SCILAB BASICS

by

Prof. P. S. V. Nataraj

Department of System & Control Engineering

About SCILAB

- 1990- The software name Scilab became after Basile (A Computer Aided Control System Design) software.
- Inspired by Matlab, Fortron software developed by Cleve Moler who later cofounded with John Little "The MathWorks" Company.
- Scilab developed by INRIA (French National Institute for Research in Computer Science and Control) with Scilab group composed of six researchers.
- First release as open source, Scilab 1.1 on January 2nd 1994.

About SCILAB - Continued

- The Scilab Group, with active collaboration of external developers developed Scilab 2.7 distributing source and binary versions on internet, at end of 2002.
- 2003 – Creation of Scilab Consortium with the support of companies and academic organizations.
- 2008 – The Scilab Consortium (Phase 2) integrates the Digiteo Research Network

How to Download Scilab?

- Scilab can be downloaded from ftp://ftp.iitb.ac.in/misc_packages/Scilab/
- The website of Scilab is: www.scilab.org
- It is distributed in source code format.
- Binaries for Windows95/NT, Unix/Linux/Mac OS/X are also available. All the binary versions include tk/tcl interface.

Main Features of Scilab

- Environment for numerical computer applications
- Good Mathematical Library, compiled in C, uses ICC (Intel C Compiler)
- Interpreted high level language
- *.m file to Scilab Translator
- Fortran and C language compatibility through 'intersci' function
- Good graphics facility like Matlab
- Very Important : – Free Software

Main Features of Scilab -Continued

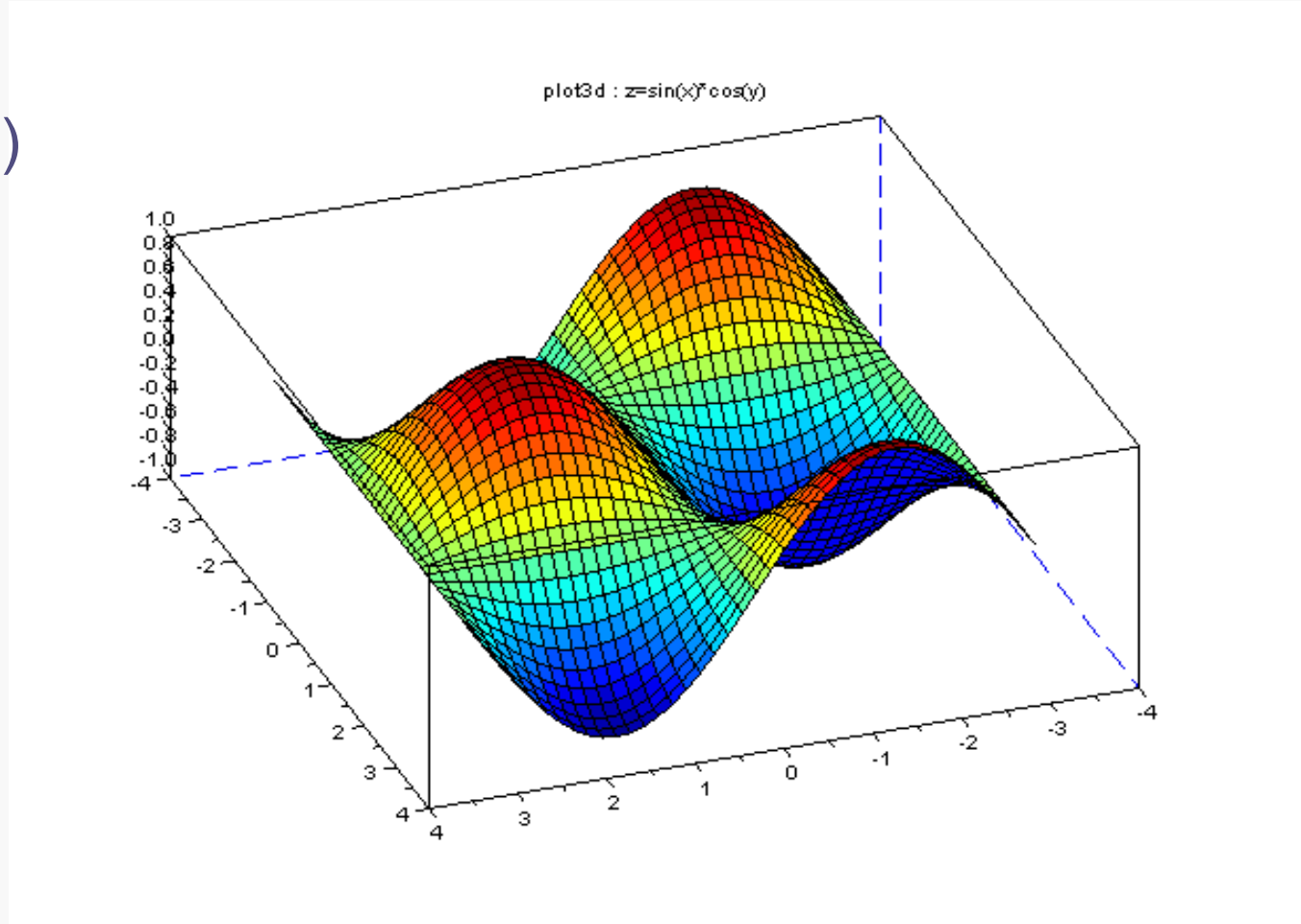
- Hundreds of mathematical functions
- High level programming language
- 2-D and 3-D graphics
- Advanced data structures and user defined data types
- Xcos: hybrid dynamic systems modeler and simulator

