

MODULE I: INTRODUCTION

Data Analysis in MATLAB

I. Introducing MATLAB

What is MATLAB?

MATrix LABoratory

A data analysis toolbox

A financial modeling tool

A teaching tool

A signal processing platform

An optimization tool

A calculator

A bioinformatics framework

A programming language

A visualization suite

A modeling platform

A symbolic math tool

An application developer

A graphical plotter

What is MATLAB?

- “A numerical computing environment and fourth-gen programming language developed by MathWorks”
- wikipedia.org
- Initially developed by Cleve Moler, U. NM in 1970's
- Now a commercial for-profit product developed by Mathworks (Natick, MA)



Why MATLAB?

- Powerful interpreted language for math manipulations
- The “right” tool for data exploration & analysis
- Excellent, well-documented, and easy-to-use interface
- Many specialized and powerful toolboxes (stats, controls, bioinformatics, optimization,...)
- High-quality, customizable, publication-standard graphics
- Simple interfaces, but can access “under the hood”
- Freely available site license to all UIUC personnel (need VPN access for off-campus IP addresses)

Why **not** MATLAB?

- Interpreted language, and therefore **slow**...



- ...but Matlab compiler can generate stand-alone executables



What alternatives are there?

- Commercial competitors:
Maple, Mathematica, IDL
- Open source (free) alternatives:
GNU Octave, Python, Scilab, FreeMat
- Online GNU Octave server (useful in a pinch):
http://www.online-utility.org/math/math_calculator.jsp

Can't I use Excel?

- **The short answer**

NO.

Can't I use Excel?

- **The longer answer**
- It is a goal of this course to develop competency and proficiency in scientific software, including Matlab
- All analyses and plots submitted as part of the homework projects are expected to be done in Matlab

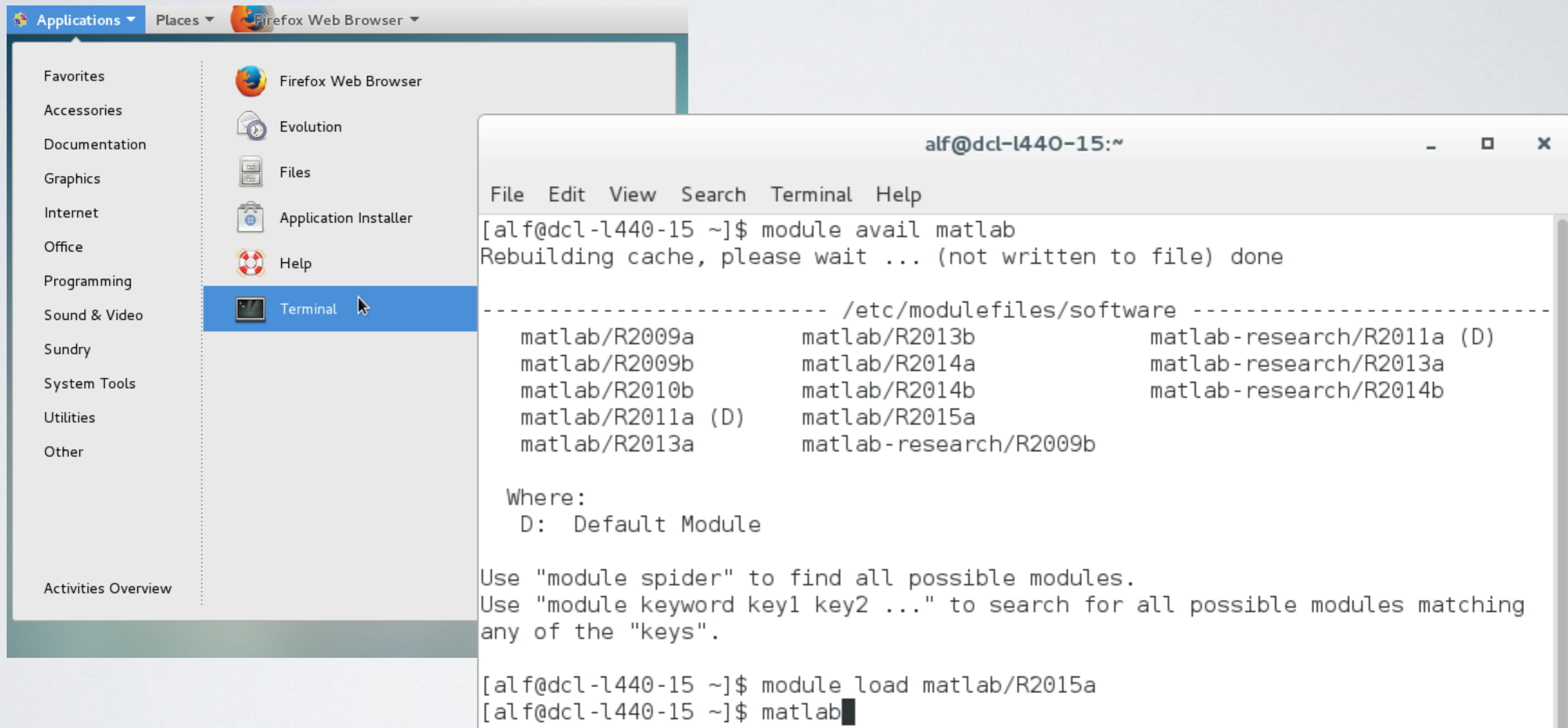
Can't I use Excel?

- **The real answer**
- Excel is a terrific tool for quick and dirty data analysis, data storage, and spreadsheets
- It lacks math firepower for sophisticated data analysis
- Analysis is invariably less efficient and clunkier than Matlab
- Graphics are not of publication quality

II. Basics

Loading MATLAB on EWS Linux

MATLAB must be loaded from the terminal, it is not currently available through the Applications menu



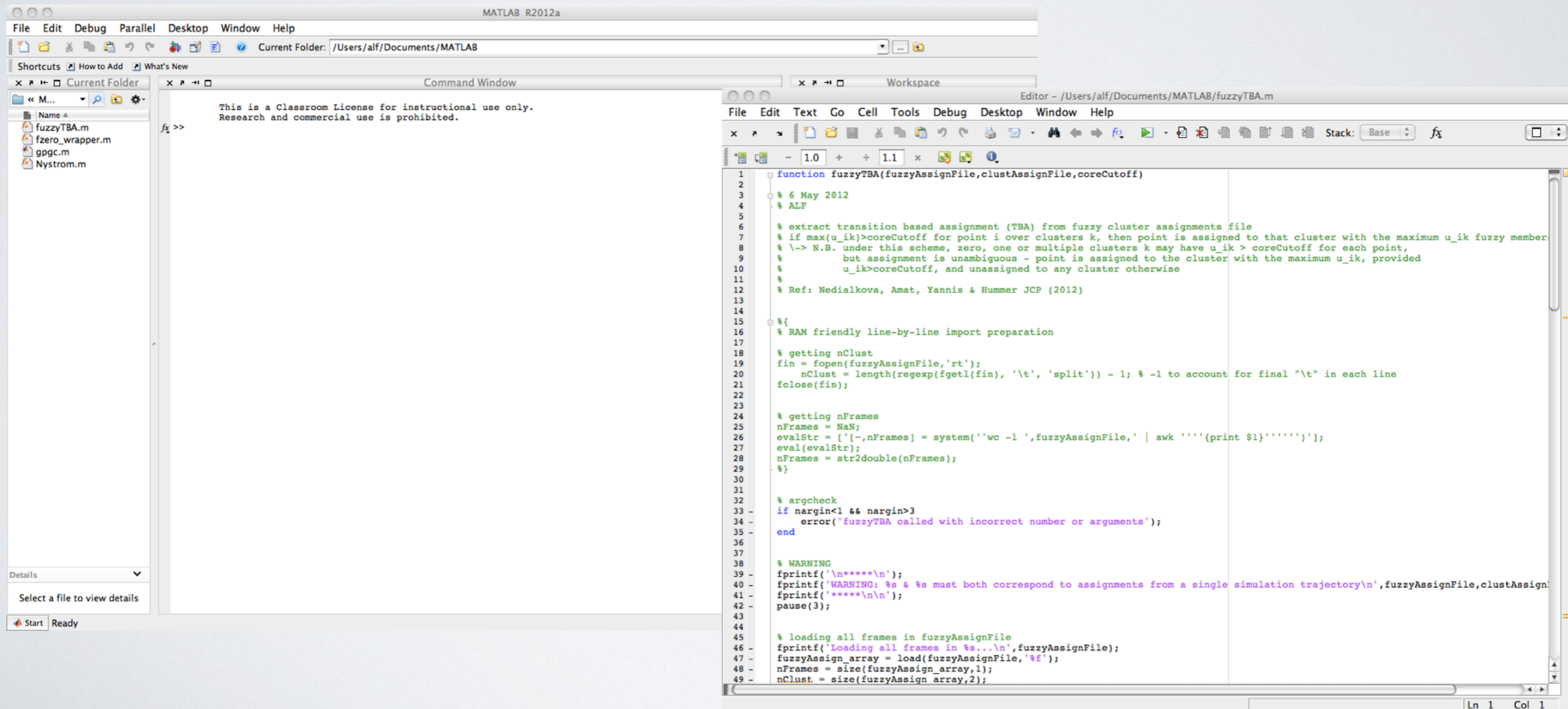
The screenshot shows a Linux desktop environment with the Applications menu open. The Terminal application is highlighted. The terminal window displays the following output:

```
alf@dcl-l440-15:~  
File Edit View Search Terminal Help  
[alf@dcl-l440-15 ~]$ module avail matlab  
Rebuilding cache, please wait ... (not written to file) done  
  
----- /etc/modulefiles/software -----  
matlab/R2009a          matlab/R2013b          matlab-research/R2011a (D)  
matlab/R2009b          matlab/R2014a          matlab-research/R2013a  
matlab/R2010b          matlab/R2014b          matlab-research/R2014b  
matlab/R2011a (D)      matlab/R2015a  
matlab/R2013a          matlab-research/R2009b  
  
Where:  
D: Default Module  
  
Use "module spider" to find all possible modules.  
Use "module keyword key1 key2 ..." to search for all possible modules matching  
any of the "keys".  
  
[alf@dcl-l440-15 ~]$ module load matlab/R2015a  
[alf@dcl-l440-15 ~]$ matlab
```

Bash ninjas may add `module load matlab/R2019a` to their `.bash_profile`

The MATLAB interface

- MATLAB is a high-level, interpreted programming language
- The main interface is through typing text into the command line or executing functions/scripts



The MATLAB interface

The screenshot displays the MATLAB R2012a interface. The top menu bar includes File, Edit, Debug, Parallel, Desktop, Window, and Help. The current folder is set to /Users/alf/Documents/MATLAB. The Command Window shows a message: "This is a Classroom License for instructional use only. Research and commercial use is prohibited." The Workspace panel is empty. The Command History panel shows a list of executed commands, including file operations and a 3D scatter plot command.

