SM Lab File

MANAV RACHNA INTERNATIONAL INSTITUTE OF RESEARCH AND STUDIES

FACULTY OF ENGINEERING AND TECHNOLOGY

Practical File

For

7th Semester (Academic Year 2022-2023)

SUBJECT NAME: SM-LAB

SUBJECT CODE: BCS-DS-721

Submitted By:-
Student Name: PARESH GUPTA
Roll No.: 1/19/FET/BCS/081
Branch: B Tech CSE
Section: 7CSB$_1$

Submitted To:-
Faculty Name: Dr. SAVITA
Designation: Professor
Department: Computer Science and Engineering (CSE)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SM Lab File

INDEX

SM Lab File

11.	We have a about vehicle performance, Miles per gallon is represented By matrix m and corresponding weight of car is represented by W matrix. Find Covariance and Correlation between these parameters. Plot the data set.		
12	Write a program to find the structural stability of the given trussbridge.		
13.	Write a program to implement single server queue.		
14.	Write a program for pure pursuit problem using SCI lab.		

SM Lab File

PRACTICAL - 1
Aim -Introduction to Sci Lab

<u>Overview</u>

Scilab is a programming language associated with a rich collection of numerical algorithms covering many aspects of scientific computing problems. From the software point of view, Scilab is an interpreted language. This generally allows to get faster development processes, because the user directly accesses to a high level language, with a rich set of features provided by the library. The Scilab language is meant to be extended so that user-defined data types can be defined with possibly overloaded operations. Scilab users can develop their own module so that they can solve their particular problems. The Scilab language allows to dynamically compile and link other languages such as Fortran and C: this way, external libraries can be used as if they were a part of Scilab built-in features. Scilab also interfaces LabVIEW, a platform and development environment for a visual programming language from National Instruments.

From the license point of view, Scilab is a free software in the sense that the user does not pay for it and Scilab is an open source software, provided under the Cecill license [2]. The software is distributed with source code, so that the user has an access to Scilab most internal aspects. Most of the time, the user downloads and installs, a binary version of Scilab since the Scilab consortium provides Windows, Linux and Mac OS executable versions. An online help is provided in many local languages. From a scientific point of view, Scilab comes with many features. At the very beginning of Scilab, features were focused on linear algebra. But, rapidly, the number of features extended to cover many areas of scientific computing.

The following is a short list of its capabilities:
• Linear algebra, sparse matrices,
• Polynomials and rational functions,
• Interpolation, approximation,
• Linear, quadratic and non linear optimization,
• Ordinary Differential Equation solver and Differential Algebraic Equations solver,
• Classic and robust control, Linear Matrix Inequality optimization,
• Differentiable and non-differentiable optimization,
• Signal processing
• Statistics.

Scilab provides many graphics features, including a set of plotting functions, which allow to create 2D and 3D plots as well as user interfaces. The Xcos environment provides an hybrid dynamic systems modeler and simulator.

1.1 How to get and install Scilab Whatever your platform is (i.e. Windows, Linux or Mac), Scilab binaries can be downloaded directly from the Scilab homepage 3 Figure 1: Scilab console under Windows. http://www.scilab.org or from the Download area http://www.scilab.org/download Scilab binaries are provided for both 32 and 64 bits platforms so that it matches the target installation machine. Scilab can also be downloaded in source form, so that you can compile Scilab by yourself and produce your own binary. Compiling Scilab and generating a binary is especially interesting when we want to understand or debug an existing feature, or when we want to add a new feature. To compile Scilab, some prerequisites binary files are necessary, which are also provided in the Download center. Moreover, a Fortran and a C compiler are required. Compiling Scilab is a process which will not be detailed further in this document, because this chapter is mainly devoted to the external behavior of Scilab.

1.1.1 Installing Scilab under Windows Scilab is distributed as a Windows binary and an installer is provided so that the installation is really easy. The Scilab console is presented in figure 1. Several comments may be done about this installation process. On Windows, if your machine is based on an Intel processor, the Intel Math Kernel Library (MKL) [4] enables Scilab to perform faster numerical computations. 1.1.2 Installing Scilab under Linux Under Linux, the binary versions are available from Scilab website as .tar.gz files. There is no need for an installation program with Scilab under Linux: simply unzip the file in one target 4 directory. Once done, the binary file is located in /scilab-5.2.0/bin/scilab. When this script is executed, the console immediately appears and looks exactly the same as on Windows. Notice that Scilab is also distributed with the packaging system available with Linux distributions based on Debian (for example, Ubuntu). That installation method is extremely simple and efficient. Nevertheless, it has one little drawback: the version of Scilab packaged for your Linux distribution may not be up-to-date. This is because there is some delay (from several weeks to several months) between the availability of an up-to-date version of Scilab under Linux and its release in Linux distributions. For now, Scilab comes on Linux with a linear algebra library which is optimized and guarantees portability. Under Linux, Scilab does not come with a binary version of ATLAS [1], so that linear algebra is a little slower for that platform. 1.1.3 Installing Scilab under Mac OS Under Mac OS, the binary versions are available from Scilab website as a .dmg file. This binary works for Mac OS versions starting from version 10.5. It uses the Mac OS installer, which provides a classical installation process. Scilab is not available on Power PC systems. Scilab version 5.2 for Mac OS comes with a Tcl / Tk library which is disabled for technical reasons. As a consequence, there are some small limitations on the use of Scilab on this platform. For example, the Scilab / Tcl interface (TclSci), the graphic editor and the variable editor are not working. These features will be rewritten in Java in future versions of Scilab and these limitations will disappear. Still, using Scilab on Mac OS system is easy, and uses the shorcuts which are familiar to users of this platform. For example, the console and the editor use the Cmd key (Apple key) which is found on Mac keyboards. Moreover, there is no right-click on this platform. Instead, Scilab is sensitive to the Control-Click keyboard event. For now, Scilab comes on Mac OS with a linear algebra library which is optimized and guarantees

portability. Under Mac OS, Scilab does not come with a binary version of ATLAS [1], so that linear algebra is a little slower for that platform.

1.2 How to get help The most simple way to get the online help integrated to Scilab is to use the function help. The figure 2 presents the Scilab help window. To use this function, simply type "help" in the console and press the key, as in the following session. help Suppose that you want some help about the optim function. You may try to browse the integrated help, find the optimization section and then click on the optim item to display its help. Another possibility is to use the function help, followed by the name of the function which help is required, as in the following session. help optim Scilab automatically opens the associated entry in the help. We can also use the help provided on Scilab web site http://www.scilab.org/product/man 5 Figure 2: Scilab help window. That page always contains the help for the up-to-date version of Scilab. By using the "search" feature of my web browser, I can most of the time quickly find the help I need. With that method, I can see the help of several Scilab commands at the same time (for example the commands derivative and optim, so that I can provide the cost function suitable for optimization with optim by computing derivatives with derivative). A list of commercial books, free books, online tutorials and articles is presented on the Scilab homepage: http://www.scilab.org/publication

PRACTICAL: 2(a)

Aim: Study of matrix operation using Sci lab.

Ques. What is a matrix ?

Ans. In mathematics, a matrix (plural matrices) is a rectangular arrayof numbers, symbols, or expressions, arranged in rows and columns. For example, the dimension of the matrix below is 2 × 3 (read "two by three"), because there are two rows and three columns:

Types of matrixes

There are several types of matrices, but the most commonly used are:
- Rows Matrix
- Columns Matrix
- Rectangular Matrix
- Square Matrix
- Diagonal Matrix
- Scalar Matrix
- Identity Matrix
- Triangular Matrix
- Null or Zero Matrix
- Transpose of a Matrix

Row Matrix:A matrix is said to be a row matrix if it has only one row.

Column Matrix:A matrix is said to be a column matrix if it has only one column.

Rectangular Matrix:A matrix is said to be rectangular if the number of rows is not equal to the number of columns.

Square Matrix:A matrix is said to be square if the number of rows is equal to the number of columns.

Diagonal Matrix:A square matrix is said to be diagonal if at least one element of principal diagonal is non-zero and all the other elements are zero.

Scalar Matrix:A diagonal matrix is said to be scalar if all of its diagonal elements are the same.

Identity or Unit Matrix:A diagonal matrix is said to be identity if all of its diagonal elements are equal to one, denoted by II.

Triangular Matrix:A square matrix is said to be triangular if all of its elements above the principal diagonal are zero (lower triangular matrix) or all of its elements below the principal diagonal are zero (upper triangular matrix).

Null or Zero Matrixes:A matrix is said to be a null or zero matrix if all of its elements are equal to zero. It is denoted by OO.