Main Features of Scilab -Continued

- 2-D and 3-D visualization :
  - Lines
  - Pie charts
  - Histograms
  - Surfaces
  - Animations
  - Graphics export in many formats: GIF, BMP, JPEG, SVG, PDF...

Main Features of Scilab -Continued

z=sin(x)*cos(y)



Main Features of Scilab -Continued

- Numerical computation
  - Linear algebra
  - Sparse matrices
  - Polynomials and rational functions
  - Simulation: explicit and implicit systems of differential equations solvers
  - Classic and robust control
  - Differentiable and non-differentiable optimization

Main Features of Scilab -Continued

- Data analysis
  - Interpolation, approximation
  - Signal Processing
  - Statistics

Mathematical Library

- Special functions
  - Bessel
  - Gamma
  - Error function
  - Elliptic integral
- Polynomials
  - Characteristic polynomial
  - Roots
  - Multiplication
  - Division
  - Curve Fitting

Mathematical Library -Continued

- Matrix functions
  - Exponential
  - Powers
  - Log
  - Square root
- Decomposition & factorization
  - LU
  - QR
  - SVD
  - Cholesky
  - Schur
  - Inverse

Mathematical Library -Continued

- Signal Processing
  - FFT, FFT2, IFFT, IFFT2
  - Convolution
  - Deconvolution
  - Correlation coefficient
- Maple Interface

