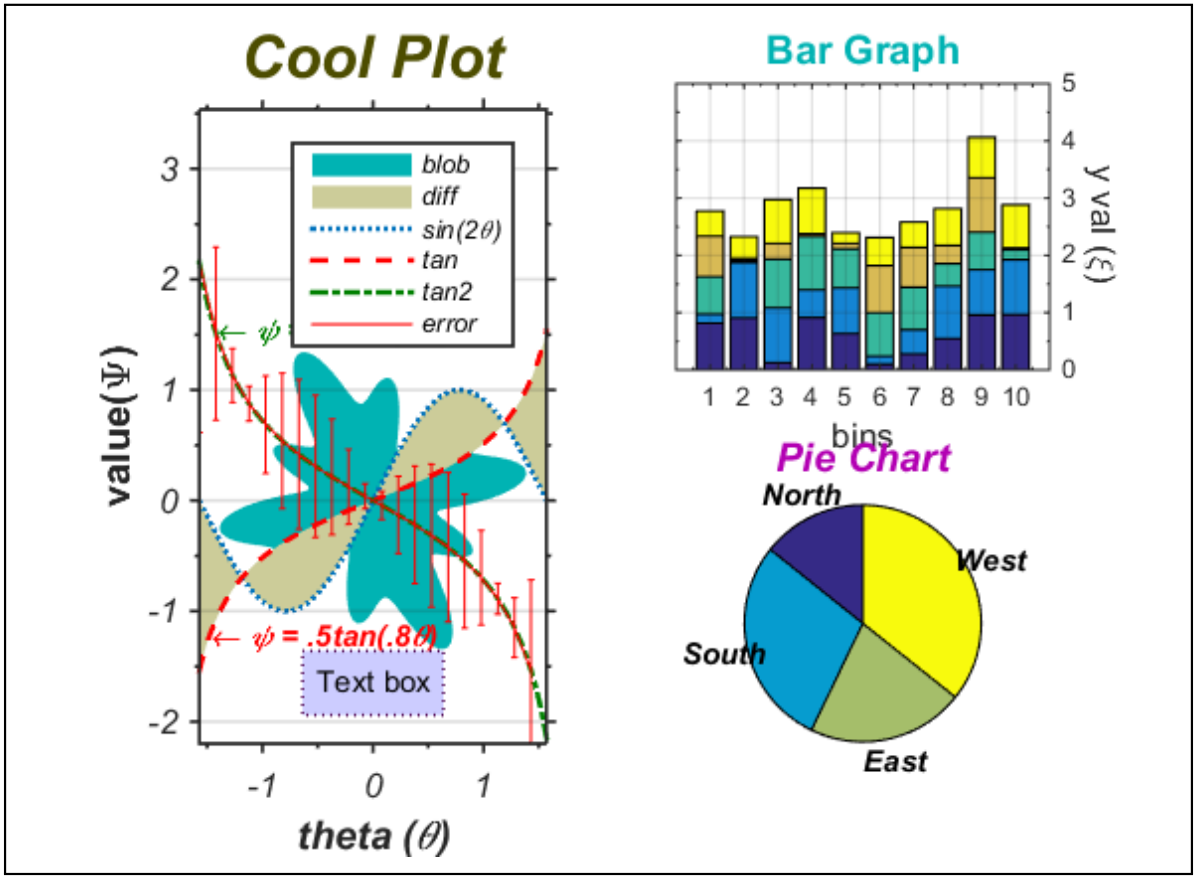


Plots

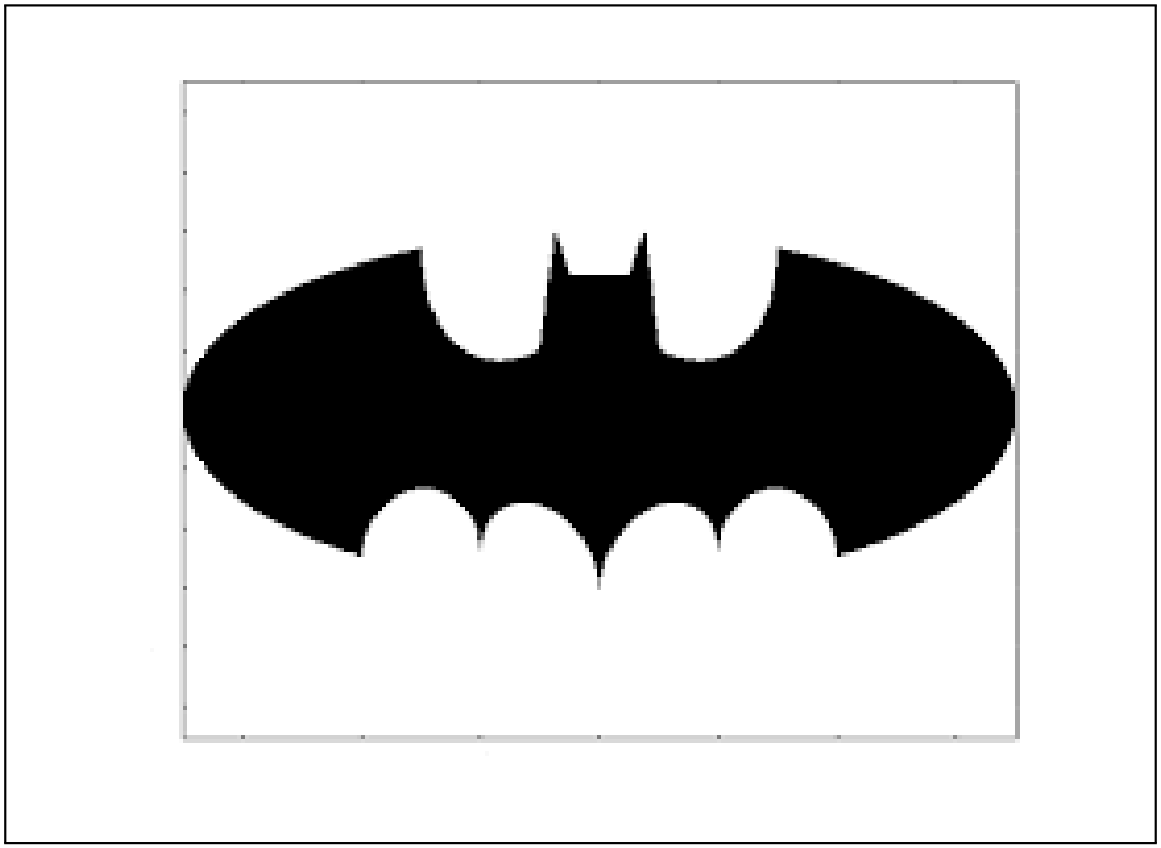

1.5
1
0.5
0
-0.5
-1
-1.5
-2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5


x*exp(−x^2−y^2)
2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
x
−2 −1.5 −1 −0.5 0 0.5 1 1.5 2
x

Plots


Cool Plot
blob
diff
sin(2θ)
tan
tan2
error
ψ = .5tan(.8θ)
Text box
value(Ψ)
theta (θ)
Bar Graph
y val (ε)
bins
Pie Chart
North
West
South
East

Plots



Help

- Type "help topic" in the command prompt.

Help

- Type "help topic" in the command prompt.
  - help sin

Help

- Type "help topic" in the command prompt.
  - help sin

```
>> help sin
 SIN    Sine of argument in radians.
    SIN(X) is the sine of the elements of X.

    See also asin, sind.

    Overloaded methods:
       codistributed/sin

    Reference page in Help browser
       doc sin
```

Help

- Type "help topic" in the command prompt.
  - help sin

```
>> help sin
 SIN    Sine of argument in radians.
    SIN(X) is the sine of the elements of X.

    See also asin, sind.

    Overloaded methods:
       codistributed/sin

    Reference page in Help browser
       doc sin
```

- Type "doc topic" in the command prompt.

M-files

- These are text files containing MATLAB programs.
  - Have a ".m" extension.

M-files

- These are text files containing MATLAB programs.
  - Have a ".m" extension.
- M-files can be called form the command line or from other M-files.

M-files

- These are text files containing MATLAB programs.
  - Have a ".m" extension.
- M-files can be called form the command line or from other M-files.
- Two kinds
  - Script file.
  - Function file.

M-files

- These are text files containing MATLAB programs.
  - Have a ".m" extension.
- M-files can be called form the command line or from other M-files.
- Two kinds
  - Script file.
  - Function file.
- A script file contains a sequence of MATLAB statements.
  - All variables are added to the workspace.

M-files

- A function file is a program (or routine) that accepts input arguments and returns output arguments.
  - Same name as the defined function.
  - Has its own workspace, which is also called the local workspace.
  - Can accept more than one input arguments and may return more than one output arguments.
  - The first line starts with the keyword function.

M-files

- A function file is a program (or routine) that accepts input arguments and returns output arguments.
  - Same name as the defined function.
  - Has its own workspace, which is also called the local workspace.
  - Can accept more than one input arguments and may return more than one output arguments.
  - The first line starts with the keyword function.
- Syntax.
  - function [y1, ..., yN] = myfun (x1, ..., xM).

