
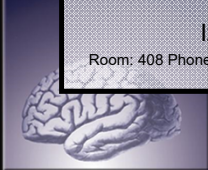

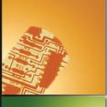




Introduction to programming 2017/18 Toolboxes & tools







Izhar Bar-Gad
Room: 408 Phone: 7141 Email: izhar.bar-gad@biu.ac.il



Week 12: Toolboxes

- Toolboxes
 - Statistics toolbox
 - Symbolic toolbox
- Miscellaneous tools


IBG



Toolboxes


- MATLAB features a family of **add-on** application-specific solutions called **toolboxes**.
- Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems and are installed under a **common directory** (or **directory tree**).

IBG




Toolboxes & Tools I

- *Mathworks* (the makers of MATLAB) features a few dozen toolboxes and other tools.
- The internet contains at least a factor more of additional toolboxes from other sources.
- Before starting a large software project look around for useful toolboxes.
- In this lecture we will discuss only a few toolboxes and miscellaneous tools.
- You will meet a few more in upcoming courses: "Signal & Data Analysis", "Neural Networks", etc.




Toolboxes & Tools II

- A few useful toolboxes for neuroscience
 - **Statistics toolbox**
 - **Symbolic toolbox**
 - Signal processing toolbox
 - Neural networks toolbox
 - Data Acquisition toolbox
- Miscellaneous tools
 - **MATLAB compiler**
 - **Distributed computing engine**
 - Real time targets




Examining a toolbox

- Use MATLAB's help system
 - *helpdesk* → *content* → *toolbox*
 - *help* toolboxName
- Study the examples of the toolbox
- Use *demo* to study the toolbox demos
- Use Mathworks' documentation
http://www.mathworks.com/products/product_listing/index.html




Statistics toolbox

- The statistics toolbox is a collection of tools typical to most toolboxes.
 1. Probability and statistics functions useful as building blocks for other programs
 2. Graphical, interactive tools for statistical processing
 3. Demonstration programs
- Functions may be grouped to different topics
 - Probability distributions
 - Descriptive statistics
 - Linear models
 - Statistical plots
 - Many others...







Statistics - Probability distributions

- Functions
 - Probability Density Functions (pdf)
`p=pdf('normal',2,0,1)` or `p=normpdf(2,0,1)` $p \rightarrow 0.0540$
 - Cumulative Distribution Function (cdf)
`p=cdf('normal',2,0,1)` or `p=normcdf(2,0,1)` $p \rightarrow 0.9772$
 - Inverse Cumulative Distribution Function
`x=icdf('normal',0.975,0,1)` or `norminv(0.975,0,1)` $x \rightarrow 1.96$
 - Random Number Generator
`x=random('normal',0,1)` or `x=randn` $x \rightarrow ?$
- Distributions
 - Normal
 - Binomial
 - ChiSquare
 - Gamma
 - ... (about 25)



Statistics – descriptive statistics

- Central tendency (location)
 - mean, median, mode, ...
- Dispersion
 - var, std, range
- Percentile
 - boxplot, prctile

IBG

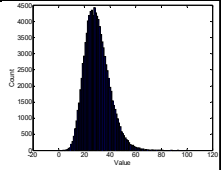
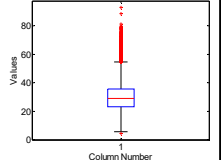
Statistics – descriptive statistics





```

g=random('gamma',10,3,[100000,1]);
hist(g,[0:1:100]);

mean(g) -> 30.0436
median(g) -> 29.0668
boxplot(g)

```








IBG

Statistics – linear models

- Linear models represent the relationship between a continuous response variable and one or more predictor variables.
- Regression
 - regress, polyfit, ...
- ANOVA
 - anova1, anova2, ...

IBG

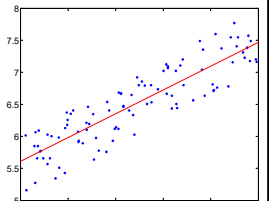
Statistics – linear models example


```

x=rand(100,1);
y=2*x+rand(100,1)+5;
b=regress(y,[x ones(100,1)]);
b -> 1.8531
      5.6115

plot(x,y,'b');
hold on
plot([0 1], b(2)+[0 1]*b(1),'r');


```



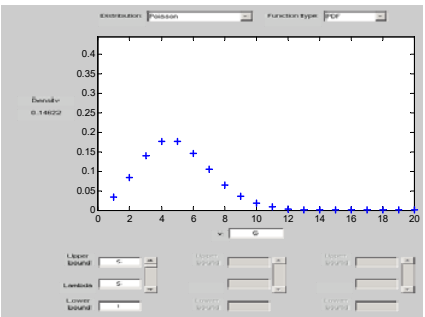



Statistics - Graphical tools

- In addition to the large number of functions, the toolbox provides a few graphical interactive programs.
- **disttool** - interactive plots of probability distributions functions (pdf & cdf).
- **dftool(data)** – Interactive fitting of data to distributions.
- **randtool** – Interactive generation of random numbers according to different distributions.