Useful Commands

- demos

  Gives demos on several different things

- apropos

  Helps locate commands associated with a word

- help
- diary

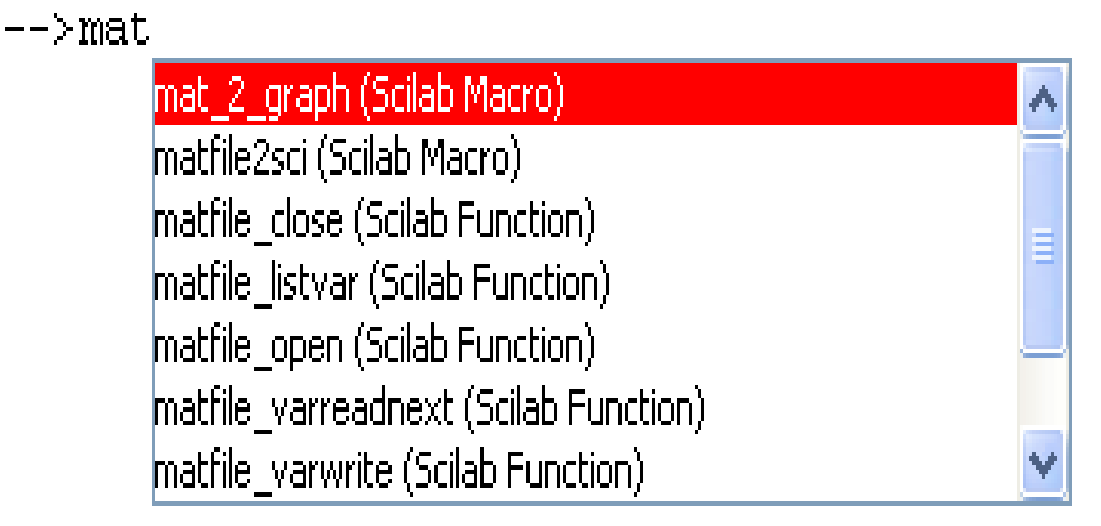
  Stores all commands and resulting outputs

- browsevar

  Variable list window

Try This ...

- Write some alphabets of command and use 'Tab' button of keyboard like this



Try This ...

→ x = 2; y = 33.5; z = 9.999;

(; signifies – not to show on console)

→ p = x+y+z;

- Now use keyboard's up arrow key and delete ' ;'

→ p=x+y+z

p =

45.499

Try This ...

→ Type 'format' on command prompt and hit 'Enter' key

```
ans  =

         1.    10.
```

→ Type 'format(20)' or 'format('v',20)' and hit 'Enter' key

→ p

```
p  =

         45.499000000000023
```

→ format('e',20)

→ p

Try This ...

Use last calculated 'p'

→ p

→ ceil(p)

→ floor(p)

→ fix(p)

→ round(p)

→ p=2.9

→ round(p)

→ help <ceil/floor/fix/round>

Try This ...

Different Ways to Specify a List:

→ x = [0 .1*%pi .2*%pi .3*%pi .4*%pi .5*%pi .6*%pi ...
.7*%pi .8*%pi .9*%pi %pi];

→ x = (0:0.1:1)*%pi;

→ x = linspace(0,%pi,11);

→ help <list/mlist/tlist>

Try This ...

→ ieee(1); 1/0

→ ieee(2);1/0,log(0)

→ help ieee

→ %e

→ sin(%pi/2), cos(%pi/2)

→ (10+5*%i)*(2*%i)

→ 5*cos(%pi/4)

→ clc

- clear the screen

Matrix/Cell Operation

→ x; (Use last x)

→ y=sin(x);

→ y(4);

→ z=cell(2,8);

→ z(2,2).entries= y(4);

→ z.dims;

→ a=[1 2 3], b=[2 3 4];

→ z(1,1).entries=a, z(2,1).entries=b;

→ help <cell/eye/once/zeros>

Matrix/Cell Operation

→ a';

→ a*b';

→ a.*b;

→ a'*b;

→ a'*b'; (error 20: Inconsistent multiplication.)

→ size(a);

→ length(a);

→ diag(a);

→ help <size/length/diag>

Matrix/Cell Operation

```
→ A=1:4;                    //This is a comment
→ B=2:2:8;                  // range
→ B([3 4]);                 // submatrix extraction
→ A(6)=6;                   // add an element
                            // oops! Forgot the fifth element!
→ A($-1)=5;                 // "$" is last element
→ B=2*A;                    // reassignment
→ B=[B 2*B; 3*B];           // new rows
→ B($ + 1,:)=4*B(1,:);
→ C=B([1 2], 2:$-1);        //submatrix extraction
```