Anonymous Functions

- A function that is not stored in a program file, but is associated with a variable whose data type is function_handle.

Anonymous Functions

- A function that is not stored in a program file, but is associated with a variable whose data type is function_handle.
- Anonymous functions can accept inputs and return outputs, just as standard functions do.
  - They can contain only a single executable statement.

Anonymous Functions

- A function that is not stored in a program file, but is associated with a variable whose data type is function_handle.
- Anonymous functions can accept inputs and return outputs, just as standard functions do.
  - They can contain only a single executable statement.
- Many MATLAB functions accept function handles as inputs.
- Anonymous functions are created by @ operator.

Symbolic/Algebraic computation

- Symbolic Math Toolbox provides functions for solving, plotting, and manipulating symbolic math equations.

Symbolic/Algebraic computation

- Symbolic Math Toolbox provides functions for solving, plotting, and manipulating symbolic math equations.
- The toolbox provides functions in common mathematical areas such as
  - Calculus
  - Linear Algebra
  - Algebraic and ordinary differential equations
  - Plotting of analytical functions in 2D and 3D

Parallel computing

- Parallel Computing Toolbox lets you solve computationally and data-intensive problems using multicore processors, GPUs, and computer clusters.

Parallel computing

- Parallel Computing Toolbox lets you solve computationally and data-intensive problems using multicore processors, GPUs, and computer clusters.
- The toolbox lets you use the full processing power of multicore machines by executing applications on MATLAB computational engines that run locally.

Parallel computing

- Parallel Computing Toolbox lets you solve computationally and data-intensive problems using multicore processors, GPUs, and computer clusters.
- The toolbox lets you use the full processing power of multicore machines by executing applications on MATLAB computational engines that run locally.
- Applications can be parallelized without Compute Unified Device Architecture (CUDA) or Message Passing Interface (MPI) programming.

Data import/export

- Data import and export functions provide access to data from files, other applications, web services, and external devices.

Data import/export

- Data import and export functions provide access to data from files, other applications, web services, and external devices.
- You can read popular file formats
  - Excel spreadsheets
  - Text
  - Images
  - Audio and video
  - Scientific data formats

Exercise: Some useful built-in functions and commands

- A = magic (4)

Exercise: Some useful built-in functions and commands

- A = magic (4)
- B = ones (4)

Exercise: Some useful built-in functions and commands

- A = magic (4)
- B = ones (4)
- C = eye (4)

Exercise: Some useful built-in functions and commands

- A = magic (4)
- B = ones (4)
- C = eye (4)
- D = rand (4)

Exercise: Some useful built-in functions and commands

- A = magic (4)
- B = ones (4)
- C = eye (4)
- D = rand (4)
- E = linspace $(0, 1, 100)$

Exercise: Some useful built-in functions and commands

- A = magic (4)
- B = ones (4)
- C = eye (4)
- D = rand (4)
- E = linspace $(0, 1, 100)$
- F = find (D < 0.5)

Exercise: Some useful built-in functions and commands

- who

Exercise: Some useful built-in functions and commands

- who
- whos

Exercise: Some useful built-in functions and commands

- who
- whos
- figure

Exercise: Some useful built-in functions and commands

- who
- whos
- figure
- clear variables

Exercise: Some useful built-in functions and commands

- who
- whos
- figure
- clear variables
- close all

Exercise: Some useful built-in functions and commands

- who
- whos
- figure
- clear variables
- close all
- clc

Exercise: Some useful built-in functions and commands

- who
- whos
- figure
- clear variables
- close all
- clc
- A = magic (4)

Exercise: Some useful built-in functions and commands

- who
- whos
- figure
- clear variables
- close all
- clc
- A = magic (4) ;

Exercise: Matrix operations

- B = A;

Exercise: Matrix operations

- B = A;
- A ^ 2

Exercise: Matrix operations

- B = A;
- A ^ 2
- A .^ 2

Exercise: Matrix operations

- B = A;
- A ^ 2
- A .^ 2
- A / B

Exercise: Matrix operations

- B = A;
- A ^ 2
- A .^ 2
- A / B
- A ./ B

Exercise: Matrix operations

- B = A;
- A ^ 2
- A .^ 2
- A / B
- A ./ B
- A \ B

Exercise: Matrix operations

- B = A;
- A ^ 2
- A .^ 2
- A / B
- A ./ B
- A \ B
- inv (A) * B

Exercise: Matrix operations

- A1 = magic(4);

Exercise: Matrix operations

- A1 = magic(4);
- B1 = magic(6);