File Edit Debug Parallel Desktop Window Help

Current Folder: /Users/alf/Documents/MATLAB

Shortcuts How to Add What's New

Current Folder

Name ▲

- fuzzyTBA.m
- fzero_wrapper.m
- gpgc.m
- Nystrom.m

Details ▼

Select a file to view details

Start Ready

Command Window

```
f> >>
```

This is a Classroom License for instructional use only.
Research and commercial use is prohibited.

Workspace

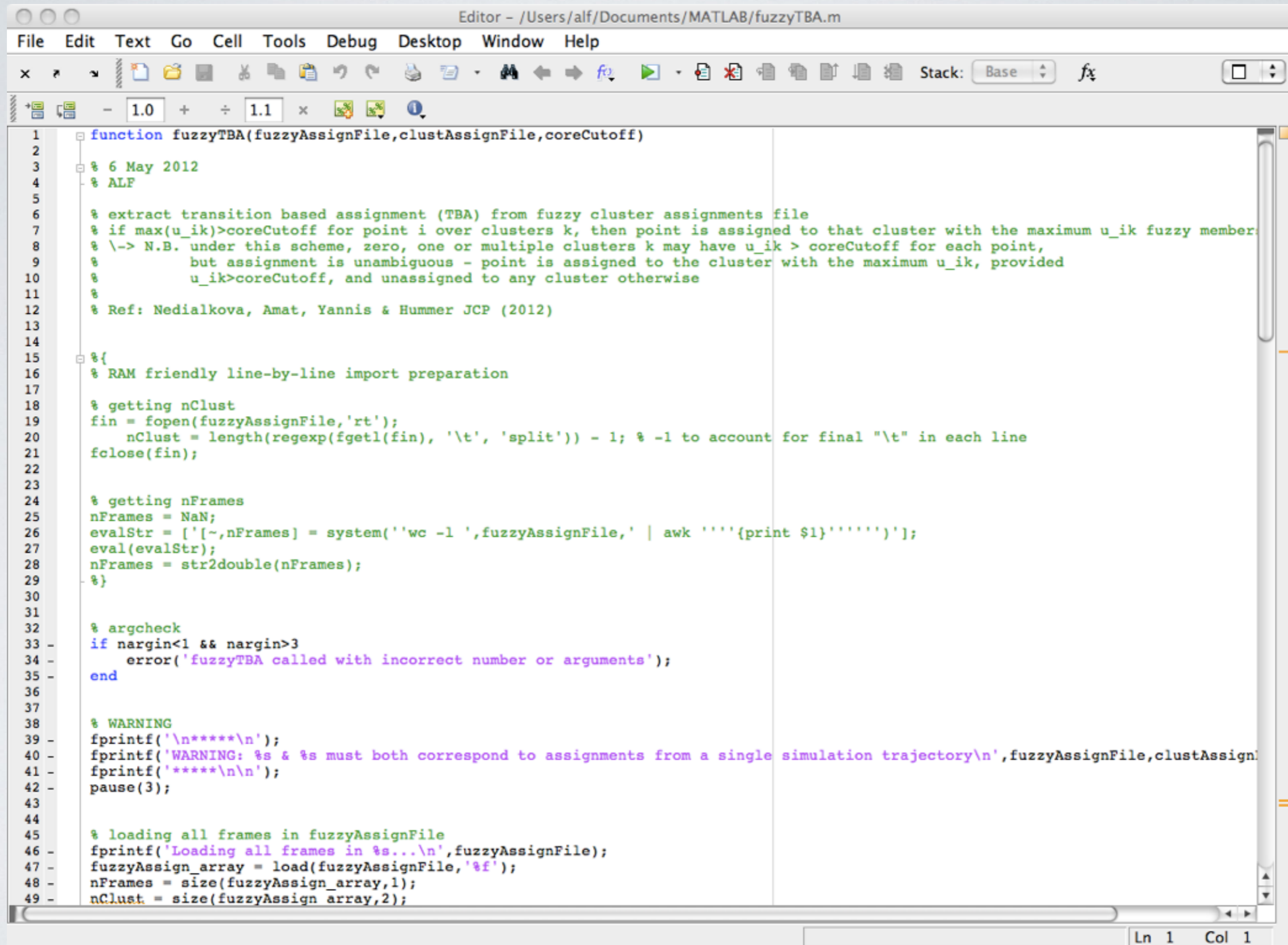
Select data to plot ▼

Name ▲	Value	Min	Max
--------	-------	-----	-----

Command History

```
mol_NP_v2  
close all  
%-- 7/30/13 12:12 PM --%  
run('/Users/alf/Dropbox/JJ_NP/3_sph  
close all  
cd('/Users/alf/Dropbox/JJ_NP/3_sph  
edit mol_NP_v2.m  
mol_NP_v2  
close all  
%-- 7/31/13 2:51 PM --%  
beadthread_array  
beadthread_array{1}  
cd('/Users/alf/Desktop/dMap')  
load('data_uniform_2500.mat','data'  
figure;  
scatter3(data(:,1),data(:,2),data(:  
epsScan('data_uniform_2500.mat',exp  
dMap('data_uniform_2500.mat',1.0,10  
close all  
clear all  
bar(evals)  
%-- 7/31/13 10:13 PM --%
```

The MATLAB interface



The screenshot shows the MATLAB Editor window with the following code:

```
1 function fuzzyTBA(fuzzyAssignFile, clustAssignFile, coreCutoff)
2
3 % 6 May 2012
4 % ALP
5
6 % extract transition based assignment (TBA) from fuzzy cluster assignments file
7 % if max(u_ik) > coreCutoff for point i over clusters k, then point is assigned to that cluster with the maximum u_ik fuzzy member
8 % \-> N.B. under this scheme, zero, one or multiple clusters k may have u_ik > coreCutoff for each point,
9 % but assignment is unambiguous - point is assigned to the cluster with the maximum u_ik, provided
10 % u_ik > coreCutoff, and unassigned to any cluster otherwise
11 %
12 % Ref: Nedialkova, Amat, Yannis & Hummer JCP (2012)
13
14
15 %{
16 % RAM friendly line-by-line import preparation
17
18 % getting nClust
19 fin = fopen(fuzzyAssignFile, 'rt');
20 nClust = length(regexpi(fgetl(fin), '\t', 'split')) - 1; % -1 to account for final "\t" in each line
21 fclose(fin);
22
23
24 % getting nFrames
25 nFrames = NaN;
26 evalStr = ['~-nFrames] = system('wc -l ', fuzzyAssignFile, ' | awk '{print $1}'');
27 eval(evalStr);
28 nFrames = str2double(nFrames);
29 %}
30
31
32 % argcheck
33 if nargin < 1 && nargin > 3
34 error('fuzzyTBA called with incorrect number or arguments');
35 end
36
37
38 % WARNING
39 fprintf('\n*****\n');
40 fprintf('WARNING: %s & %s must both correspond to assignments from a single simulation trajectory\n', fuzzyAssignFile, clustAssignFile);
41 fprintf('*****\n\n');
42 pause(3);
43
44
45 % loading all frames in fuzzyAssignFile
46 fprintf('Loading all frames in %s...\n', fuzzyAssignFile);
47 fuzzyAssign_array = load(fuzzyAssignFile, '%f');
48 nFrames = size(fuzzyAssign_array, 1);
49 nClust = size(fuzzyAssign_array, 2);
```

The status bar at the bottom right indicates "Ln 1 Col 1".

The MATLAB interface

- Can access GUI or CLI via EWS remote login

CLI: ssh

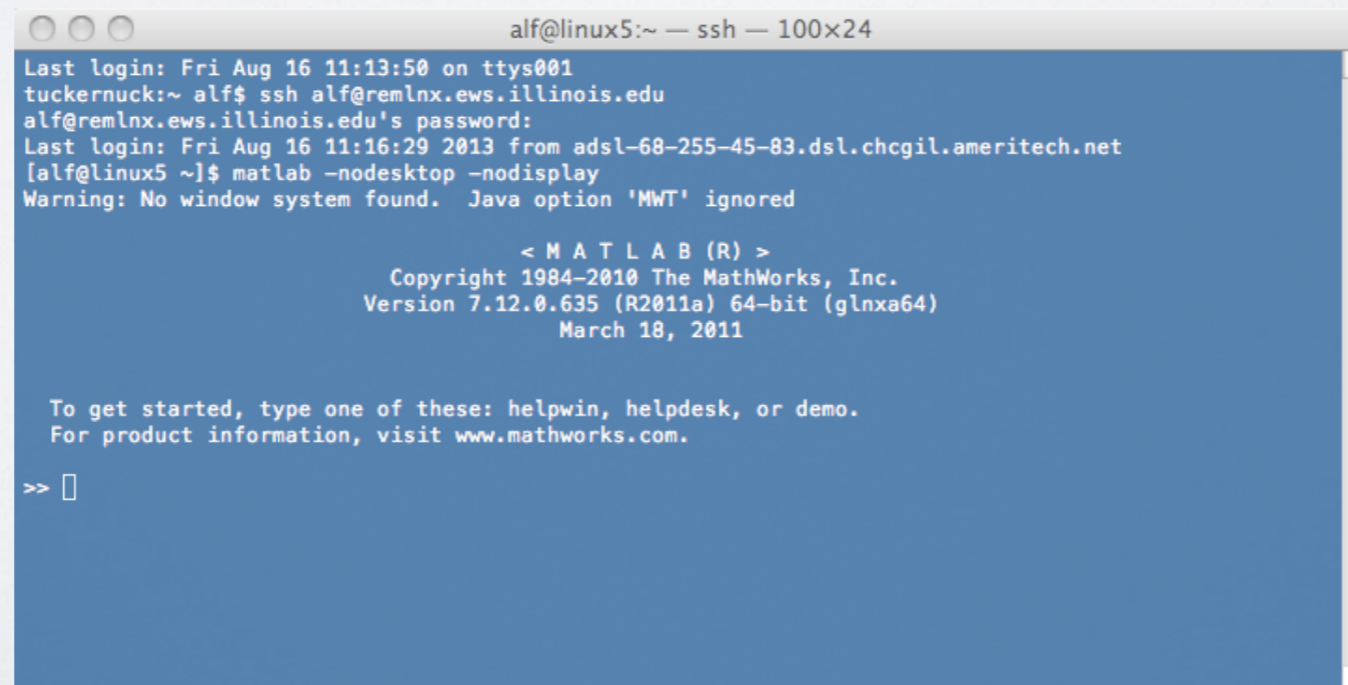
<username>@remlnx.ews.illinois.edu

module load matlab/R2019a

matlab -nodesktop -nodisplay

GUI: ssh -Y

<username>@remlnx.ews.illinois.edu



```
alf@linux5:~ -- ssh -- 100x24
Last login: Fri Aug 16 11:13:50 on ttys001
tuckernuck:~ alf$ ssh alf@remlnx.ews.illinois.edu
alf@remlnx.ews.illinois.edu's password:
Last login: Fri Aug 16 11:16:29 2013 from adsl-68-255-45-83.dsl.chcgil.ameritech.net
[alf@linux5 ~]$ matlab -nodesktop -nodisplay
Warning: No window system found. Java option 'MWT' ignored

      < M A T L A B (R) >
      Copyright 1984-2010 The MathWorks, Inc.
      Version 7.12.0.635 (R2011a) 64-bit (glnxa64)
      March 18, 2011

To get started, type one of these: helpwin, helpdesk, or demo.
For product information, visit www.mathworks.com.

>> █
```


The MATLAB ethos

Simplicity and versatility

- Naturally based around vectors, matrices and tensors
- Weakly typed
- Dynamically typed
- Structures and classes (OOP) supported
- User defined and built-in functions
- Powerful scripting and visualization interfaces

Calculator

The image shows the MATLAB R2012a interface. The main window is titled "MATLAB R2012a" and has a menu bar with "File", "Edit", "Debug", "Parallel", "Desktop", "Window", and "Help". Below the menu bar is a toolbar with various icons and a "Current Folder" path: "/Users/alf/Documents/MATLAB".