Matrix Operation

```
→ A=1:4;                    //This is a comment
→ B=2:2:8;                  // range
→ B([3 4]);                 // submatrix extraction
→ A(6)=6;                   // add an element
                            // oops! Forgot the fifth element!
→ A($-1)=5;                 // "$" is last element
→ B=2*A;                    // reassignment
→ B=[B 2*B; 3*B];           // new rows
→ B($ + 1,:)=4*B(1,:);
→ C=B([1 2], 2:$-1);        //submatrix extraction
```

Machine Epsilon Operation

```
→ num=0; EPS=1;      //EPS- Variable to machine epsilon value
→ while (1+EPS)>1
→  EPS = EPS/2;
→  num = num+1;
→ end,
→ num
        num  =
                53.
→ EPS=2*EPS
       EPS  =
              2.220D-16
```

Machine Epsilon - Continued

Machine Epsilon value, can also be calculated by standard values:

For double precision machine, 'Cal' is

→ Cal=2^-53; // Cal- Variable name for Machine Epsilon

→ Cal=2*Cal

```
Cal  =

        2.220D-16
```

Time Computation

```
→ a = ones(10000,1);
→ timer()                    // CPU time
        ans =
            0.02
→ for i = 1:10000, b(i)=a(i)+a(i); end
→ timer()
        ans =
            0.31
→ help  <timer/tic/toc/profile>
```

Try This ...

→ pwd // Current Directory path

→ cd(“path-to-directory”) // OR 'File' >Change Current Directory'

→ diary(“my-record-of-what-follows.sci”)

→ help("diary")

→ inv([1 2; 0 4]);

→ C=rand(3,3);

→ C>.5; // boolean matrix

→ find(C>.5); // Showing index

→ C(find(C>.5));

→ disp(C);

File Extension and Execution

- Commands may be put in scripts.

  Extension is .sce (**SC**ilab **E**xecutable) : Default Type
- Contains only function definitions,

  Extension is .sci (**SCI**lab file)
- These are conventions!
- File Execution:
  - On console type

→ exec(`path-to-script/script-name.sce')

  - On Scipad, goto 'Execute' > 'Load into Scilab' (Ctrl+l)

Function Writing

```
function [y1, y2, ...]=foo(x1, x2, ...)
        statement
        statement
        statement
endfunction
```

Round bracket for input argument

Square bracket for output argument

OR

```
deff("[y]=foo([x])", "statements")
```

DEFined **F**unction in direct way

Function Loading into Scilab

- To load function definitions are in a script file, use getf('path/script.sci')
- If number of function '*.sci' files are in a directory, to load all function:
  getd <path-to-directory>
- During execution use 'Load all into Scilab (Ctrl+L)'
- To see the source of a Scilab coded function use fun2string(function-name)

Example: 'Function' writing

- Try This...

  save as '**myfunction.sci**':

```
function y = myfactorial(x)
    if x==0 then, y=1
    else y = x*myfactorial(x-1)
    end,
endfunction;
```

- After **setting current directory** path: use

  getf('path/myfunction.sci')

- For **execution**: use either 'exec('myfunction.sci')' on console OR on scipad editor 'Load into Scilab (Ctrl+l)'

Example - Continued

→ getf('myfunction.sci')
→ y=6;
→ myfactorial(y)

ans =

720.

→ // try a few examples:
→ myfactorial(5), myfactorial(0)

- Now calculate using **scilab command**:

→ factorial(5),factorial(0);

Input

→ name=input(“Enter your name:”)

// oops (try entering your name in \”)
or try this:

→ name=input(“Enter your name: ", “string”)

→ disp(name);
//more comfortable with C? try this:
→ mprintf(“Your name is %s", name)

Graphics

→ x=linspace(-%pi, %pi, 40)

→ plot(x,sin(x))

// Get plain 2D sine curve
→ plot2d(x,sin(x)) // Use Rotate button from toolbar

//Get lines within 2D sine curve
→ plot2d3(x,sin(x)) // Use Rotate button from toolbar