Exercise: Matrix operations

- A1 = magic(4);
- B1 = magic(6);
- C1 = rand(4,6);

Exercise: Matrix operations

- A1 = magic(4);
- B1 = magic(6);
- C1 = rand(4,6);
- D1 = [A1, B1];

Exercise: Matrix operations

- A1 = magic(4);
- B1 = magic(6);
- C1 = rand(4,6);
- D1 = [A1, B1];
- E1 = [A1; B1];

Exercise: Matrix operations

- F1 = [A1; C1];

Exercise: Matrix operations

- F1 = [A1; C1];
- G1 = [A1, C1];

Exercise: Matrix operations

- F1 = [A1; C1];
- G1 = [A1, C1];
- A1 (2) = 0;

Exercise: Matrix operations

- F1 = [A1; C1];
- G1 = [A1, C1];
- A1 (2) = 0;
- H1 = B1 (: , 3)

Exercise: Matrix operations

- F1 = [A1; C1];
- G1 = [A1, C1];
- A1 (2) = 0;
- H1 = B1 (: , 3)
- I1 = A1 (3 : 6, 3)

Exercise: Matrix operations

- a = [1 : 3]

  b = [5 : 7]

Exercise: Matrix operations

- a = [1 : 3]

  b = [5 : 7]

- a1 = dot(a , b)

Exercise: Matrix operations

- a = [1 : 3]

  b = [5 : 7]

- a1 = dot(a , b)

- a2 = dot(a , b')

Exercise: Matrix operations

- a = [1 : 3]

  b = [5 : 7]

- a1 = dot(a , b)
- a2 = dot(a , b')
- a3 = cross(a , b)

Exercise: Matrix operations

- a = [1 : 3]

  b = [5 : 7]

- a1 = dot(a , b)
- a2 = dot(a , b')
- a3 = cross(a , b)
- a4 = kron(a , b)

Exercise: Matrix operations

- a = [1 : 3]

  b = [5 : 7]

- a1 = dot(a , b)
- a2 = dot(a , b')
- a3 = cross(a , b)
- a4 = kron(a , b)
- a5 = length(a)

Exercise: Matrix operations

- a = [1 : 3]

  b = [5 : 7]

- a1 = dot(a , b)
- a2 = dot(a , b')
- a3 = cross(a , b)
- a4 = kron(a , b)
- a5 = length(a)
- a6 = max(a)

Exercise: Matrix operations

- a = [1 : 3]

  b = [5 : 7]

- a1 = dot(a , b)
- a2 = dot(a , b')
- a3 = cross(a , b)
- a4 = kron(a , b)
- a5 = length(a)
- a6 = max(a)
- a7 = min(b)

Exercise: Matrix operations

- A = [1,2,3,4 ; 5,6,7,8 ; 9,10,11,12]

Exercise: Matrix operations

- A = [1,2,3,4 ; 5,6,7,8 ; 9,10,11,12]

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

Exercise: Matrix operations

- A = [1,2,3,4 ; 5,6,7,8 ; 9,10,11,12]

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

- B = A(0)

Exercise: Matrix operations

- A = [1,2,3,4 ; 5,6,7,8 ; 9,10,11,12]

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

- B = A(0)
- C = A(1)

Exercise: Matrix operations

- A = [1,2,3,4 ; 5,6,7,8 ; 9,10,11,12]

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

- B = A(0)
- C = A(1)
- D = A(4)

Exercise: Matrix operations

- A = [1,2,3,4 ; 5,6,7,8 ; 9,10,11,12]

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

- B = A(0)
- C = A(1)
- D = A(4)
- E = A(end)

Exercise: Matrix operations

- F = size(A)

Exercise: Matrix operations

- F = size(A)
- G = length(A)

Exercise: Matrix operations

- F = size(A)
- G = length(A)
- H = A(2 , 3)

Exercise: Matrix operations

- F = size(A)
- G = length(A)
- H = A(2 , 3)
- A(2,2) = 20

Exercise: Matrix operations

- F = size(A)
- G = length(A)
- H = A(2 , 3)
- A(2,2) = 20
- I = A(: , 2)

Exercise: Matrix operations

- F = size(A)
- G = length(A)
- H = A(2 , 3)
- A(2,2) = 20
- I = A(: , 2)
- J = sum(A)

Exercise: Matrix operations

- F = size(A)
- G = length(A)
- H = A(2 , 3)
- A(2,2) = 20
- I = A(: , 2)
- J = sum(A)
- K = max(A)

Exercise: Matrix operations

- Create a 5×5 "magic square". Verify that the sum of the integers in each row, column and diagonal is equal.