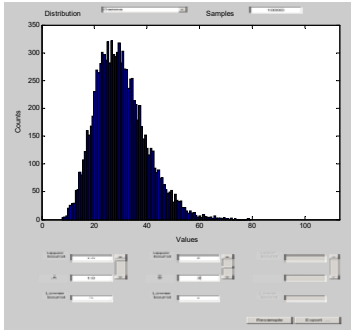






Disttool






Randtool







Symbolic toolbox


Numerical	Symbolic
Variables represent numbers	Variables are an entity by themselves
Answers may only be numbers	Answers can contain variables and functions.
Numeric computation is natural for the computer	Symbolic computation is unnatural to standard programming languages.







Symbolic toolbox and class

- The symbolic toolbox is an atypical toolbox as it is built around a unique class.
- Symbolic calculation in MATLAB is performed using the dedicated class of symbolic objects (**sym**).
- Much of the toolbox functionality is achieved by overloading the 'standard' numerical operators and functions with their symbolic counterparts.




Constructing symbolic objects


- Definition before initial usage
 - Syntax: `v=sym(name);`
Example: `symX = sym('x')` or `symVar = sym(5)`
 - Syntax: `syms symVar1 symVar2 ...`
Example: `syms symX symY symZ`
- Assignment using other symbolic variables
- **Example:**

```

symA = sym(3);
symX = sym('x');
symB =symA * symX;




```






Symbolic vs. Numeric

<u>Numeric</u>	<u>Symbolic</u>
numA = 12;	symA = sym(12);
numB = 18;	symB = sym(18);
numC = numA / numB	symC = symA / symB
numC → 0.6667	symC → 2/3

IBG



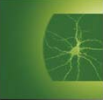


Symbolic toolbox: Calculus


- Calculus is built on two major complementary ideas
 - Differential calculus
 - Integral calculus
- Functions include: diff, int, limit, taylor
- Example single variable:


```
syms x;
f = sin(5*x);
d = diff(f)           d → 5*cos(5*x)
i = int(f)            i → 1/5*cos(5*x)
```
- Example multiple variable:


```
syms a b;
f = a * sin(b);
da = diff(f,a)        da → sin(b)
db = diff(f,b)        db → a*cos(b)
```

IBG






Symbolic toolbox: Equations I

- Algebraic equations are solved using `solve`
- Differential equations are solved using `dsolve`
- Example






```
syms a b c x
S = a*x^2 + b*x + c;

ss = solve(S)  ss → [1/2/a*(-b+(b^2-4*a*c)^(1/2))]
               [1/2/a*(-b-(b^2-4*a*c)^(1/2))]

bs = solve(S,b) bs → -(a*x^2+c)/x
```

IBG





Symbolic toolbox: Equations II

- Multiple equations


```
syms u v x y
S = solve(x+2*y-u, 4*x+5*y-v);
S.x -> -5/3*u+2/3*v
S.y -> 4/3*u-1/3*v
```
- Differential equations


```
S = dsolve('Dx=-x')      S -> C1*exp(-t)
S = dsolve('Dx=-x','x(0)=3')  S -> 3*exp(-t)
```





IBG

Symbolic toolbox: misc.

- Additional parts of the symbolic toolbox include:
 - Simplification of expressions
 - Linear algebra
 - Variable precision arithmetic
 - Special mathematical function
- The symbolic toolbox provides access to the underlying engine called **Maple**.

IBG


Linear algebra

```
>> syms t;
>> G = [cos(t) sin(t); -sin(t) cos(t)]
G =
[ cos(t), sin(t) ]
[ -sin(t), cos(t) ]

>> A = G*G
A =
[cos(t)^2-sin(t)^2, 2*cos(t)*sin(t)]
[-2*cos(t)*sin(t), cos(t)^2-sin(t)^2]


>> A = simple(A)
A =
[ cos(2*t), sin(2*t) ]
[ -sin(2*t), cos(2*t) ]
```

IBG




Extended Symbolic Math Toolbox

- The symbolic toolbox is a wrapper around some functionality of the **Maple** package.
- Full access to all of Maple (non-graphical) capabilities by running the MATLAB's *maple* function which can run code written the Maple language.







MATLAB - miscellaneous

- In addition to the function groups, MATLAB has other features packaged as toolboxes.
- **MATLAB compiler**
- **Distributed computing engine**
- Real time target
- Interaction with other languages



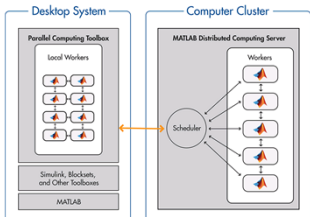
Matlab compiler

- Convert Matlab code to executables or libraries which may be distributed to non-Matlab environments.
- The simplest way to build an executable is `mcc -m fileName.m`
- To run the resulting *fileName.exe* file you must first install the runtime environment using *MCRInstaller.exe*.
- More complex cases are handled using the *deploytool*


IBG

Distributed computing

- Matlab is basically single threaded and thus does not use multiple computers or even multiple cores.



The diagram illustrates the architecture of distributed computing. It is divided into two main sections: 'Desktop System' and 'Computer Cluster'. The 'Desktop System' contains a 'Parallel Computing Toolbox' which manages 'Local Workers' (represented by four small icons) and interfaces with 'Simulink, Blockset, and Other Toolboxes' and 'MATLAB'. The 'Computer Cluster' contains a 'MATLAB Distributed Computing Server' which manages 'Workers' (represented by four small icons) and a central 'Scheduler'. A double-headed arrow connects the 'Parallel Computing Toolbox' on the desktop to the 'Scheduler' in the cluster, indicating bidirectional communication.