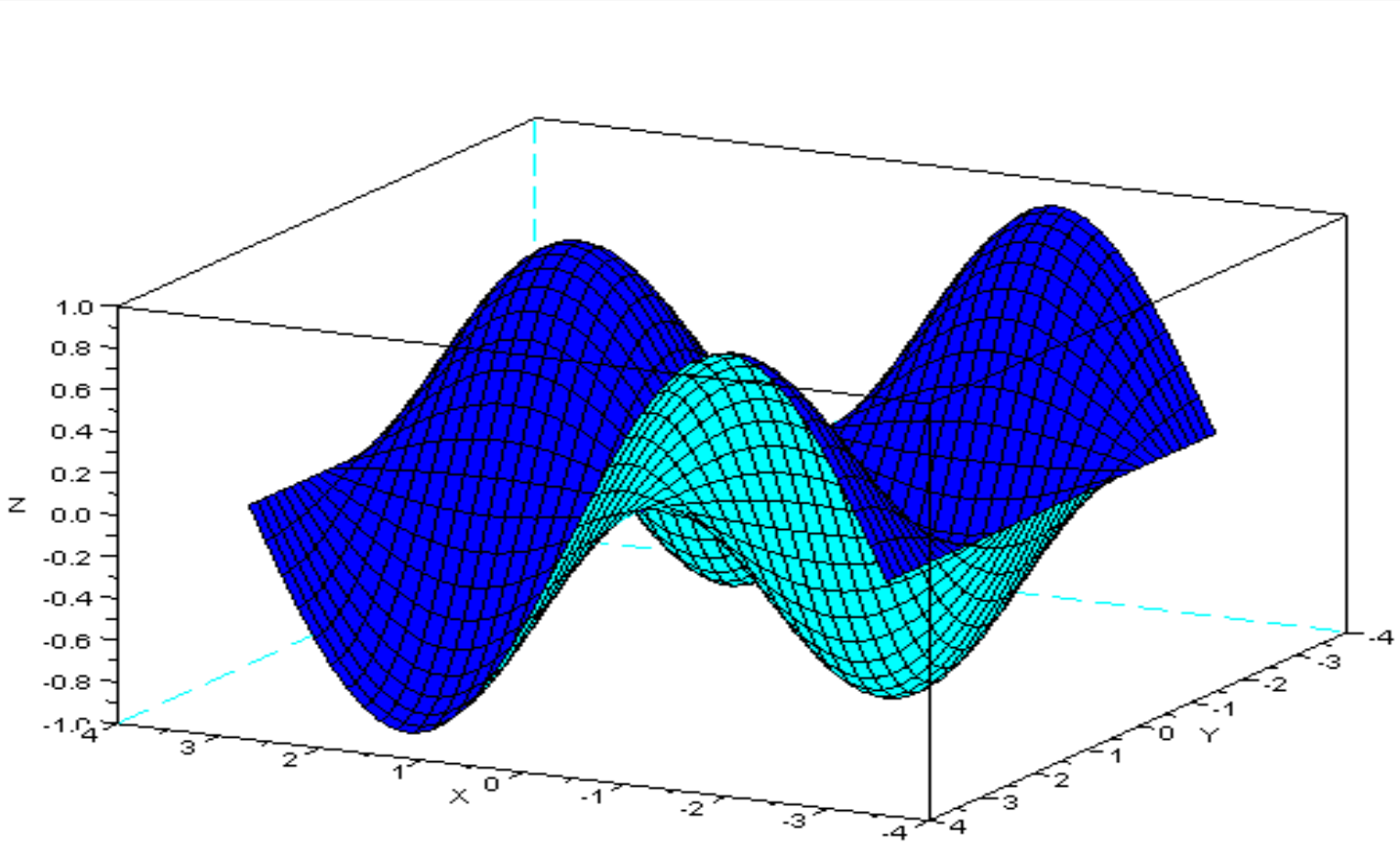
Graphics - Continued

→ y=x;