The interface is divided into several panes:

- Current Folder:** Shows a list of files: "fuzzyTBA.m", "fzero_wrapper.m", "gpgc.m", and "Nystrom.m".
- Command Window:** Contains the following text:

```
>> 10+17
ans =
    27
fx >>
```
- Workspace:** Displays a table with the following data:

Name	Value	Min	Max
ans	27	27	27
- Command History:** Shows a list of commands and their execution times:

```
close all
clear all
bar(evals)
%-- 7/31/13 10:13 PM --%
%-- 7/31/13 10:38 PM --%
10+17
clc
10+17
```

A "Start" button is located at the bottom left of the interface.

Creating variables

The image shows the MATLAB R2012a interface. The Command Window contains the following code and output:

```
>> a=27
a =
    27
>> b=[1 3 5 7 9]
b =
     1     3     5     7     9
>> c=[1 2 3; 6 9 12; 33 66 99]
c =
     1     2     3
     6     9    12
    33    66    99
```

The Workspace window shows the following table of variables:

Name	Value	Min	Max
a	27	27	27
ans	27	27	27
b	[1,3,5,7,9]	1	9
c	[1,2,3;6,9,12;33,66,99]	1	99

The Command History window shows the following commands:

```
mol_NP_v2
close all
%-- 7/31/13 2:51 PM --%
beadthread_array
beadthread_array{1}
cd('/Users/alf/Desktop/dMap')
load('data_uniform_2500.mat','data')
figure;
scatter3(data(:,1),data(:,2),data(:,3));
epsScan('data_uniform_2500.mat',exp
dMap('data_uniform_2500.mat',1.0,10)
close all
clear all
bar(evals)
%-- 7/31/13 10:13 PM --%
%-- 7/31/13 10:38 PM --%
10+17
10+17
clc
10+17
clc
a=27
b=[1 3 5 7 9]
```

Variable names

- Use short, descriptive names



index

sideLength

temperature



a

theSizeOfTheBoxAtTimeZero

fajfoiunejhnuhnnjkfa

- Naming rules: <63 characters
can't start with a number
not a keyword

- 17 keywords:

▪ for	▪ if
▪ function	▪ elseif
▪ otherwise	▪ continue
▪ try	▪ global
▪ break	▪ while
▪ end	▪ case
▪ return	▪ else
▪ switch	▪ persistent
▪ catch	

Calling functions

The image displays the MATLAB R2012a environment. The Command Window shows the execution of the command `bar(b)`. The Workspace window shows the following variables:

Name	Value	Min	Max
a	27	27	27
ans	27	27	27
b	[1,3,5,7,9]	1	9
c	[1,2,3,6,9,12;33,...]	1	99

The Figure window, titled 'Figure 1', displays a bar chart of the values in variable `b`. The x-axis is labeled 1 through 5, and the y-axis ranges from 0 to 9. The bars represent the values 1, 3, 5, 7, and 9.

Writing functions

The image shows the MATLAB R2012a environment with the following components:

- File Explorer:** Shows the current folder containing files: fuzzyTBA.m, fzero_wrapper.m, gpgc.m, and Nystrom.m.
- Command Window:** Shows the command `>> edit gpgc.m` and the prompt `fx >>`.
- Workspace:** Shows a table with columns for Name, Value, Min, and Max.
- Editor:** Displays the code for `gpgc.m`. The code includes:
 - Line 1: `%load('G.mat','G1','G2')`
 - Line 3: `G1=[`
 - Lines 4-10: Matrix G1 with values:

```
0 1 0 1 0 0 0
1 0 1 0 0 0 0
0 1 0 1 1 0 0
1 0 1 0 0 0 0
0 0 1 0 0 1 1
0 0 0 0 1 0 0
0 0 0 0 1 0 0
```
 - Line 11: `];`
 - Line 12: `G2=[`
 - Lines 13-17: Matrix G2 with values:

```
0 1 0 0 0
1 0 1 0 0
0 1 0 1 1
0 0 1 0 0
0 0 1 0 0
```
 - Line 18: `];`
 - Line 20: `n1=size(G1,1);`
 - Line 21: `if size(G1,2)~=n1`
 - Line 22: `error('G1 is not square');`
 - Line 23: `end`
 - Line 25: `n2=size(G2,1);`
 - Line 26: `if size(G2,2)~=n2`
 - Line 27: `error('G2 is not square');`
 - Line 28: `end`
 - Line 30: `H1=zeros(n1);`
 - Line 31: `H2=zeros(n2);`
 - Line 33: `for i=1:n1`
 - Line 34: `for j=1:n1`
 - Line 35: `if i==j`
 - Line 36: `H1(i,j)=1;`
 - Line 37: `elseif G1(i,j)==1`
 - Line 38: `H1(i,j)=exp(-G1(i,j));`
 - Line 39: `end`
 - Line 40: `end`
 - Line 41: `end`
 - Line 43: `for i=1:n2`
 - Line 44: `for j=1:n2`
 - Line 45: `if i==j`
 - Line 46: `H2(i,j)=1;`
 - Line 47: `elseif G2(i,j)==1`
 - Line 48: `H2(i,j)=exp(-G2(i,j));`
 - Line 49: `end`
 - Line 50: `end`

14 usages of "n1" found

script

Ln 21 Col 18

Getting help

The screenshot shows the MATLAB R2012a interface with the Command Window displaying the help text for the `bar` function. The Workspace window shows variables `a`, `ans`, `b`, and `c`. The Command History window shows the sequence of commands executed.

```
>> help bar
bar Bar graph.
bar(X,Y) draws the columns of the M-by-N matrix Y as M groups of N
vertical bars. The vector X must not have duplicate values.

bar(Y) uses the default value of X=1:M. For vector inputs, bar(X,Y)
or bar(Y) draws LENGTH(Y) bars. The colors are set by the colormap.

bar(X,Y,WIDTH) or bar(Y,WIDTH) specifies the width of the bars. Values
of WIDTH > 1, produce overlapped bars. The default value is WIDTH=0.8

bar(...,'grouped') produces the default vertical grouped bar chart.
bar(...,'stacked') produces a vertical stacked bar chart.
bar(...,LINESPEC) uses the line color specified (one of 'rgbymckw').

bar(AX,...) plots into AX instead of GCA.

H = bar(...) returns a vector of handles to barseries objects.

Use SHADING FACETED to put edges on the bars. Use SHADING FLAT to
turn them off.

Examples: subplot(3,1,1), bar(rand(10,5),'stacked'), colormap(cool)
          subplot(3,1,2), bar(0:.25:1,rand(5),1)
          subplot(3,1,3), bar(rand(2,3),.75,'grouped')

See also hist, plot, barh, bar3, bar3h.

Overloaded methods:
  fints/bar

Reference page in Help browser
  doc bar

>> doc bar
fx >>
```

Name	Value	Min	Max
a	27	27	27
ans	27	27	27
b	[1,3,5,7,9]	1	9
c	[1,2,3;6,9,12;33,...]	1	99

```
dMap('data_uniform_2500.mat',1.0,10)
close all
clear all
bar(evals)
%-- 7/31/13 10:13 PM --%
7/31/13 10:38 PM --%
10+17
clc
10+17
clc
a=27
b=[1 3 5 7 9]
c=[1 2 3; 6 9 12; 33 66 99]
disp
disp(b)
bar(b)
clc
bar(b)
help
help disp
clc
help bar
```

■ The MATLAB documentation is excellent

Getting help



matlab bar



Web

Images

Maps

Shopping

Videos

More ▾

Search tools

About 5,680,000 results (0.16 seconds)

[Bar graph - MATLAB bar - MathWorks](#)

www.mathworks.com/help/matlab/ref/bar.html ▾

This **MATLAB** function draws one **bar** for each element in Y.

[Bar chart - MATLAB bar, barh - MathWorks](#)

www.mathworks.com/help/finance/barbarh.html ▾

tsobj. Financial time series object. width. Width of the **bars** and separation of **bars** within a group. (Default = 0.8.) If width is 1, the **bars** within a group touch one ...

[Bar and Area Graphs - MATLAB & Simulink - MathWorks](#)

www.mathworks.com/help/matlab/creating.../bar-and-area-graphs.html ▾

View results over time, comparing results, and displaying individual contribution to a total amount.