1. **Creation of matrix**

    1.1 Create a row scaler matrix
        $\longrightarrow$
        ----   a= [1,2,3]
    1.2 Create a vector scaler matrix
        $\longrightarrow$
        ----   b=[1; 2; 3]
    1.3 Create a 3x3 matrix
        $\longrightarrow$
        ----   a=[1,2,3;4,5,6;7,8,9]
    1.4 Create a 3x3 matrix
        $\longrightarrow$
        ----   b=[1,2,3;4,5,6;7,8,9]

```
Scilab 6.0.2 Console
Startup execution:
    loading initial environment

--> a=[1,2,3]
 a   =

    1.    2.    3.


--> b=[1; 2; 3;]
 b   =

    1.
    2.
    3.

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

    1.    2.    3.
    4.    5.    6.
    7.    8.    9.


--> b=[1,2,3;4,5,6;7,8,9]
 b   =

    1.    2.    3.
    4.    5.    6.
    7.    8.    9.
```

2. Mathematical operations on matrix

2.1 Addition

---- $\longrightarrow$ c=a+b

2.2 Subtraction

---- $\longrightarrow$ d=a-b

2.3 Multiplication

---- $\longrightarrow$ e=a*b

2.4 Transporse

Suppose AA is a given matrix, then the matrix obtained by interchanging its rows into columns is called the transpose of AA. It is denoted by AtAt.

---- $\longrightarrow$ f=(1,2,3)

2.5 Inverse

---- $\longrightarrow$ inv(a)

2.6 Determinant

D ---- $\longrightarrow$ det(a)

2.7 Eiyen

---- $\longrightarrow$ spec(a)

2.8 Zeros of matrix

---- $\longrightarrow$ zeros(a)

2.9 Diagonal of matrix

---- $\longrightarrow$ diag(a)

2.10 Ones of matrix

---- $\longrightarrow$ ones(a)

```
Scilab 6.0.2 Console

-> c=a+b
c   =

    2.      4.      6.
    8.     10.     12.
   14.     16.     18.

-> d=a-b
d   =

    0.      0.      0.
    0.      0.      0.
    0.      0.      0.

-> e=a*b
e   =

   30.     36.     42.
   66.     81.     96.
  102.    126.    150.

-> f=[1 2 3]'
f   =

    1.
    2.
    3.
```

```
Scilab 6.0.2 Console
--> inv(a)
Warning :
matrix is close to singular or badly scaled. rcond = 2.2028E-18
 ans  =

    3.153D+15   -6.305D+15    3.153D+15
   -6.305D+15    1.261D+16   -6.305D+15
    3.153D+15   -6.305D+15    3.153D+15


--> det(a)
 ans  =

   -9.516D-16


--> spec(a)
 ans  =

    16.116844
   -1.116844
   -1.046D-15


--> zeros(a)
 ans  =

    0.    0.    0.
    0.    0.    0.
    0.    0.    0.
```

```
--> ones(a)
 ans  =

    1.    1.    1.
    1.    1.    1.
    1.    1.    1.


--> diag(a)
 ans  =

    1.
    5.
    9.
```

PRACTICAL: 2(b)

Aim: Find 30 terms using recursive function using Sci lab.

What is recursive function ?
A recursive function is a function that calls itself during its execution. This enables the function to repeat itself several times, outputting the result at the end of each iteration.The process of recursive calls always has to end up in a call that is solved directly, without the need of invoking the function again. This step will always be needed in order to avoid a never ending loop.

→
---   u(1)=1
→
   for n=1:30

→
   u(n+1)=u(n)+2*n+3

→
   disp([n,u(n)])

→
  end

```
--> for n=1:30
>  u(n+1)=u(n)+2*n+3
>  disp([n,u(n)])
>  end
```

Sci lab output:

| TERMS | 1 | 6 | 13 | 22 | 33 | 46 | 61 | 78 | 97 | 118 | 141 | 166 | 193 | 222 | 253 | 286 | 321 | 358 | 397 | 438 | 481 | 526 | 573 | 622 | 673 | 726 | 781 | 838 | 897 | 958 |

PRACTICAL: 3

Aim: Find the computer generated random numbers using Sci lab.

What are random numbers?

A random number is a number chosen as if by chance from some specified distribution such that selection of a large set of these numbers reproduces the underlying distribution. Almost always, such numbers are also required to be independent, so that there are no correlations between successive numbers. Computer-generated random numbers are sometimes called pseudorandom numbers while the term "random" is reserved for the output of unpredictable physical processes. When used without qualification, the word "random" usually means "random with a uniform distribution" Other distributions are of course possible. For example, the Box-Muller transformation allows pairs of uniform random numbers to be transformed to corresponding random numbers having a two-dimensional normal distribution.

--- $\longrightarrow$ X= rand(10,10,'Uniform')

The current random generator is set to a uniform random generator. Random numbers are uniformly distributed in the interval (0,1).

```
--> x=rand(10,10,'Uniform')
 x  =

        column 1 to 6

  0.2113249    0.5608486    0.3076091    0.5015342    0.2806498    0.4094825
  0.7560439    0.6623569    0.9329616    0.4368588    0.1280058    0.8784126
  0.0002211    0.7263507    0.2146008    0.2693125    0.7783129    0.113836
  0.3303271    0.1985144    0.312642     0.6325745    0.211903     0.1998338
  0.6653811    0.5442573    0.3616361    0.4051954    0.1121355    0.5618661
  0.6283918    0.2320748    0.2922267    0.9184708    0.6856896    0.5896177
  0.8497452    0.2312237    0.5664249    0.0437334    0.1531217    0.685398
  0.685731     0.2164633    0.4826472    0.4818509    0.6970851    0.8906225
  0.8782165    0.8833888    0.3321719    0.2639556    0.8415518    0.5042213
  0.068374     0.6525135    0.5935095    0.4148104    0.4062025    0.3493615

        column 7 to 10

  0.3873779    0.537623     0.587872     0.6488563
  0.9222899    0.1199926    0.4829179    0.9923191
  0.9488184    0.2256303    0.2232865    0.050042
  0.3435337    0.6274093    0.8400886    0.7485507
  0.3760119    0.7608433    0.1205996    0.4104059
  0.7340941    0.0485566    0.2855364    0.6084526
  0.2615761    0.672395     0.8607515    0.8544211
  0.4993494    0.2017173    0.8494102    0.0642647
  0.2638578    0.3911574    0.5257061    0.8279083
  0.5253563    0.8300317    0.993121     0.9262344


  --> rand("normal")

  --> rand('info')
   ans  =

   normal
```

$\longrightarrow$

--- Rand("normal")

The current random generator is set to a Gaussian (with mean 0 and variance 1) random number generator.

$\longrightarrow$

--- Rand('info')

return the type of the default random generator ('uniform' or 'normal')

$\longrightarrow$

--- Y= rand(x,'normal')

returns the current value of the seed.

```
--> y=rand(x,'normal')
 y  =


        column 1 to 6

   1.7487359  -1.3770621   0.2301981  -0.8575198   2.4976095   1.1316248
   1.8651792   0.7042731  -2.7290777  -0.1043591  -1.2875914   0.3759656
   0.1645912  -0.9063738  -0.2563031   0.2973099   0.6450695  -1.3667445
  -1.035891    0.2634747  -0.5003797   0.5308516   0.6696589  -0.0346505
   0.9182207   1.2296215   1.1937458  -1.5404673  -0.4483985  -1.3850463
  -0.9355485  -1.1579022  -1.5206394  -0.3966362  -1.5316782   0.3828792
   0.0259118  -0.4577385   1.8655071   0.5163255  -0.7218988  -1.6280467
   0.2720404   0.0168437   0.1910551   0.0075659  -2.3544971  -0.4386207
   0.7953703  -0.5875092  -1.3189197   1.0422456  -0.5485232   0.7757721
  -1.681167   -1.4029475   0.9307226   2.6705108   0.0207588  -0.6024774


        column 7 to 10

  -1.1049895   0.0979236  -1.0179837   0.282304
  -1.7970057   1.1997935   0.7794314   0.5000272
   0.7604477   0.2946056  -1.9598191  -2.0049863
   1.0562604  -0.382814    0.3252542   1.2498744
  -0.65881     1.1474321   0.0927515   1.1758126
   1.2431698  -0.6155043  -0.5864591   1.0197259
   0.1068464   0.2685505   1.2311603  -0.257048
  -0.9032336  -0.2032709   0.3027452  -1.0665698
   0.7469882   0.878063    0.6697461  -0.1595813
  -1.2220922   0.4345772  -0.269772   -2.696727
```