Notice Transpose

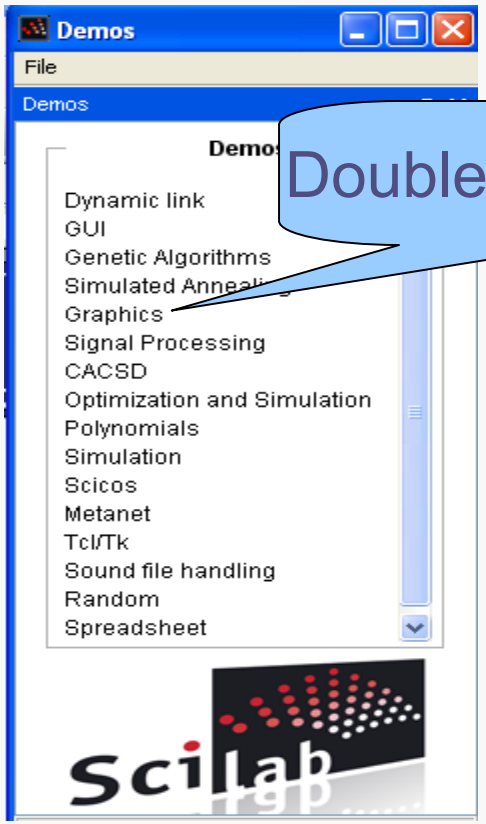
→ plot3d(x, y, sin(x)'*cos(y));

A=sin(x)' *cos(y)



Graphics Demos

- Click from console menu bar '?' and select 'Scilab Demonstrations'



Double click on Graphics

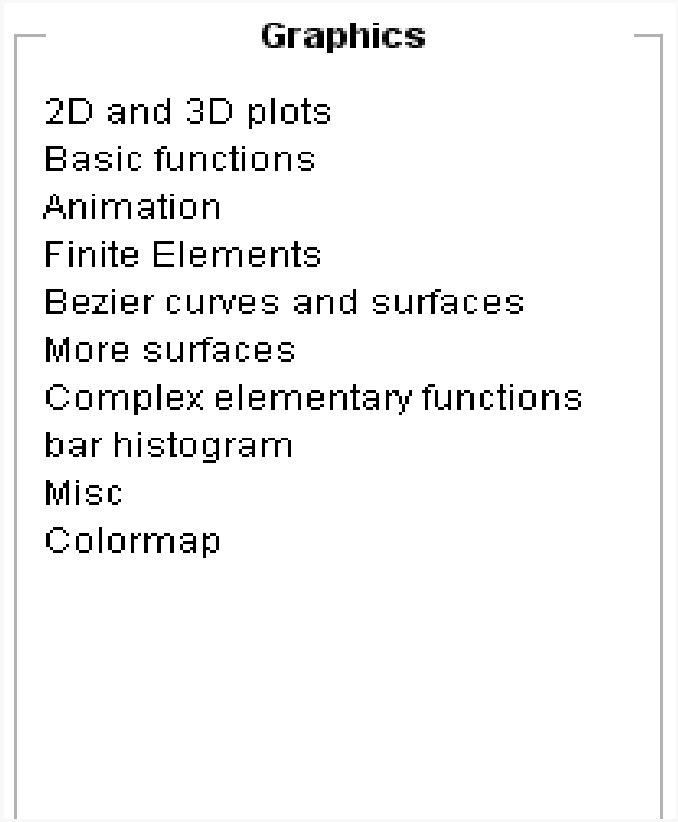
List Appears

Graphics

2D and 3D plots
Basic functions
Animation
Finite Elements
Bezier curves and surfaces
More surfaces
Complex elementary functions
bar histogram
Misc
Colormap

Graphics Demos - Continued

- Click on '2D and 3D plots':



Try out,

plot2D1
plot2D3
fplot3D1
contour etc.....

-

For viewing source code click on 'View Code' from menu bar

References

- www.scilab.org
- www.scilab.in
- http://scilab.in/cgi-bin/mailman/listinfo/scilab-india
- "Modeling and Simulation in Scilab/Scicos" by Stephen L.Campbell, Jean-Philippe Chancelier and Ramine Nikoukah,(Springer)

THANK YOU !!!