[MATLAB Plot Gallery - Vertical Bar Plot: Vertical_Bar_Plot - MathW...](#)

www.mathworks.com/matlabcentral/...matlab...bar.../Vertical_Bar_Plot.ht... ▾

This is an example of how to create a vertical **bar** chart in **MATLAB**®. Read about the **bar** function in the **MATLAB**® documentation. Go to **MATLAB** Plot Gallery

[MATLAB Plot Gallery - Stacked Bar Chart: Stacked_Bar_Chart - Fil...](#)

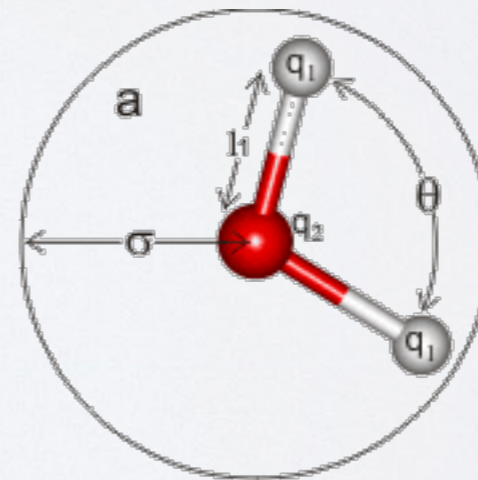
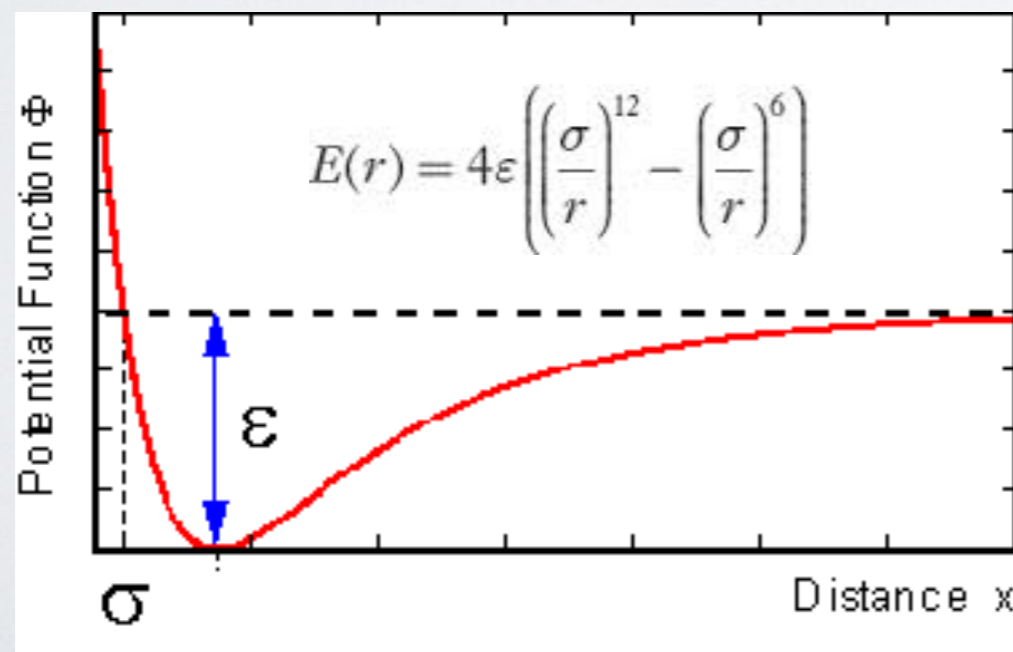
www.mathworks.com/matlabcentral/...matlab...bar.../Stacked_Bar_Chart.... ▾

This is an example of how to create a stacked **bar** chart in **MATLAB**®. Read about the **bar** function in the **MATLAB**® documentation. Go to **MATLAB** Plot Gallery

III. Data Visualization

Data Visualization

- Let's together run through a number of common (but powerful) data visualization and exploration techniques
- To make things concrete, we shall analyze the σ and ϵ parameters for a number of water models used in molecular dynamics simulations



Load data

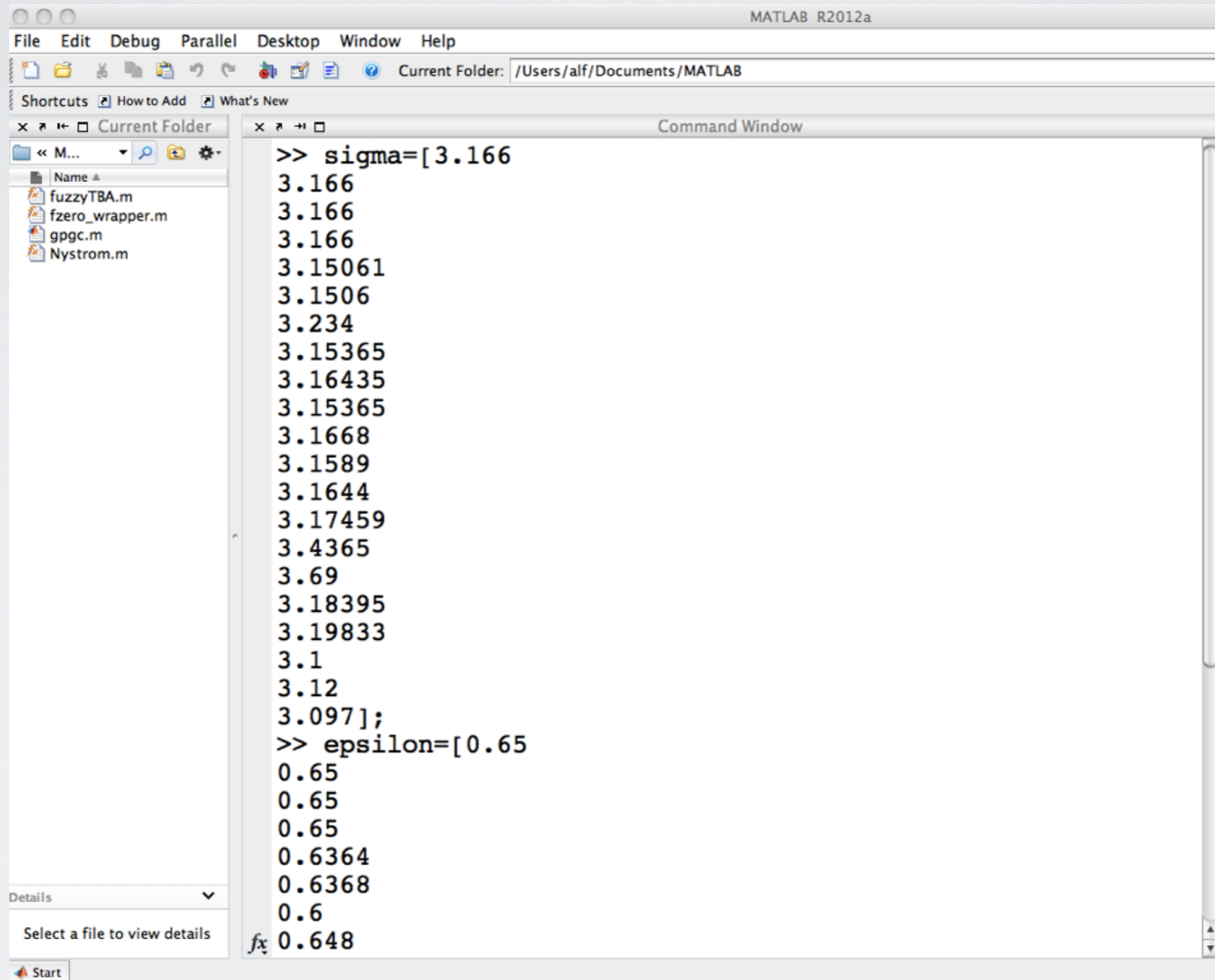
- Copy the file water_models.csv from /class/mse404pla
- csv = comma separated values
- Can open in Excel:

<http://www.lsbu.ac.uk/water/models.html>

Water Model	sigma / Ang	eps / kj/mol
SPC	3.166	0.65
SPC/E	3.166	0.65
SPC/HW (D2O)	3.166	0.65
SPC/Fw	3.166	0.65
TIP3P	3.15061	0.6364
TIP3P/Fw	3.1506	0.6368
PPC	3.234	0.6
TIP4P	3.15365	0.648
TIP4P-Ew	3.16435	0.680946
TIP4P-FQ	3.15365	0.648
TIP4P/Ice	3.1668	0.8822
TIP4P/2005	3.1589	0.7749
TIP4P/2005f	3.1644	0.7749
COS/G3	3.17459	0.9445
COS/D	3.4365	0.5119
GCPM	3.69	0.9146
SWM4-NDP	3.18395	0.88257
SWM6	3.19833	0.67781
ST2	3.1	0.31694
TIP5P	3.12	0.6694
TIP5P-Ew	3.097	0.7448

Load data

(i) Copy and paste



The image shows a screenshot of the MATLAB R2012a Command Window. The window title is "MATLAB R2012a" and the menu bar includes "File", "Edit", "Debug", "Parallel", "Desktop", "Window", and "Help". The current folder is "/Users/alf/Documents/MATLAB". The Command Window displays the following code and output:

```
>> sigma=[3.166
3.166
3.166
3.15061
3.1506
3.234
3.15365
3.16435
3.15365
3.1668
3.1589
3.1644
3.17459
3.4365
3.69
3.18395
3.19833
3.1
3.12
3.097];
>> epsilon=[0.65
0.65
0.65
0.6364
0.6368
0.6
fx 0.648
```