$\longrightarrow$

--- X=rand(2,2)

```
--> x=rand(2,2)
 x  =

   0.7131577  -1.2140244
  -0.4213568  -0.7860841
```

$\longrightarrow$

--- X=rand(2,2,2)

```
--> x=rand(2,2,2)
 x  =

(:,:,1)

  -2.3432411  -2.0499751
  -0.6921925   0.4215026
(:,:,2)

  -0.4183469  -0.2768419
  -0.481796   -0.0518056
```

Sci lab output :

```
--> x=rand(10,10,'Uniform')
 x  =

        column 1 to 6

  0.2113249   0.5608486   0.3076091   0.5015342   0.2806498   0.4094825
  0.7560439   0.6623569   0.9329616   0.4368588   0.1280058   0.8784126
  0.0002211   0.7263507   0.2146008   0.2693125   0.7783129   0.113836
  0.3303271   0.1985144   0.312642    0.6325745   0.211903    0.1998338
  0.6653811   0.5442573   0.3616361   0.4051954   0.1121355   0.5618661
  0.6283918   0.2320748   0.2922267   0.9184708   0.6856896   0.5896177
  0.8497452   0.2312237   0.5664249   0.0437334   0.1531217   0.685398
  0.685731    0.2164633   0.4826472   0.4818509   0.6970851   0.8906225
  0.8782165   0.8833888   0.3321719   0.2639556   0.8415518   0.5042213
  0.068374    0.6525135   0.5935095   0.4148104   0.4062025   0.3493615

        column 7 to 10

  0.3873779   0.537623    0.587872    0.6488563
  0.9222899   0.1199926   0.4829179   0.9923191
  0.9488184   0.2256303   0.2232865   0.050042
  0.3435337   0.6274093   0.8400886   0.7485507
  0.3760119   0.7608433   0.1205996   0.4104059
  0.7340941   0.0485566   0.2855364   0.6084526
  0.2615761   0.672395    0.8607515   0.8544211
  0.4993494   0.2017173   0.8494102   0.0642647
  0.2638578   0.3911574   0.5257061   0.8279083
  0.5253563   0.8300317   0.993121    0.9262344

--> rand("normal")
```

```
Scilab 6.0.2 Console                                                     ?  ↗

--> rand('info')
 ans  =

 normal


--> y=rand(x,'normal')
 y  =


        column 1 to 6

  1.7487359   -1.3770621    0.2301981   -0.8575198    2.4976095    1.1316248
  1.8651792    0.7042731   -2.7290777   -0.1043591   -1.2875914    0.3759656
  0.1645912   -0.9063738   -0.2563031    0.2973099    0.6450695   -1.3667445
 -1.035891     0.2634747   -0.5003797    0.5308516    0.6696589   -0.0346505
  0.9182207    1.2296215    1.1937458   -1.5404673   -0.4483985   -1.3850463
 -0.9355485   -1.1579022   -1.5206394   -0.3966362   -1.5316782    0.3828792
  0.0259118   -0.4577385    1.8655071    0.5163255   -0.7218988   -1.6280467
  0.2720404    0.0168437    0.1910551    0.0075659   -2.3544971   -0.4386207
  0.7953703   -0.5875092   -1.3189197    1.0422456   -0.5485232    0.7757721
 -1.681167    -1.4029475    0.9307226    2.6705108    0.0207588   -0.6024774
```

```
Scilab 6.0.2 Console

        column 7 to 10

 -1.1049895    0.0979236   -1.0179837    0.282304
 -1.7970057    1.1997935    0.7794314    0.5000272
  0.7604477    0.2946056   -1.9598191   -2.0049863
  1.0562604   -0.382814     0.3252542    1.2498744
 -0.65881      1.1474321    0.0927515    1.1758126
  1.2431698   -0.6155043   -0.5864591    1.0197259
  0.1068464    0.2685505    1.2311603   -0.257048
 -0.9032336   -0.2032709    0.3027452   -1.0665698
  0.7469882    0.878063     0.6697461   -0.1595813
 -1.2220922    0.4345772   -0.269772    -2.696727


--> x=rand(2,2)
 x  =

   0.7131577   -1.2140244
  -0.4213568   -0.7860841


--> x=rand(2,2,2)
 x  =

(:,:,1)

  -2.3432411   -2.0499751
  -0.6921925    0.4215026
(:,:,2)

  -0.4183469   -0.2768419
  -0.481796    -0.0518056
```

SM Lab File

PRACTICAL: 4

Aim: Write a Sci- lab program for inventory control management.

Consider the following conditions:

- There are 3-day lag between order and delivery/arrival. The merchandise is ordered at the end of the day(i) and received at the start of fourth day(i+3).

- For each unit of inventory the carrying cost of each night is Rs.0.75.

- Each unit out of stock when ordered results into the loss of goodwill worth Rs.2.0 per unit plus loss of Rs.16.00 net income, that would have resulted in its sale. Or a total loss of Rs. 18.00 per unit forever.

- The demand in a day can be for any number of units between 0 and 99, each equiprobable.

- There is never more than one replenishment order outstanding.

- Initially we have 115 units on hand and no reorder outstanding.

What is inventory condition?

- INVENTORY CONTROL : Webster's has defined Inventory as "The quantity of goods or the materials on hand"

- Goods or the materials are the essential element of any of the organization right from hospital, industry, private enterprise or the government department.

- Thus inventory control is the method of maintaining of stock at a level at which purchasing and stocking costs are at the lowest possible without interference with the supply.

- Thus it plays the vital role in maintaining the balance between the two. If the items like drugs are purchased in the large quantity, the supply can be made easily and immediately.

- The risk of the Out-Of-Stock is avoided.

```
P=input('p=');

Q=input('q=');

C=0.0

S=115

i(1)=1

UD=0

DD=0

ES=0

for i=1:180;

if(DD~=i) then

DEM=rand(00,99)*100;

else

S=S+Q

UD=0

end

if(DEM<S) then

S=S-DEM

C=C+(S)*75

else

C+((DEM)-(S))*18.0

S=0

UD=Q
```

DD=i+3
C=C+75.0

end

disp(C)

end

Sci lab output:


```
Scilab 5.5.2 Console                                    ? ↗ ×
        12450.

        12525.

        12600.

        12675.

        12750.

        12825.

        12900.

        12975.

        13050.

        13125.

        13200.

        13275.

        13350.

        13425.

        13500.

    C=13500
```

PRACTICAL: 5

Aim: Given the age of different persons with their frequencies, calculate simple mean of age and plot graph between age and frequency.
- **What is mean?**

The mean is the average of the numbers. It is easy to calculate: add up all the numbers, then divide by how many numbers there are. In other words it is the sum divided by the count.

- **What is frequency?**

A frequency distribution is a list, table or graph that displays the frequency of various outcomes in a sample. Each entry in the table contains the frequency or count of the occurrences of values within a particular group or interval.

- **What is a graph?**

A simple graph usually shows the relationship between two numbers or measurements in the form of a grid. If this is a rectangular graph using Cartesian coordinate system, the two measurements will be arranged into two different lines at right angle to one another. One of these lines will be going up (the vertical axis). The other one will be going right (the horizontal axis). These lines (or axes, the plural of axis) meet at their ends in the lower left corner of the graph.

Both of these axes have tick marks along their lengths. You can think of each axis as a ruler drawn on paper. So each measurement is indicated by the length of the associated tick mark along the particular axis.

A graph is a kind of chart or diagram. However, a chart or a diagram may not relate one quantity to other quantities. Flowchartsand tree diagrams are charts or diagrams that are not graphs.