Exercise: Matrix operations

- Create a 5×5 "magic square". Verify that the sum of the integers in each row, column and diagonal is equal.

```
A = magic(5);
```

Exercise: Matrix operations

- Create a 5×5 "magic square". Verify that the sum of the integers in each row, column and diagonal is equal.

```
A = magic(5);
A1 = sum(A)
```

Exercise: Matrix operations

- Create a 5×5 "magic square". Verify that the sum of the integers in each row, column and diagonal is equal.

```
A = magic(5);

A1 = sum(A)

A2 = sum(A,2)
```

Exercise: Matrix operations

- Create a 5×5 "magic square". Verify that the sum of the integers in each row, column and diagonal is equal.

```
A = magic(5);

A1 = sum(A)

A2 = sum(A,2)

A3 = trace(A)
```

Exercise: Matrix operations

- Create a 5×5 "magic square". Verify that the sum of the integers in each row, column and diagonal is equal.

```
A = magic(5);

A1 = sum(A)

A2 = sum(A,2)

A3 = trace(A)

A4 = trace(flip(A))
```

Exercise: Matrix operations

- Create a 5×5 "magic square". Verify that the sum of the integers in each row, column and diagonal is equal.

```
A = magic(5);

A1 = sum(A)

A2 = sum(A,2)

A3 = trace(A)

A4 = trace(flip(A))
```

- Extract the 3rd row of matrix A.

Exercise: Matrix operations

- Create a 5×5 "magic square". Verify that the sum of the integers in each row, column and diagonal is equal.

```
A = magic(5);

A1 = sum(A)

A2 = sum(A,2)

A3 = trace(A)

A4 = trace(flip(A))
```

- Extract the 3rd row of matrix A.

```
A5 = A(3 , :)
```

Exercise: Matrix operations

- Take transpose of the matrix A.

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

- Determine the highest integer in the matrix A.

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

- Determine the highest integer in the matrix A.

  C2 = max(max(A))

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

- Determine the highest integer in the matrix A.

  C2 = max(max(A))

- Sort the matrix A column-wise, in ascending order.

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

- Determine the highest integer in the matrix A.

  C2 = max(max(A))

- Sort the matrix A column-wise, in ascending order.

  C3 = sort(A)

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

- Determine the highest integer in the matrix A.

  C2 = max(max(A))

- Sort the matrix A column-wise, in ascending order.

  C3 = sort(A)

- Sort the matrix A row-wise, in descending order.

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

- Determine the highest integer in the matrix A.

  C2 = max(max(A))

- Sort the matrix A column-wise, in ascending order.

  C3 = sort(A)

- Sort the matrix A row-wise, in descending order.

  C4 = sort(A , 2 ,'descend')

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

- Determine the highest integer in the matrix A.

  C2 = max(max(A))

- Sort the matrix A column-wise, in ascending order.

  C3 = sort(A)

- Sort the matrix A row-wise, in descending order.

  C4 = sort(A , 2 ,'descend')

- Take determinant of the matrix A.

Exercise: Matrix operations

- Take transpose of the matrix A.

  C1 = A'

- Determine the highest integer in the matrix A.

  C2 = max(max(A))

- Sort the matrix A column-wise, in ascending order.

  C3 = sort(A)

- Sort the matrix A row-wise, in descending order.

  C4 = sort(A , 2 ,'descend')

- Take determinant of the matrix A.

  C5 = det(A)

References

- https://www.mathworks.com/help/matlab/ref/function.html
- https://www.mathworks.com/help/matlab/data-import-and-export.html
- https://www.mathworks.com/help/matlab/matlab_prog/anonymous-functions.html
- https://www.mathworks.com/products/symbolic.html
- https://www.mathworks.com/products/parallel-computing.html
- https://www.mathworks.com/help/matlab/import_export/supported-file-formats.html
- https://www.mathworks.com/help/matlab/math/basic-matrix-operations.html
- https://www.mathworks.com/help/matlab/ref/find.html
- https://www.mathworks.com/matlabcentral/fileexchange/33124-batman-equation-the-legend

References

- https://www.mathworks.com/help/matlab/matlab_prog/loop-control-statements.html
- https://www.mathworks.com/help/matlab/ref/plot.html
- https://www.mathworks.com/help/matlab/ref/plot3.html
- https://www.mathworks.com/help/matlab/ref/surf.html
- https://www.mathworks.com/help/matlab/ref/contour.html
- https://www.mathworks.com/help/matlab/creating_plots/types-of-matlab-plots.html