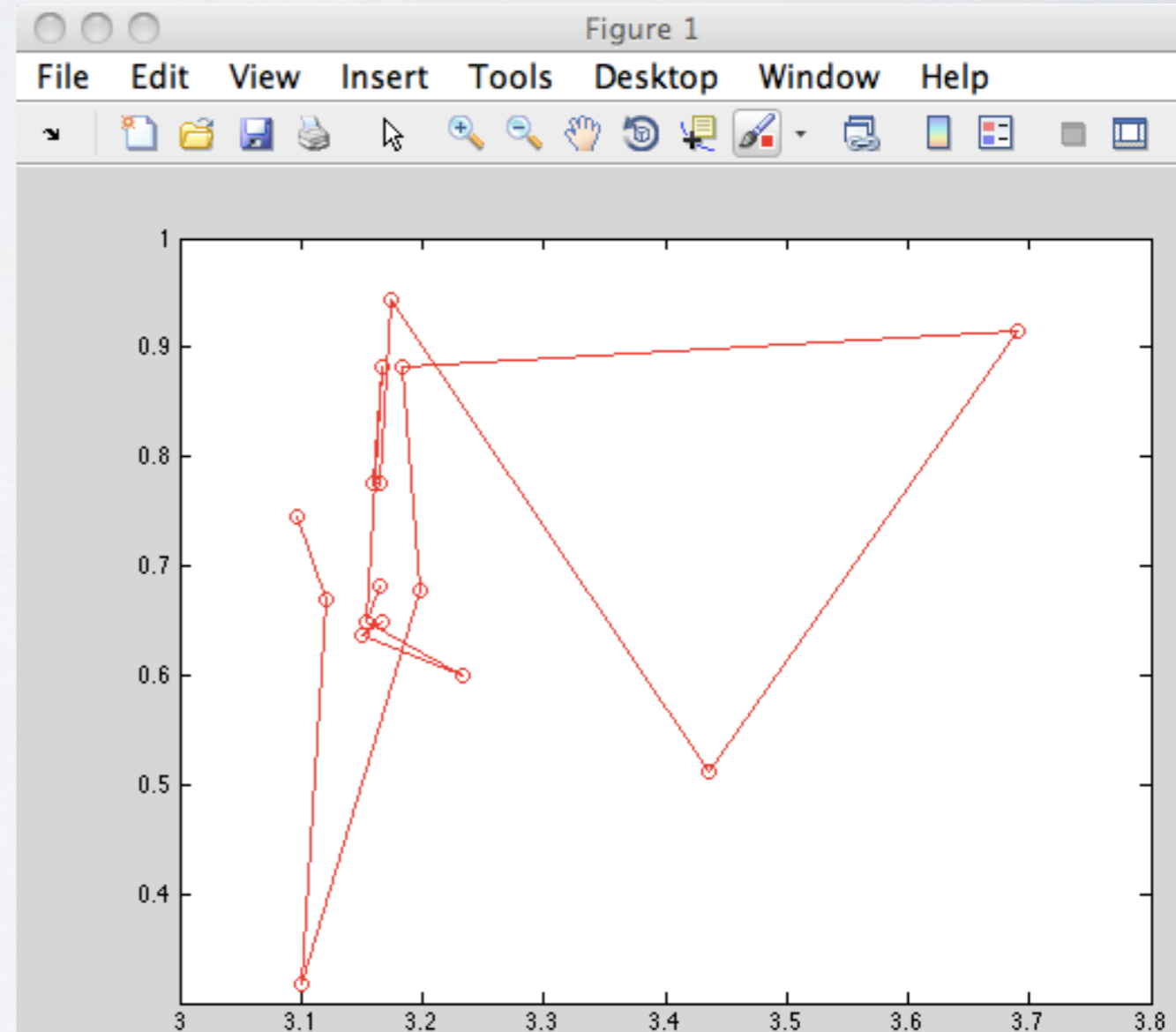
Load data

- (ii) textscan

```
>> fid=fopen('water_models.csv','rt');  
>> C=textscan(fid,'%*s %f  
%f','headerlines',3,'delimiter',' ');  
>> fclose(fid);  
>> sigma=C{1};  
>> epsilon=C{2};
```

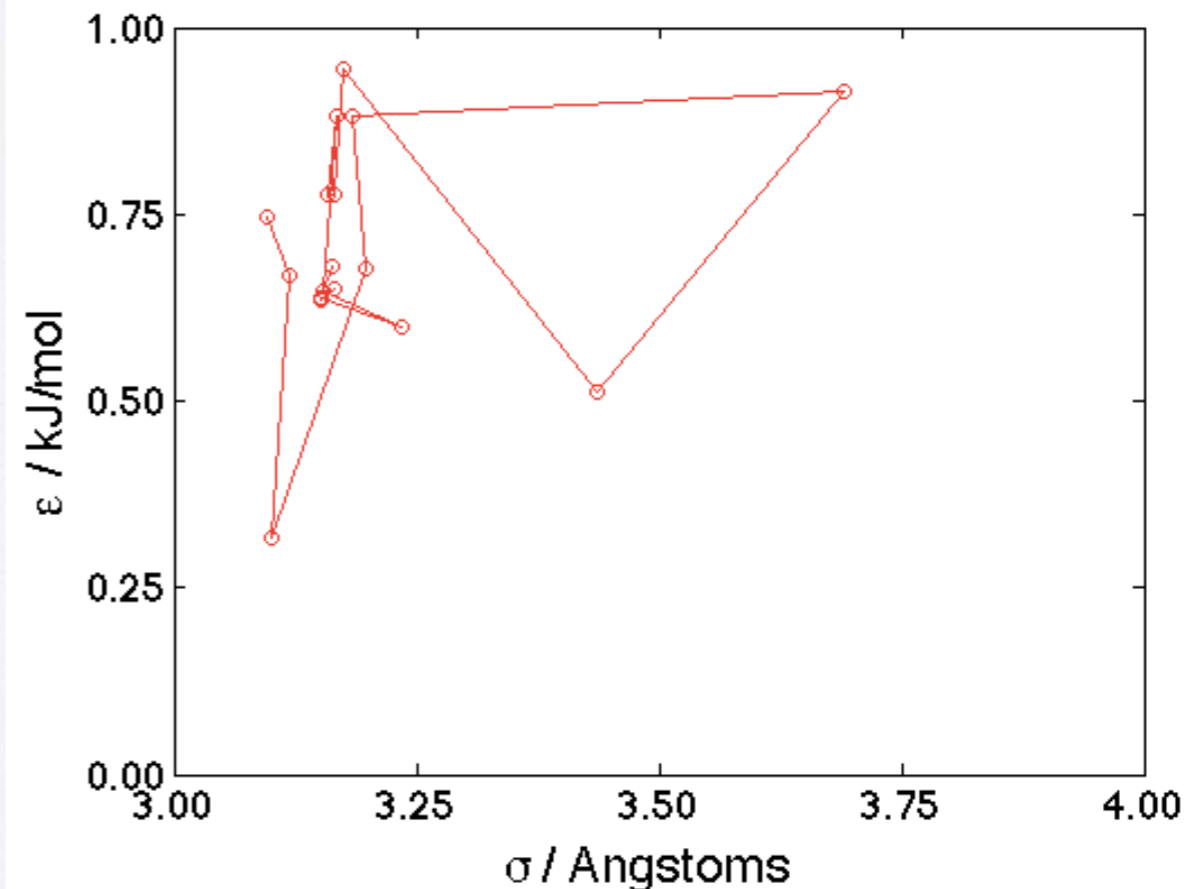
I. plot

```
>> scrsz = get(0, 'ScreenSize');  
>> figure('Position', [0 scrsz(4)/2 scrsz(3)/2 scrsz(4)/2])  
>> plot(sigma, epsilon, 'ro-')  
  
>> saveas(gcf, 'myFigure', 'fig')  
>> saveas(gcf, 'myFigure', 'jpg')
```



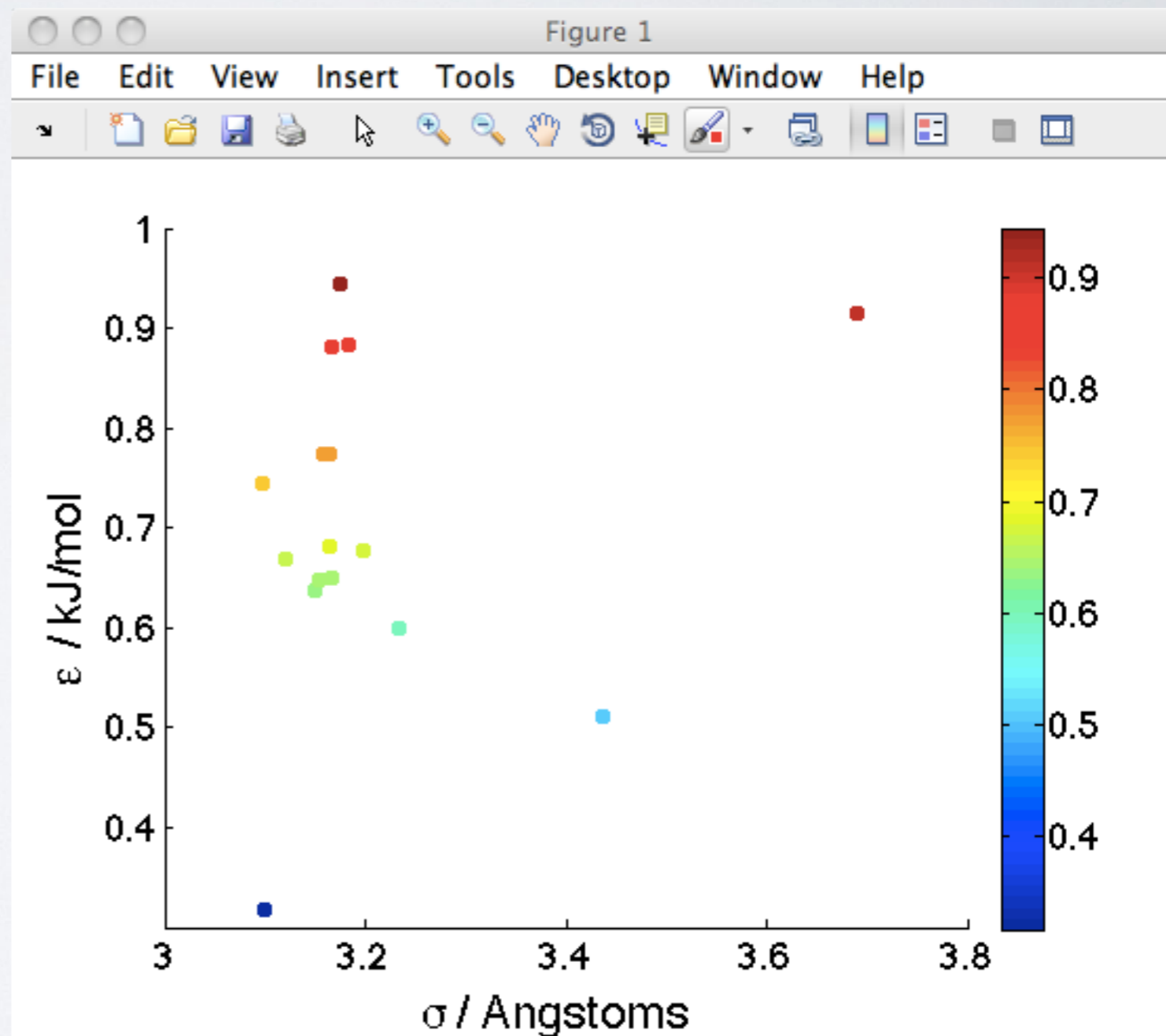
I. plot

```
>> set(gca,'fontsize',18)
>> set(gcf,'color','w')
>> ylabel('\epsilon / kJ/mol','fontsize',22)
>> xlabel('\sigma / Angstroms','fontsize',22)
>> xlim([3 4])
>> set(gca,'xtick',3:0.25:4)
>> set(gca,'xticklabel',{'3.00','3.25','3.50','3.75','4.00'})
>> ylim([0 1])
>> set(gca,'ytick',0:0.25:1)
>> set(gca,'yticklabel',{'0.00','0.25','0.50','0.75','1.00'})
```