-->age=[15,16,17,18,19,20];

--> freq=[2;5;11;9;14;13];

--> total=age*freq;

--> mean=total/sum(freq)

mean = 18.240741

--> Plot (age,freq)

Sci lab output :

```
Scilab 6.0.2 Console

--> age=[15,16,17,18,19,20];

--> freq=[2;5;11;9;14;13];

--> total=age*freq;

--> mean=total/sum(freq)
 mean  =

   18.240741


--> plot(age,freq)
WARNING: Transposing row vector X to get compatible dimensions
```
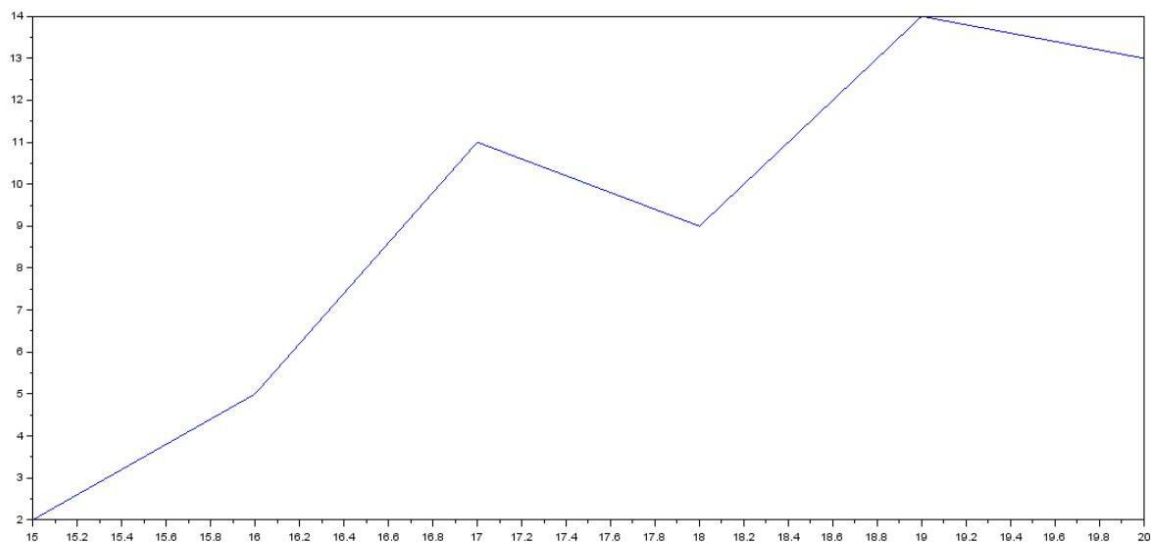
SM Lab File

PRACTICAL: 6

Aim: Find the expected profit if we have given 3 different types of profits and their probabilities.

Profits 1= 10;   prob 1=0.2

Profits 2= 20;   prob 2=0.8

Profits 3= 40;   prob 3=0.3

- **What is probability ?**

Probability is a numerical description of how likely an event is to occur or how likely it is that a proposition is true. Probability is a number between 0 and 1, where, roughly speaking, 0 indicates impossibility and 1 indicates certainty. The higher the probability of an event, the more likely it is that the event will occur. A simple example is the tossing of a fair (unbiased) coin. Since the coin is fair, the two outcomes ("heads" and "tails") are both equally probable; the probability of "heads" equals the probability of "tails"; and since no other outcomes are possible, the probability of either "heads" or "tails" is 1/2 (which could also be written as 0.5 or 50%).

--> profit=[10;20;40];

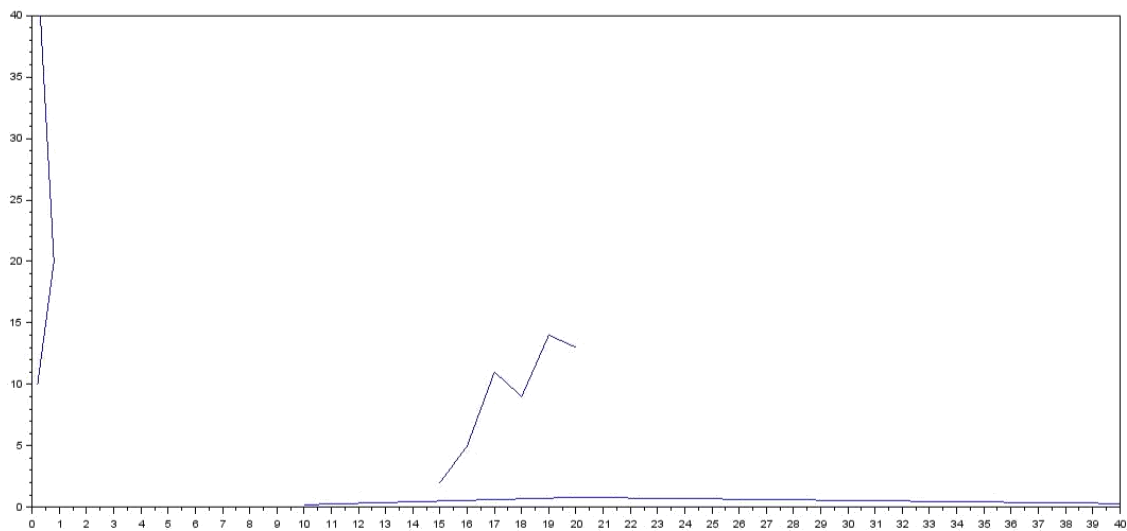--> prob=[0.2,0.8,0.3];

--> total=prob*profit

 total = 30

--> plot(profit,prob)

WARNING: Transposing row vector Y to get compatible dimensions

SM Lab File

Sci lab outputs

```
Scilab 6.0.2 Console

--> profit=[10;20;40];

--> prob=[0.2,0.8,0.3];

--> total=prob*profit
 total  =

    30.


--> plot(profit,prob)
WARNING: Transposing row vector Y to get compatible dimensions
```

PRACTICAL: 7

Aim: To develop a program that finds out whether a tank will overflow or not, write the shape of the tannk, its dimensions and rate of flow.

- **If –else condition**

The if/else statement executes a block of code if a specified condition is true. If the condition is false, another block of code can be executed.

The if/else statement is a part of JavaScript's "Conditional" Statements, which are used to perform different actions based on different conditions.

In JavaScript we have the following conditional statements:

- Use if to specify a block of code to be executed, if a specified condition is true
- Use else to specify a block of code to be executed, if the same condition is false
- Use else if to specify a new condition to test, if the first condition is false
- Use switch to select one of many blocks of code to be executed

--> f=input('enter the value of flow rate');

enter the value of flow rate 5

--> t=input('enter the time to fill the tank');

enter the time to fill the tank 2

--> r=input('enter the radius of the tank');

enter the radius of the tank 1

--> h=input('enter the height of the tank');

enter the height of the tank 2

--> vtank=%pi*r*r;

--> vliquid=f*t;

--> if(vliquid>vtank)

> then

> disp('tank is overflowing')

> else

> disp('tank is not overflowing')

> end

 tank is overflowing

Sci lab outputs:

```
Scilab 6.0.2 Console

--> f=input('enter the value of flow rate');
enter the value of flow rate 5


--> t=input('enter the time to fill the tank');
enter the time to fill the tank 2


--> r=input('enter the radius of the tank');
enter the radius of the tank 1


--> h=input('enter the height of the tank');
enter the height of the tank 2


--> vtank=%pi*r*r;

--> vliquid=f*t;

--> if(vliquid>vtank)
  > then
  > disp('tank is overflowing')
  > else
  > disp('tank is not overflowing')
  > end

 tank is overflowing
```

SM Lab File

PRACTICAL: 8

Aim: Find Mean and Variance of rolling the disc for 6 times

- **What is mean?**

The mean is the average of the numbers. It is easy to calculate: add up all the numbers, then divide by how many numbers there are. In other words it is the sum divided by the count.

- **What is Variance?**

Variance (σ_2) in statistics is a measurement of the spread between numbers in a data set. That is, it measures how far each number in the set is from the mean and therefore from every other number in the set.

- **How to calculate variance?**

To calculate the variance follow these steps:

- Work out the mean (the simple average of the numbers)
- Then for each number: subtract the Mean and square the result (the squared difference).
- Then work out the average of those squared differences.

```
→disp('Enter the value of dice roll')
→A(1,:)=[1,2,3,4,5,6]
→M=sum(A)/6;
→disp(M)
→for i=1:6;
→T(i)=(M - A(1,i))^2
→V=(sum(T(i))/5);
→end
→disp(V)
→plot(A(i),V,"*"
```

s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes

File Edit Format Options Window Execute ?