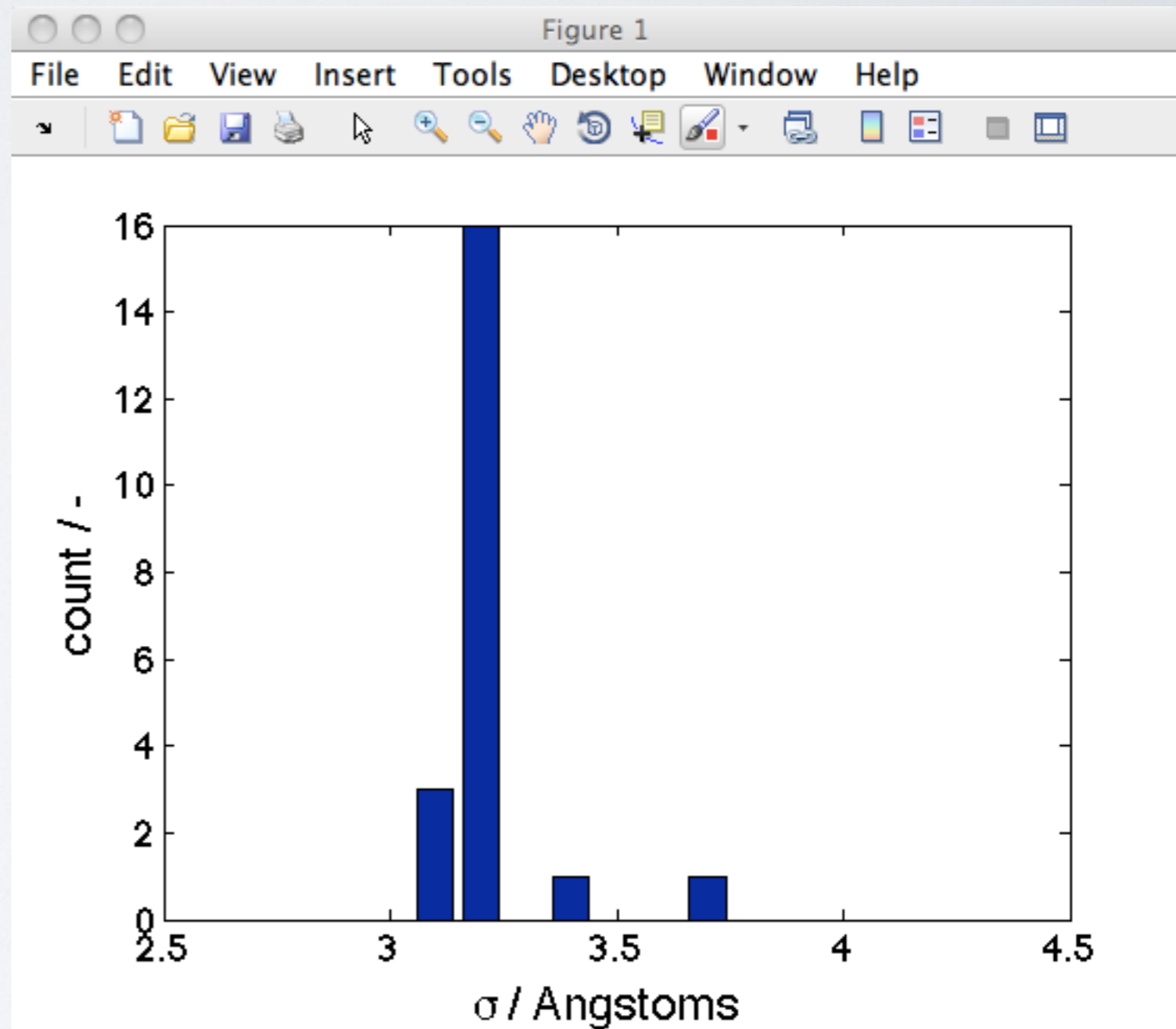
2. scatter

```
>> scatter(sigma,epsilon,55,epsilon,'filled')
>> colorbar
>> set(gca,'fontsize',18)
>> xlabel('\sigma / Angstroms','fontsize',22)
>> ylabel('\epsilon / kJ/mol','fontsize',22)
>> set(gcf,'color','w')
```



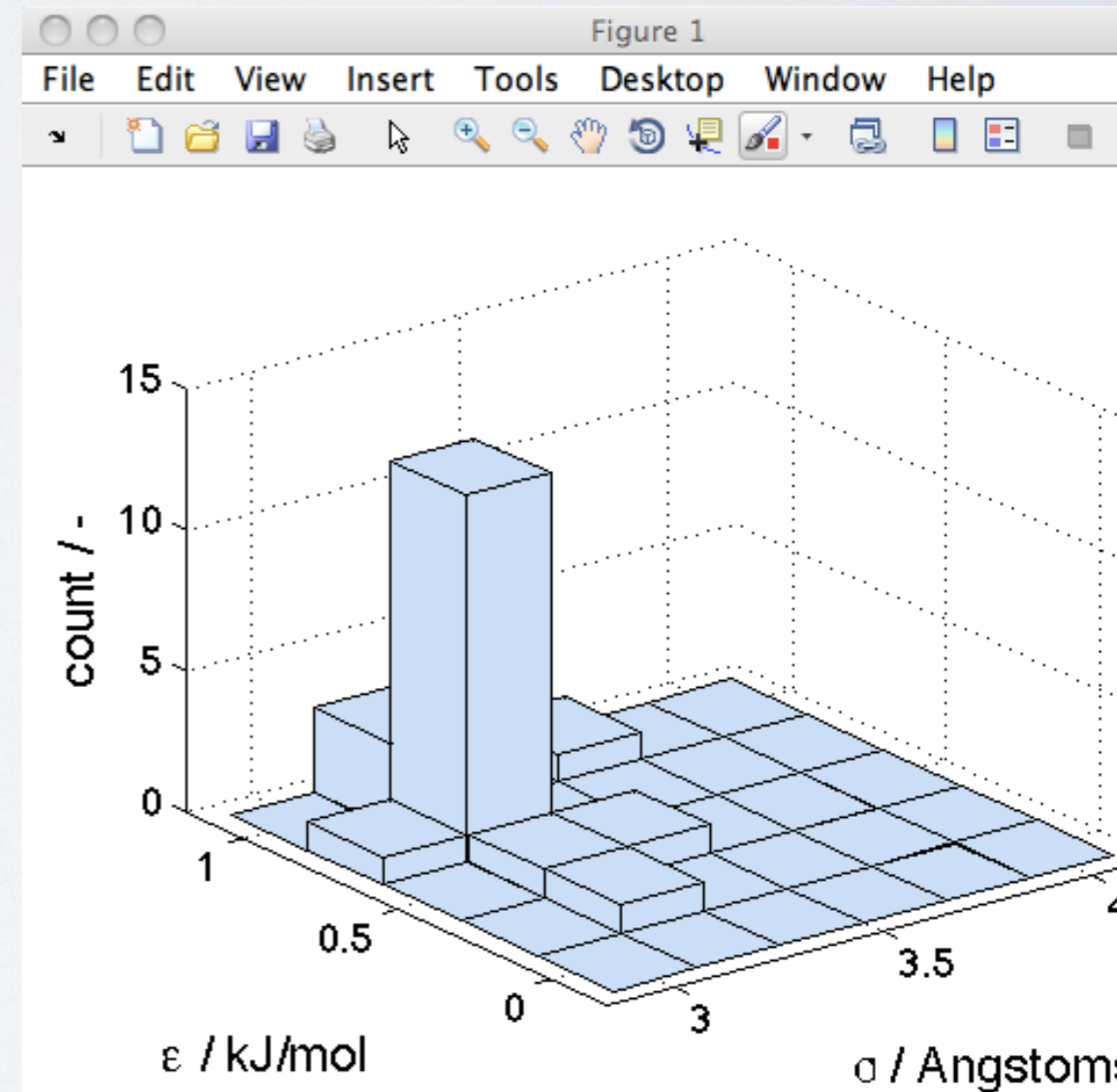
3. hist

```
>> [count,bins]=hist(sigma,3:0.1:4);  
>> bar(bins,count)  
>> xlabel('\sigma / Angstroms','fontsize',22)  
>> ylabel('count / -','fontsize',22)  
>> set(gcf,'color','w')  
>> set(gca,'fontsize',18)
```



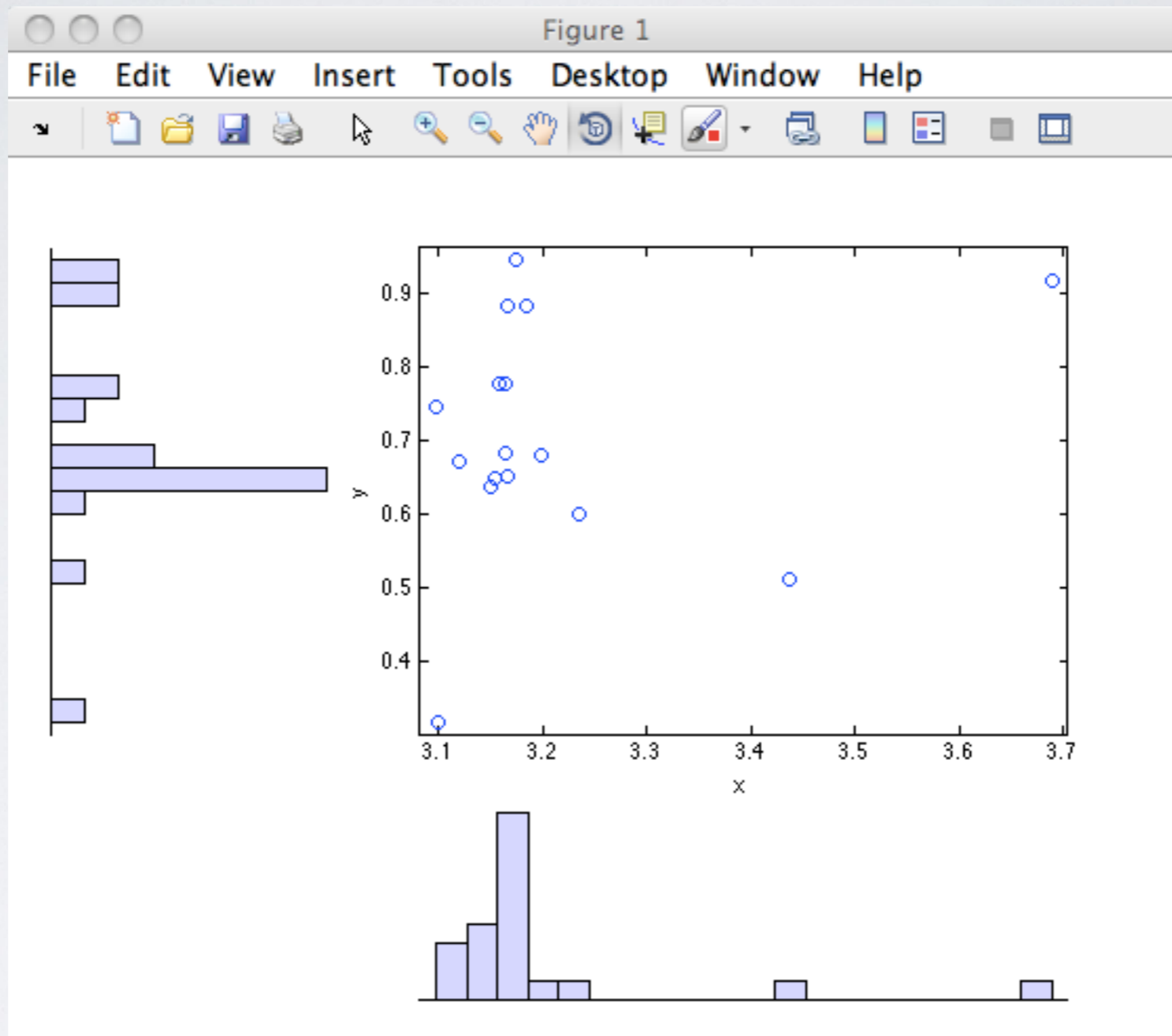
4. hist3

```
>> data=cat(2,sigma,epsilon);  
  
>> bins_sigma=3:0.2:4;  
>> bins_epsilon=0:0.25:1;  
>> bins=cell(2,1);  
>> bins{1}=bins_sigma; bins{2}=bins_epsilon;  
  
>> hist3(data,bins)  
>> xlabel('\sigma / Angstroms',...  
    'fontsize',22)  
>> ylabel('\epsilon / kJ/mol',...  
    'fontsize',22)  
>> zlabel('count / -',...  
    'fontsize',22)  
>> set(gca,'fontsize',18)  
>> set(gcf,'color','w')
```



5. scatterhist

```
>> scatterhist(sigma,epsilon,'NBins',[20,20])
```



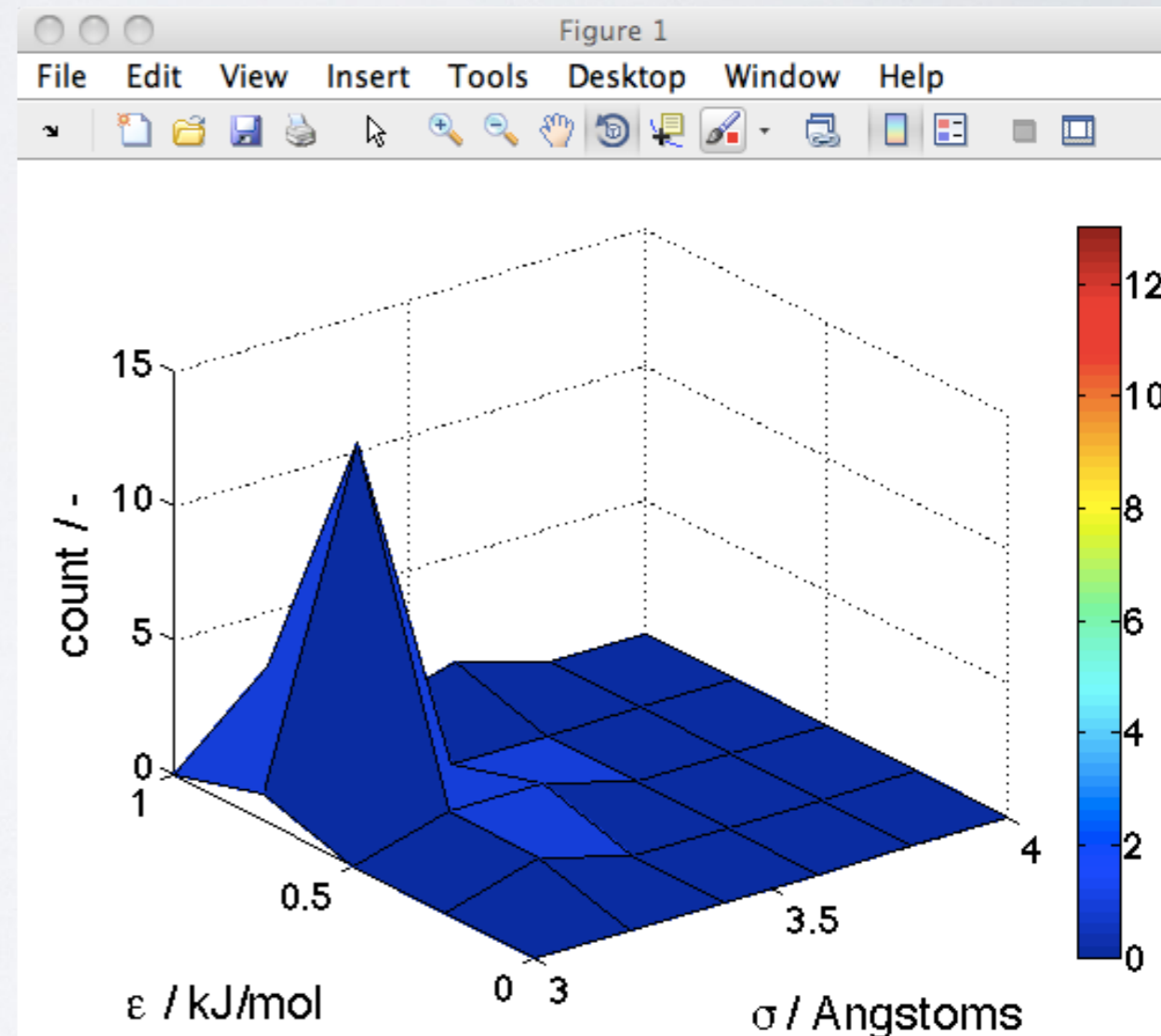
6. surf

```
>> [count,bins] = hist3(data,bins)

>> bins_X = bins{1};
>> bins_Y = bins{2};
>> [X,Y]=meshgrid(bins{1},bins{2})

>> surf(X,Y,count')
>> colorbar
>> xlabel('\sigma / Angstroms',...
'fontsize',22)
>> ylabel('\epsilon / kJ/mol',...
'fontsize',22)
>> zlabel('count / -',...
'fontsize',22)
>> set(gca,'fontsize',18)
>> set(gcf,'color','w')
```

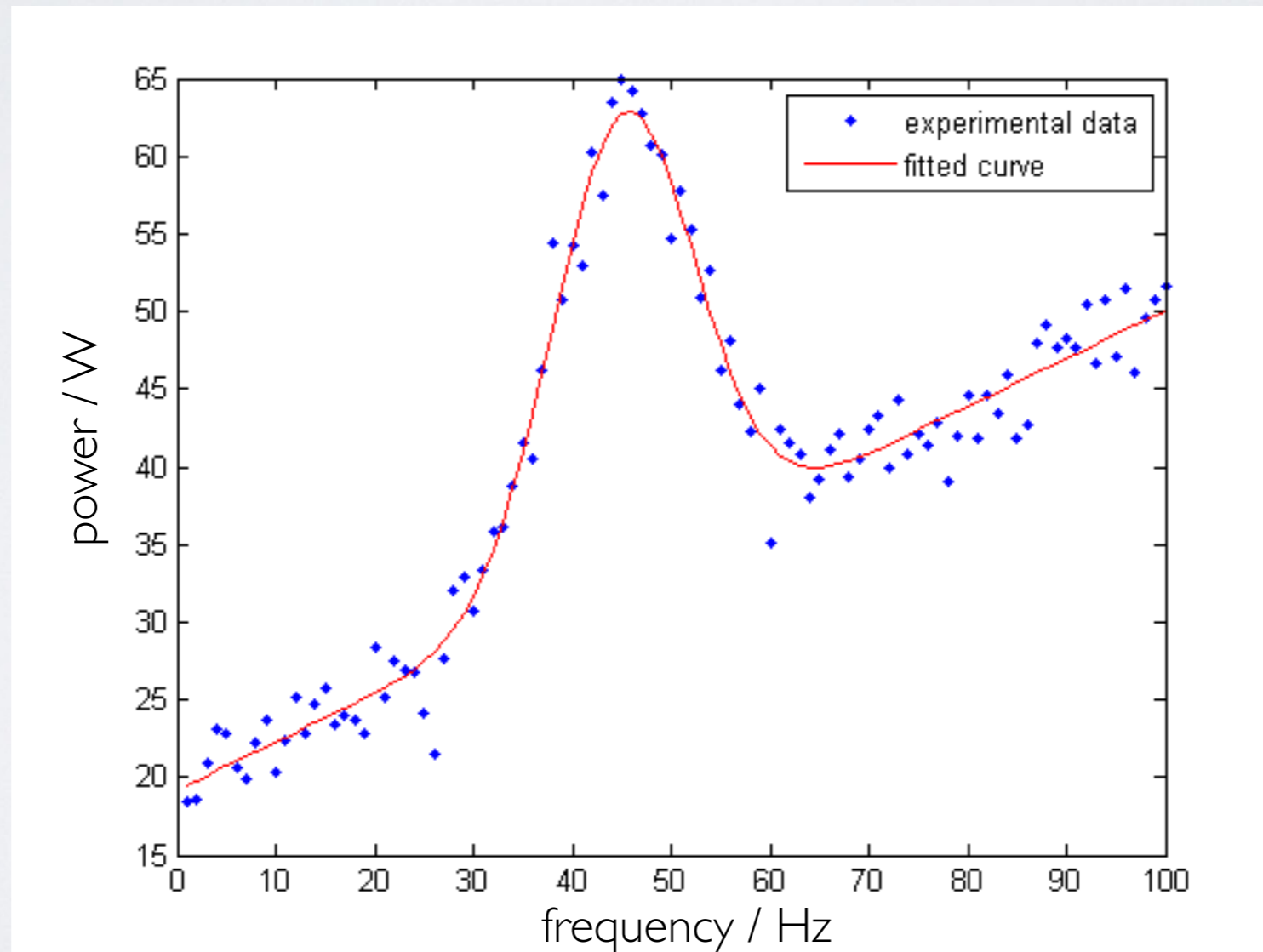
What happens when you replace surf with mesh / meshc?



IV. Data Analysis

Data Analysis

- Now let's consider a number of useful data analysis tools and statistical tests



Null hypothesis

null hy·poth·e·sis

noun

(in a statistical test) the hypothesis that there is no significant difference between specified populations, any observed difference being due to sampling or experimental error.