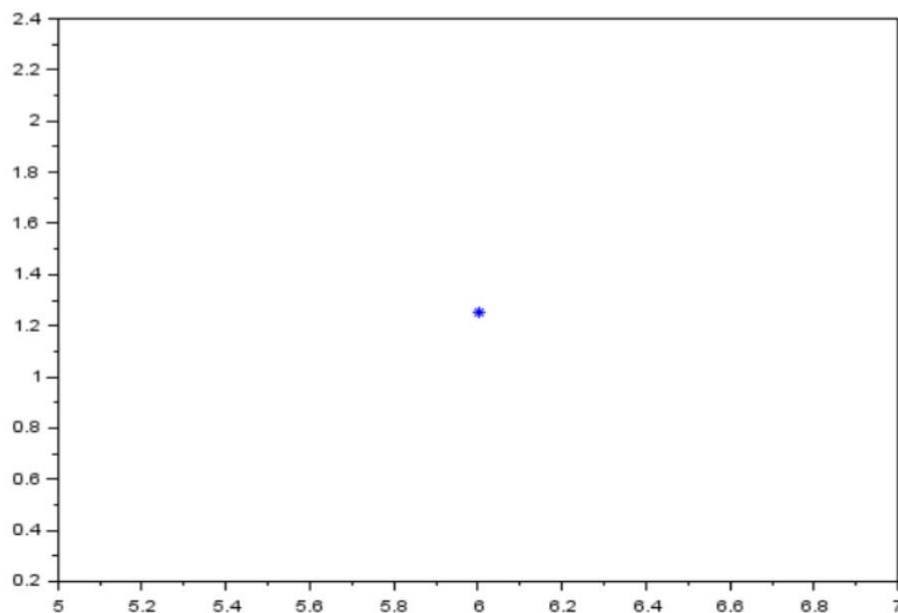s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes

s 8.sce

```
1  disp('Enter the value of dice roll');
2  A(1,:)=[1,2,3,4,5,6];
3  M=sum(A)/6;
4  disp(M)
5  for i=1:6;
6  T(i)=(M - A(1,i))^2;
7  V=(sum(T(i))/5);
8  end
9  disp(V);
10 plot(A(i),V,"*")
11
```

Graphic window number 0   —   □   ×

File Tools Edit ?

Graphic window number 0   ?

SM Lab File

PRACTICAL: 9

Aim: To find the mean and variance for where given data where age is represented by Z and frequency by Y.

- **What is mean?**

The mean is the average of the numbers. It is easy to calculate: add up all the numbers, then divide by how many numbers there are. In other words it is the sum divided by the count.

- **What is Variance?**

Variance (σ^2) in statistics is a measurement of the spread between numbers in a data set. That is, it measures how far each number in the set is from the mean and therefore from every other number in the set.

- **How is variance related to mean?**

The variance is the average of the squared differences from the mean. To figure out the variance, first calculate the difference between each point and the mean; then, square and average the results. For example, if a group of numbers ranges from 1 to 10, it will have a mean of 5.5.

```
→Z(1,:) =[46,53,29,61,36,39,47,49,52,38,55,32,57,54,44]
→C=sum(Z);
→disp(C)→Y(1,:) =[12,15,7,17,10,11,11,12,14,9,16,8,18,14,12];
→n=100;
→M=C/n;
→disp(M);
→for i=1:15
→M1=M*M;
→S=sum((Y(i)*(Z(i)*Z(i)))-(n*M1));
→SS=S/(n-1);
→end
→disp(M1)
→disp(SS)
```

```
Scilab 6.1.0 Console

--> editor

--> exec('C:\Users\Richa\Desktop\9.sce', -1)

   692.

   6.92

   47.8864

   186.29657

-->
```

9.sce (C:\Users\Richa\Desktop\9.sce) - SciNotes

File Edit Format Options Window Execute ?

9.sce (C:\Users\Richa\Desktop\9.sce) - SciNotes

9.sce

```
1  Z(1,:) =[46,53,29,61,36,39,47,49,52,38,55,32,57,54,44]
2  C=sum(Z);
3  disp(C)
4  Y(1,:) =[12,15,7,17,10,11,11,12,14,9,16,8,18,14,12];
5  n=100;
6  M=C/n;
7  disp(M);
8  for i=1:15
9  M1=M^M;
10 S=sum((Y(i)*(Z(i)*Z(i)))-(n*M1));
11 SS=S/(n-1);
12 end
13 disp(M1)
14 disp(SS)
```

SM Lab File

PRACTICAL: 10(A)

Aim: To find the Covariance and Correlation

- **What is correlation?**

Correlation is a statistical technique that can show whether and how strongly pairs of variables are related. For example, height and weight are related; taller people tend to be heavier than shorter people. The relationship isn't perfect. People of the same height vary in weight, and you can easily think of two people you know where the shorter one is heavier than the taller one. Nonetheless, the average weight of people 5'5'' is less than the average weight of people 5'6'', and their average weight is less than that of people 5'7'', etc. Correlation can tell you just how much of the variation in peoples' weights is related to their heights.

- **What is Covariance?**

Covariance measures the directional relationship between the returns on two assets. A positive covariance means that asset returns move together while a negative covariance means they move inversely. Covariance is calculated by analysing at-return surprises (standard deviation from the expected return) or by multiplying the correlation between the two variables by the standard deviation of each variable.

- **How to calculate covariance and correlation?**

Population Covariance Formula

$$Cov(x,y) = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{N}$$

Sample Covariance

$$Cov(x,y) = \frac{\Sigma(x_i - \bar{x})(y_i - y)}{N-1}$$

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[n\Sigma x^2 - (\Sigma x)^2][n\Sigma y^2 - (\Sigma y)^2]}}$$

→A=[4.0 2.0 0.60

 4.2 2.1 0.59

3.9 2.0 0.58

4.3 2.1 0.62

4.1 2.2 0.63];
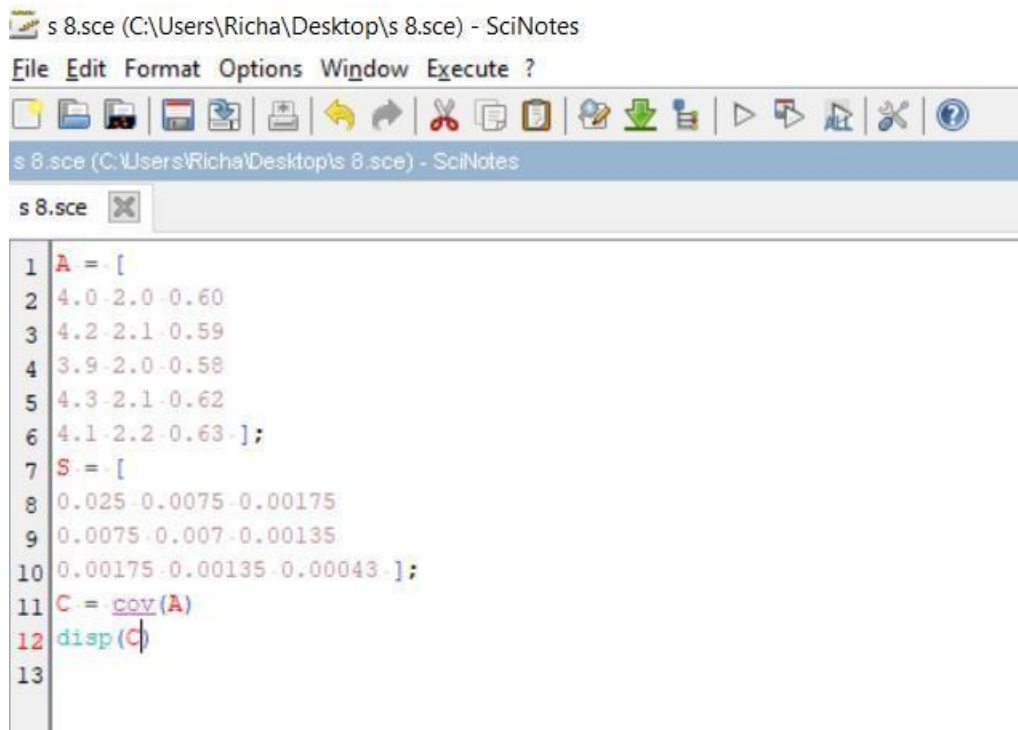
→S = [0.025 0.0075 0.00175

0.0075 0.007 0.00135

0.00175 0.00135 0.00043];

→C = cov(A)
→disp(C)

Sci lab output


```
s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes
File Edit Format Options Window Execute ?

s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes
s 8.sce
1  A = [
2  4.0 2.0 0.60
3  4.2 2.1 0.59
4  3.9 2.0 0.58
5  4.3 2.1 0.62
6  4.1 2.2 0.63 ];
7  S = [
8  0.025 0.0075 0.00175
9  0.0075 0.007 0.00135
10 0.00175 0.00135 0.00043 ];
11 C = cov(A)
12 disp(C)
13
```

```
Scilab 6.1.0 Console                                                    ? ↗ ✕

--> exec('C:\Users\Richa\Desktop\s 8.sce', -1)

   0.025      0.0075     0.00175
   0.0075     0.007      0.00135
   0.00175    0.00135    0.00043

-->
```

PRACTICAL: 10(B)

Aim: To find the Covariance

- **What is Covariance?**

Covariance measures the directional relationship between the returns on two assets. A positive covariance means that asset returns move together while a negative covariance means they move inversely. Covariance is calculated by analysing at-return surprises (standard deviation from the expected return) or by multiplying the correlation between the two variables by the standard deviation of each variable.

$\rightarrow$x = [230; 181; 165; 150; 97; 192; 181; 189; 172; 170];

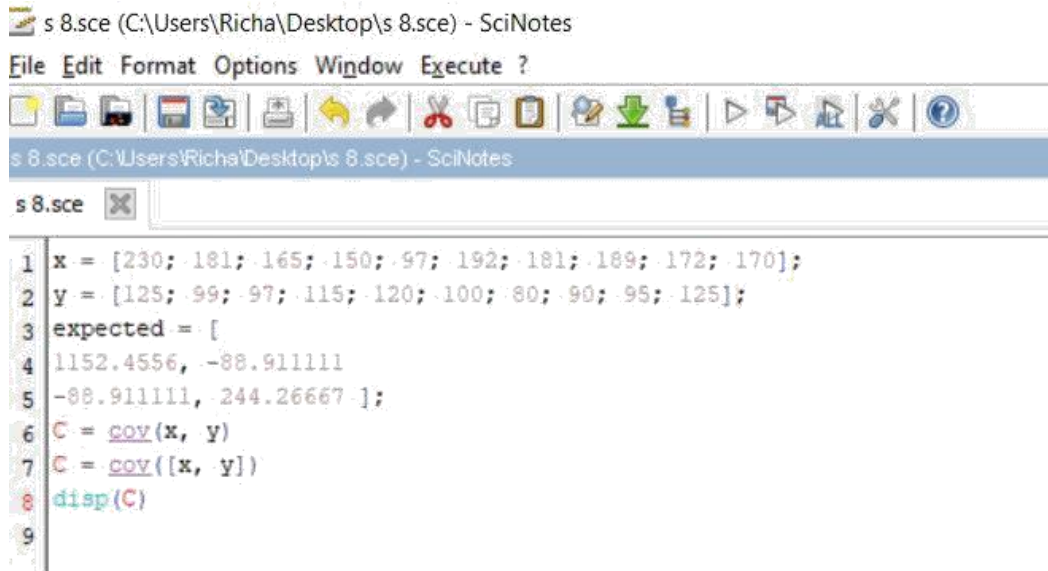$\rightarrow$y = [125; 99; 97; 115; 120; 100; 80; 90; 95; 125];

$\rightarrow$expected = [1152.4556, -88.911111 -88.911111, 244.26667];
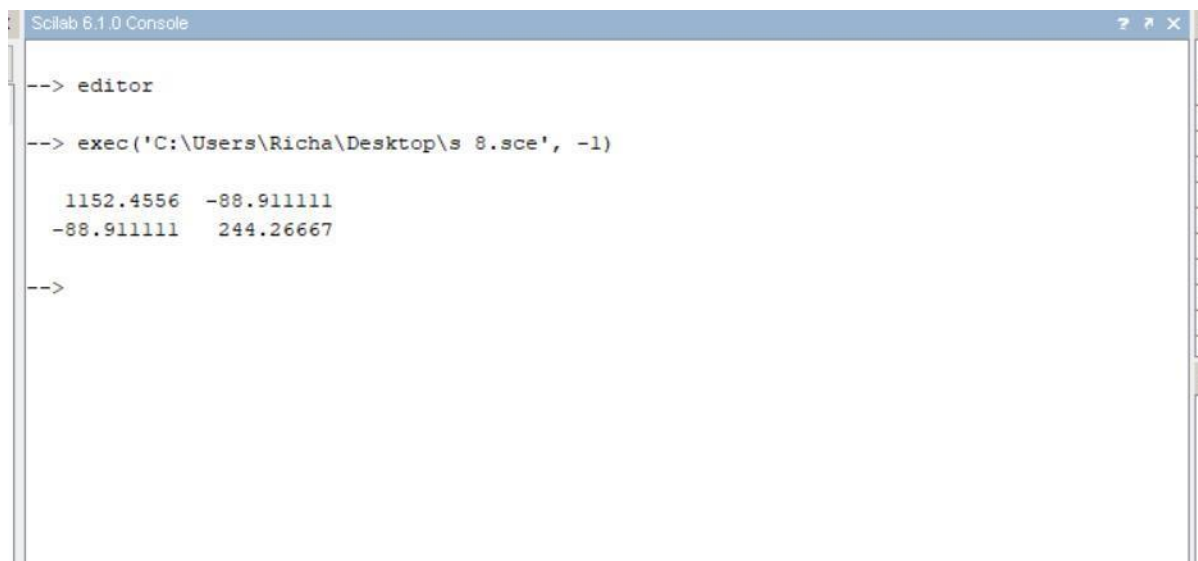
$\rightarrow$C = cov(x, y)

$\rightarrow$C = cov([x, y])

Sci lab output

s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes

File Edit Format Options Window Execute ?

s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes

s 8.sce

```
1  x = [230; 181; 165; 150; 97; 192; 181; 189; 172; 170];
2  y = [125; 99; 97; 115; 120; 100; 80; 90; 95; 125];
3  expected = [
4  1152.4556, -88.911111
5  -88.911111, 244.26667 ];
6  C = cov(x, y)
7  C = cov([x, y])
8  disp(C)
9
```

Scilab 6.1.0 Console

```
--> editor

--> exec('C:\Users\Richa\Desktop\s 8.sce', -1)

  1152.4556   -88.911111
 -88.911111    244.26667

-->
```

SM Lab File

PRACTICAL: 10(c)

Aim: To find the Correlation
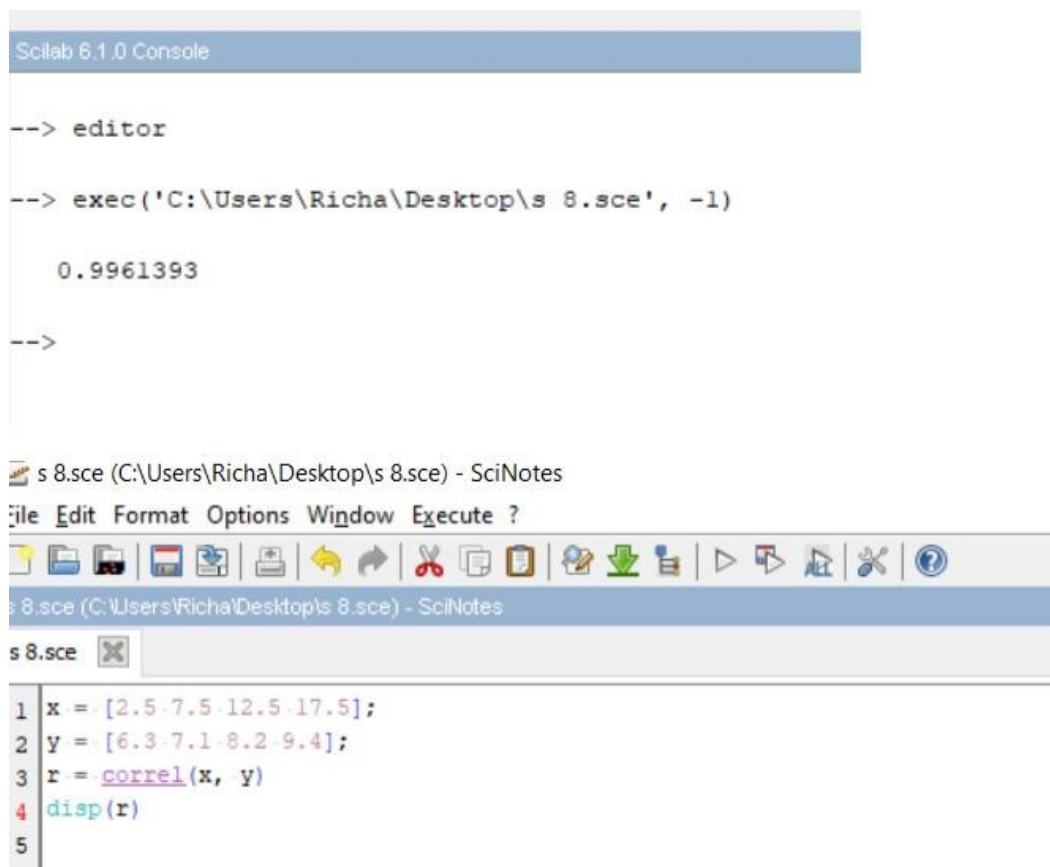
- **What is correlation?**

Correlation is a statistical technique that can show whether and how strongly pairs of variables are related. For example, height and weight are related; taller people tend to be heavier than shorter people. The relationship isn't perfect. People of the same height vary in weight, and you can easily think of two people you know where the shorter one is heavier than the taller one. Nonetheless, the average weight of people 5'5'' is less than the average weight of people 5'6'', and their average weight is less than that of people 5'7'', etc. Correlation can tell you just how much of the variation in peoples' weights is related to their heights.

→x = [2.5 7.5 12.5 17.5];
→y = [6.3 7.1 8.2 9.4];
→r = correl(x, y)

Sci lab output:

PRACTICAL: 11

Aim: We have a about vehicle performance, Miles per gallon is represented By matrix m and corresponding weight of car is represented by W matrix. Find Covariance and Correlation between these parameters. Plot the data set.

- **What is correlation?**

Correlation is a statistical technique that can show whether and how strongly pairs of variables are related. For example, height and weight are related; taller people tend to be heavier than shorter people. The relationship isn't perfect. People of the same height vary in weight, and you can easily think of two people you know where the shorter one is heavier than the taller one. Nonetheless, the average weight of people 5'5'' is less than the average weight of people 5'6'', and their average weight is less than that of people 5'7'', etc. Correlation can tell you just how much of the variation in peoples' weights is related to their heights.

Although this correlation is fairly obvious your data may contain unsuspected correlations. You may also suspect there are correlations, but don't know which are the strongest. An intelligent correlation analysis can lead to a greater understanding of your data

- **What is Covariance?**

Covariance measures the directional relationship between the returns on two assets. A positive covariance means that asset returns move together while a negative covariance means they move inversely. Covariance is calculated by analysing at-return surprises (standard deviation from the expected return) or by multiplying the correlation between the two variables by the standard deviation of each variable.

→m=[….]
→w=[….]
→C=cov(m,W)
→D=correl(m,W)
→xlabel("MPG"); ylabel("Weight"); plot(m,W,50)

Sci lab output

s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes

File Edit Format Options Window Execute ?

s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes

s 8.sce

```
1   m=[
2   18
3   15
4   18
5   16
6   17
7   15
8   14
9   14
10  14
11  15
12  15
13  14
14  15
15  14
16  24
17  22
18  18
19  21
20  27
21  26
22  25
23  24
24  25
25  26
26  21
27  10
28  10
29  11
30  9
31  27
32  28
33  25
34  19
35  16
36  17
37  19
38  18
39  14
40  14
```

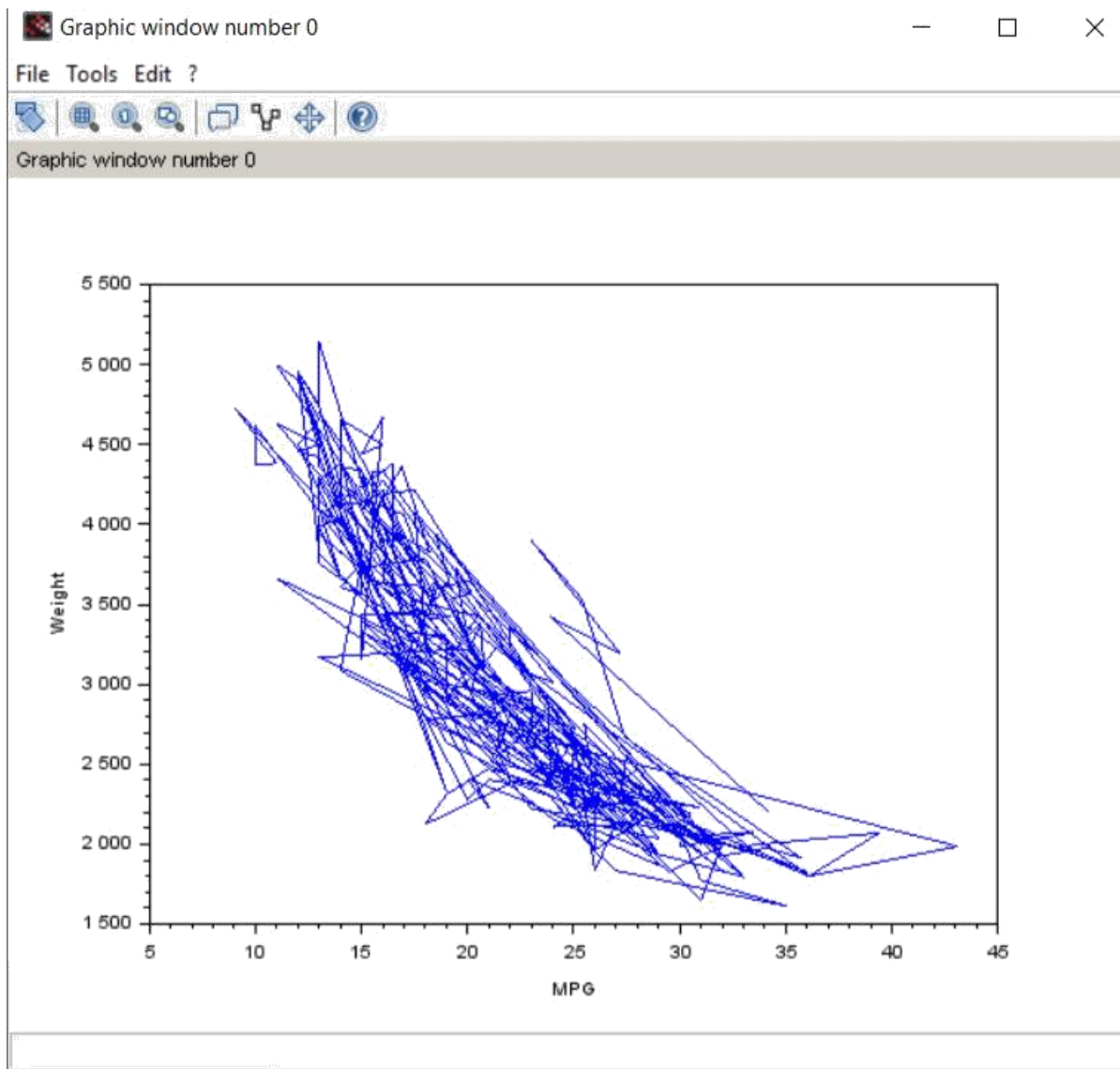s 8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes

File Edit Format Options Window Execute ?

8.sce (C:\Users\Richa\Desktop\s 8.sce) - SciNotes

s 8.sce

```
570  2300
571  2230
572  2515
573  2745
574  2855
575  2405
576  2830
577  3140
578  2795
579  3410
580  1990
581  2135
582  3245
583  2990
584  2890
585  3265
586  3360
587  3840
588  3725
589  3955
590  3830
591  4360
592  4054
593  3605
594  3940
595  1925
596  1975
597  1915
598  2670
599  3530
600  3900
601  3190
602  3420
603  2200
604  ]
605  C=cov(m,W)
606  D=correl(m,W)
607  xlabel("MPG"); ylabel("Weight"); plot(m,W,50)
608
609
```

Graphic window number 0

File Tools Edit ?

Graphic window number 0

PRACTICAL: 12

Aim: Write a program to find the structural stability of the given truss bridge.

Simulation steps to check the stability of the bridge:

STEP 1: Assume any definite shape (shapes made of straight lines).

STEP 2: Stability of the truss shall be determined using the formula, m = 2j-3 Where 'm' – No. of members in the given structure (nos.) 'j' – No. of joints in the given structure (nos.)

STEP 3: Conditions m< 2j-3   Unstable [Deficient Truss] m = 2j-3 Stable or Statically m > determinate [Perfect Truss] 2j-3 Statically indeterminate [Redundant Truss]

→M=input('Enter the Number of Members:');

→J=input('Enter the Number of Joints:');

→N=2*J-3;

→if M==N then

→disp('The Given Structure is Stable:');

→elseif M>N then

→disp('The Given Structure is In determine:');

→else

→disp('The Given Structure is Unstable:');

→end

Sci lab Output

12.sce (C:\Users\Richa\Desktop\12.sce) - SciNotes

File Edit Format Options Window Execute ?

12.sce (C:\Users\Richa\Desktop\12.sce) - SciNotes

12.sce

```
1  m=input('enter the number of members')
2  j=input('enter the no of joints')
3  n=2*j-3
4  if m==n then
5      disp('the given structure is stable')
6  elseif m>n then
7      disp('the given structure is indeterministic')
8  else
9      disp('the given structure is unstable')
10 end
```

Scilab 6.1.0 Console

```
--> editor

--> exec('C:\Users\Richa\Desktop\12.sce', -1)
enter the number of members 5

enter the no of joints 7


  "the given structure is unstable"

-->
```

Scilab 6.1.0 Console

```
--> editor

--> exec('C:\Users\Richa\Desktop\12.sce', -1)
enter the number of members 5

enter the no of joints 1


  "the given structure is indeterministic"
```

SM Lab File

Experiment- 13

Aim: Write a program to implement single server queue.

```
function FCFS()
n=input("Enter the no. of process :")
disp(" enter the burst time of process :")
for i=1:n
disp(i,"Process")
b(i)=input(" ")
a(i)=i
end
w(1)=0
avg=0
disp(w(1),a(1),"process waiting time:")
for i= 2:n
w(i)=b(i-1)+w(i-1)
disp(w(i),a(i),"Process waiting time")
avg=avg+w(i)
end


disp(avg,"total waiting time")
disp(avg/n,"total avg waiting time is")
tat(1)=b(1)
avg1=b(1)
disp(tat(1),a(1),"process turn around time:")
for k= 2:n
tat(k)=tat(k-1)+b(k)
disp(tat(k),a(k),"Process Turn around time:")
avg1=avg1+tat(k)
end
disp(avg1,"Total turn around time: ")
disp(avg1/n,"Total avg turn around time is; ")
//exec('C:\Users\Administrator\Desktop\mona sainiprjct\new
//prg\fcfs.sci', -1)
endfunction
FCFS()
```

Sci lab Output :

```
Enter the no. of process :2


  enter the burst time of process :

Process

  1.
5


Process

  2.
10


process waiting time:

  1.

  0.

Process waiting time

  2.

  5.
```

```
    5.

total waiting time

    5.

total avg waiting time is

    2.5

process turn around time:

    1.

    5.

Process Turn around time:

    2.

    15.

Total turn around time:

    20.

Total avg turn around time is;

    10.
```

PRACTICAL: 14

Aim : Write a program for pure pursuit problem using SCI lab.

- **What is pure pursuit?**

Pure pursuit is a tracking algorithm that works by calculating the curvature that will move a vehicle from its current position to some goal position. The whole point of the algorithm is to choose a goal position that is some distance ahead of the vehicle on the path. The name pure pursuit comes from the analogy that we use to describe the method. We tend to think of the vehicle as chasing a point on the path some distance ahead of it - it is pursuing that moving point. That analogy is often used to compare this method to the way humans drive. We tend to look some distance in front of the car and head toward that spot. This lookahead distance changes as we drive to reflect the twist of the road and vision occlusions.

Simulating fighter aircraft hitting a bomber

Station of a pure purse problem ample fighter aircraft sights an enemy bomber and flies directly toward it A in order to catch up with the bomber and destroy it The bomber the target) continuous lying (along a specified curve to the frontier the pure has to change its direction to keep pointed toward the target. We are interested in determining the attack course of the fighter and in mewing how long it would take for it to catch up with the bomber

If the target flies along a straight time, the problem can be solved directly with analytic techniques (The proof of such a closed-form expression which gives the course of the pure, when the target is in straight line is le as an exercise for you. Problem 1-2) However, in the path of the target is curved, the problem is much more difficult and normally cannot be solved directly. We Will use simulation to solve this problem, under the following simplifying conditions

1. The target and the pursuer are flying in the same horizontal plane when the fighter first sights the bomber, and both stay in the plane. This makes the pursuit model two-dimensional.

2. The lighter's speed the target path (e. its position as a function of time is specified. Enter a fixed time span Ar (every minute, in this case the fighter changes its direction in order to printer toward the bomber.

Let us introduce a rectangular coordinate system coincident with the horizontal plane in which the two aircraft are tying We choose the post due south of the night and due west of the target at the beginning of the be givenpur) as the origin of the coordinate system.

Explanation

C analytically we could not make a long-term prediction about the pat chat the fighter plane would take (in the initial position and path or are). But by simulation we were able to make the computer go there the inset-to predictions for as many instants as we wanted W possible only because we knew the basic process involved, namely at any particular instant.

The fighter plane system under study is essential for all simulation. Such knowledge of the simple strategy,ofpune redirecting sed intervals of time, while the target goes on ou ma y himself toward the tar predetermined pa AE effort to evade the pursuer, is called pure pursuit. in many situations, the strategy wed by the pursuer is more sophisticated.

Sci code: Xb=[100,110,120,129,140,149,158,168,179,188,198,209,219,226,234,240]

Yb=[0,3,6,10,15,20,26,32,37,34,30,27,23,19,16,14]

Xf=[];

Yf=[];

Xf[1]=0;

Yf[1]=50;
S=20;

dist=0;

for i=1:15;
plot(xb(i),yb(i),'r'');

```
title('pure persuit problem');

plot(xf(i),yf(i),'g'');
y=yb(i)-yf(i);

x=xb(i)-xf(i);

dist=sqrt(y^2+x^2);
if(dist<=12)
display('bomber destroyed at',s,i);
break;
end

xf(i+1)=xf(i)+s*((xb(i)-xf(i))/dist);

yf(i+1)=yf(i)+s*((yb(i)-yf(i))/dist);

end
```
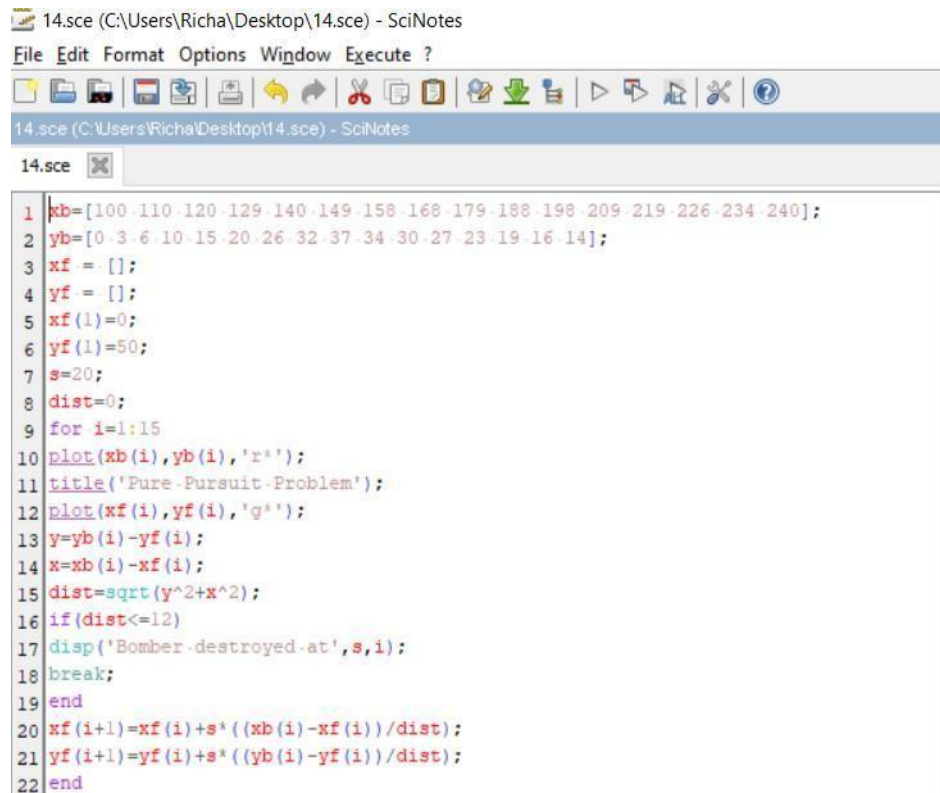
Sci lab output


```
14.sce (C:\Users\Richa\Desktop\14.sce) - SciNotes
File  Edit  Format  Options  Window  Execute  ?

14.sce (C:\Users\Richa\Desktop\14.sce) - SciNotes

14.sce

1  kb=[100 110 120 129 140 149 158 168 179 188 198 209 219 226 234 240];
2  yb=[0 3 6 10 15 20 26 32 37 34 30 27 23 19 16 14];
3  xf = [];
4  yf = [];
5  xf(1)=0;
6  yf(1)=50;
7  s=20;
8  dist=0;
9  for i=1:15
10 plot(xb(i),yb(i),'r*');
11 title('Pure Pursuit Problem');
12 plot(xf(i),yf(i),'g*');
13 y=yb(i)-yf(i);
14 x=xb(i)-xf(i);
15 dist=sqrt(y^2+x^2);
16 if(dist<=12)
17 disp('Bomber destroyed at',s,i);
18 break;
19 end
20 xf(i+1)=xf(i)+s*((xb(i)-xf(i))/dist);
21 yf(i+1)=yf(i)+s*((yb(i)-yf(i))/dist);
22 end
```