- By default, we assume that the null hypothesis is true
- We apply statistical tests to assess whether there is sufficient evidence to reject the null hypothesis
- We reject the null hypothesis if the observed relationship in the data is sufficiently unlikely to have arisen by chance if the null hypothesis were true ($p < \alpha = 0.05, 0.01$)

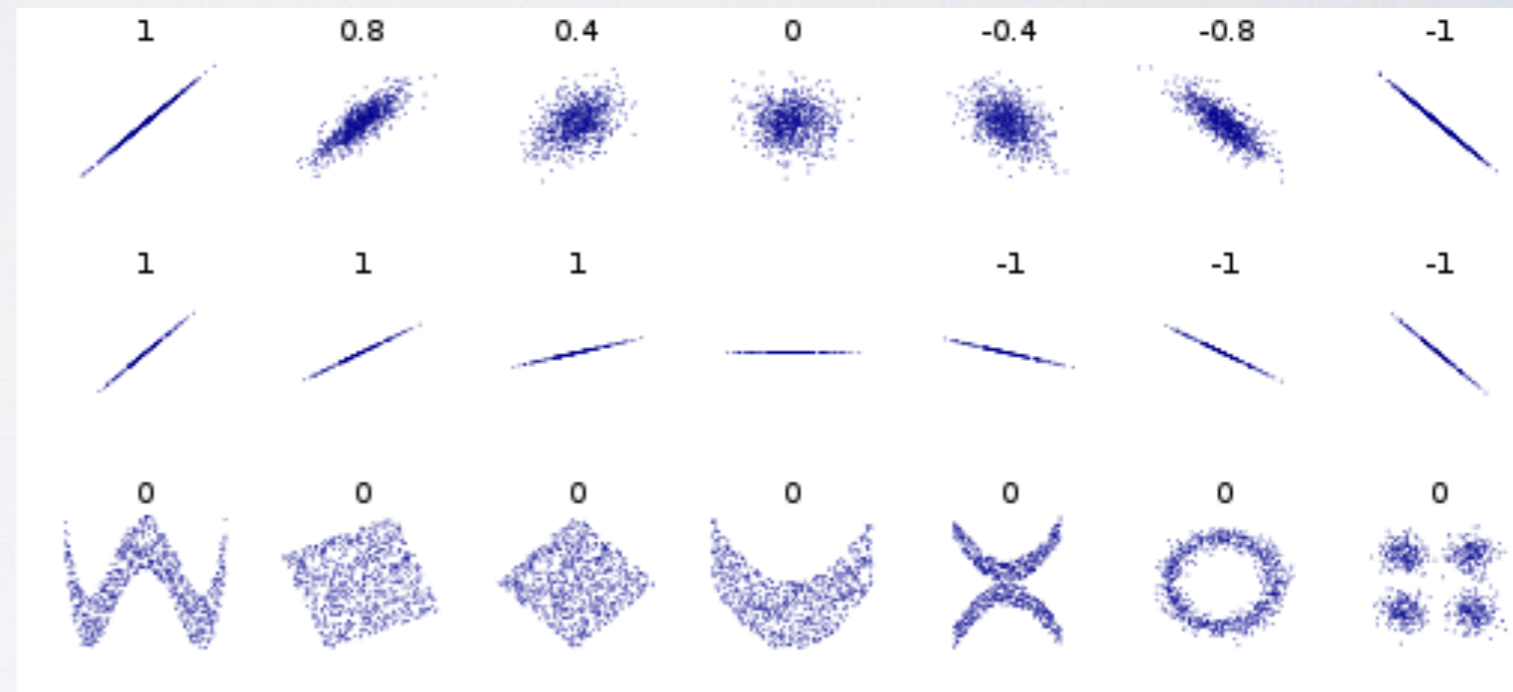
I. Pearson's correlation coefficient (r)

■ Purpose

Measure of the **linear** correlation between two variables.
Limited to range $[-1, 1]$.

■ Theory

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}$$



- Fails to uncover **nonlinear** relationships.
- Use Spearman corr coeff for **rank** correlation (monotonicity)

I. Pearson's correlation coefficient

Practice

```
>> [r,p,r_lower,r_upper]=corrcoef(sigma,epsilon)
```

```
r =
```

```
1.0000    0.2441  
0.2441    1.0000
```



pairwise Pearson corr coeff

```
p =
```

```
1.0000    0.2862  
0.2862    1.0000
```



p-value from Student's t-test

```
r_lower =
```

```
1.0000   -0.2096  
-0.2096    1.0000
```



95% confidence interval

```
r_upper =
```

```
1.0000    0.6114  
0.6114    1.0000
```

If x & y are uncorrelated Gaussian distributions, Pearson's r follows a Student's t -distribution with $(n-2)$ dof.

Using this distribution, we ask: “*what is the probability that the observed Pearson's r value arose by chance given that the true correlation is zero?*”

95% CI on Pearson's r or “*what is the expected range of r given our finite data sample?*”

2. Permutation Test

■ Purpose

A non-parametric hypothesis test.

Without assuming a distribution, answers: *“What is the probability the observed result occurred by chance?”*

■ Theory

We perform random shuffles of the data and compute the test statistic.

The p-value is the proportion of shuffled test statistics that are greater than the observed value.

2. Permutation Test

Practice

```
>> R_obs=corrcoef(sigma,epsilon);  
r_obs = R_obs(1,2)  
  
r_obs =  
  
    0.2441  
  
>> n=length(sigma);  
nTests=1000;  
r_array=zeros(nTests,1);  
  
for i=1:nTests  
    idx=randperm(n);  
    R = corrcoef(sigma,epsilon(idx));  
    r_array(i) = R(1,2);  
end  
  
p_value = sum(abs(r_array)>abs(r_obs))/nTests  
  
p_value =  
  
    0.2630
```

3. Bootstrap

■ Purpose

Non-parametric estimate of test statistic confidence interval

Without assuming a distribution, answers: “*Given our finite sample, what range of test statistics might we have seen?*”

■ Theory

Our data typically represents a finite sample from a large population (e.g., human heights, component lifetimes, etc.)

Different samples of n data points produce different results

Bootstrap simulates different samples by resampling with replacement

3. Bootstrap

Practice

```
>> n = length(sigma);
nTests = 1000;
replacement=true;
r_array = zeros(nTests,1);

for i=1:nTests
idx = randsample(n,n,replacement);
R = corrcoef(sigma(idx),epsilon(idx));
r_array(i) = R(1,2);
end

[count,bins] = hist(r_array,100);
prob = count/sum(count);
cumProb = cumsum(prob);

alpha=0.05;
[~,idx_lo] = find(cumProb<=alpha,1,'last');
CI_lo = bins(idx_lo);
[~,idx_hi] = find(cumProb>=(1-alpha),1,'first');
CI_hi = bins(idx_hi);

fprintf('\n\n');
fprintf('RESULT: %.0f%% CI = [%.2f,%.2f]\n',(1-2*alpha)*100,CI_lo,CI_hi);

RESULT: 90% CI = [-0.48,0.58]
>>
```

4. Multiple Linear Regression

■ Purpose

Attempt to recover predictor of a scalar dependent variable, y , as a linear combination of independent variables, \underline{x}

■ Theory

MLR model: $y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_m x_{im} + \epsilon_i = \vec{x}_i^T \vec{\beta} + \epsilon_i$
 $\vec{y} = \mathbf{X}\vec{\beta} + \vec{\epsilon}$

OLS estimate: $\vec{\beta}^* = \underset{\vec{\beta}}{\operatorname{argmin}} \|\vec{\epsilon}\| = \underset{\vec{\beta}}{\operatorname{argmin}} \|\vec{y} - \mathbf{X}\vec{\beta}\|$

Assumptions:

linear, independent x , homoscedastic, no multicollinearity

4. Multiple Linear Regression

Practice

```
>> y=sigma;  
>> X=[epsilon ones(length(epsilon),1)];  
>> [b,bint,r,rint,stats] = regress(y,X)
```

b =

```
0.2259  
3.0417
```

← β

bint =

```
-0.2050 0.6569  
2.7373 3.3460
```

← β 95% CI

r =

```
-0.0225  
-0.0225  
-0.0225  
-0.0225  
-0.0348  
-0.0349  
0.0568  
-0.0344  
-0.0312  
-0.0344  
-0.0742  
-0.0579  
-0.0524  
-0.0805  
0.2792  
0.4417  
-0.0571  
0.0035  
-0.0133  
-0.0729  
-0.1130
```

← ϵ

rint =

```
-0.2955 0.2505  
-0.2955 0.2505  
-0.2955 0.2505  
-0.2955 0.2505  
-0.3071 0.2374  
-0.3072 0.2373  
-0.2126 0.3261  
-0.3071 0.2382  
-0.3046 0.2422  
-0.3071 0.2382  
-0.3323 0.1839  
-0.3278 0.2121  
-0.3226 0.2179  
-0.3275 0.1665  
0.0564 0.5020  
0.3080 0.5754  
-0.3163 0.2020  
-0.2703 0.2773  
-0.2308 0.2042  
-0.3442 0.1984  
-0.3801 0.1542
```

← ϵ 95% CI

stats =

```
0.0596 1.2042 0.2862 0.0170
```

↑
 $R^2 = 1 - SSE/TSS$

↑
F-stat

↑
p-value

↑
 $s^2 = \sqrt{RMSE}$
(error variance)

4. Multiple Linear Regression

The F-test assesses whether the fitted regression model gives a **statistically significant** better fit to the data than simply describing the data by its mean.

Model 1: Mean*

$$\vec{y} = \vec{\epsilon}$$

params = 1
dof = n-1

Model 2: Regression*

$$\vec{y} = \mathbf{X}\vec{\beta} + \vec{\epsilon}$$

params = k
dof = n-k

$$F = \frac{\frac{RSS_1 - RSS_2}{dof_2 - dof_1}}{\frac{RSS_2}{n - dof_2}}$$

follows an F-distribution $F(dof_2 - dof_1, n - dof_2)$ under null hypothesis that Model 2 is not better than Model 1

p-value = probability of observing this large (or larger) F-value by chance if the null hypothesis is true
assert significance (yes/no) by specifying a significance cutoff alpha (usually alpha = 0.05)

*Model 1 must be a restriction / specialization of Model 2

ASIDE: Regression or correlation?

REGRESSION OR CORRELATION?

Linear regression and correlation are similar and easily confused. In some situations it makes sense to perform both calculations. Calculate linear correlation if you measured both X and Y in each subject and wish to quantify how well they are associated. Select the Pearson (parametric) correlation coefficient if you can assume that both X and Y are sampled from Gaussian populations. Otherwise choose the Spearman nonparametric correlation coefficient. Don't calculate the correlation coefficient (or its confidence interval) if you manipulated the X variable.

Calculate linear regressions only if one of the variables (X) is likely to precede or cause the other variable (Y). Definitely choose linear regression if you manipulated the X variable. It makes a big difference which variable is called X and which is called Y, as linear regression calculations are not symmetrical with respect to X and Y. If you swap the two variables, you will obtain a different regression line. In contrast, linear correlation calculations are symmetrical with respect to X and Y. If you swap the labels X and Y, you will still get the same correlation coefficient.

5. Akaike Information Criterion (AIC)

■ Purpose

Model discrimination criterion, “*what model should I choose?*”
Trade-off between goodness-of-fit and model complexity

“With four parameters I can fit an elephant, and with five I can make him wiggle his trunk”

- *John von Neumann*

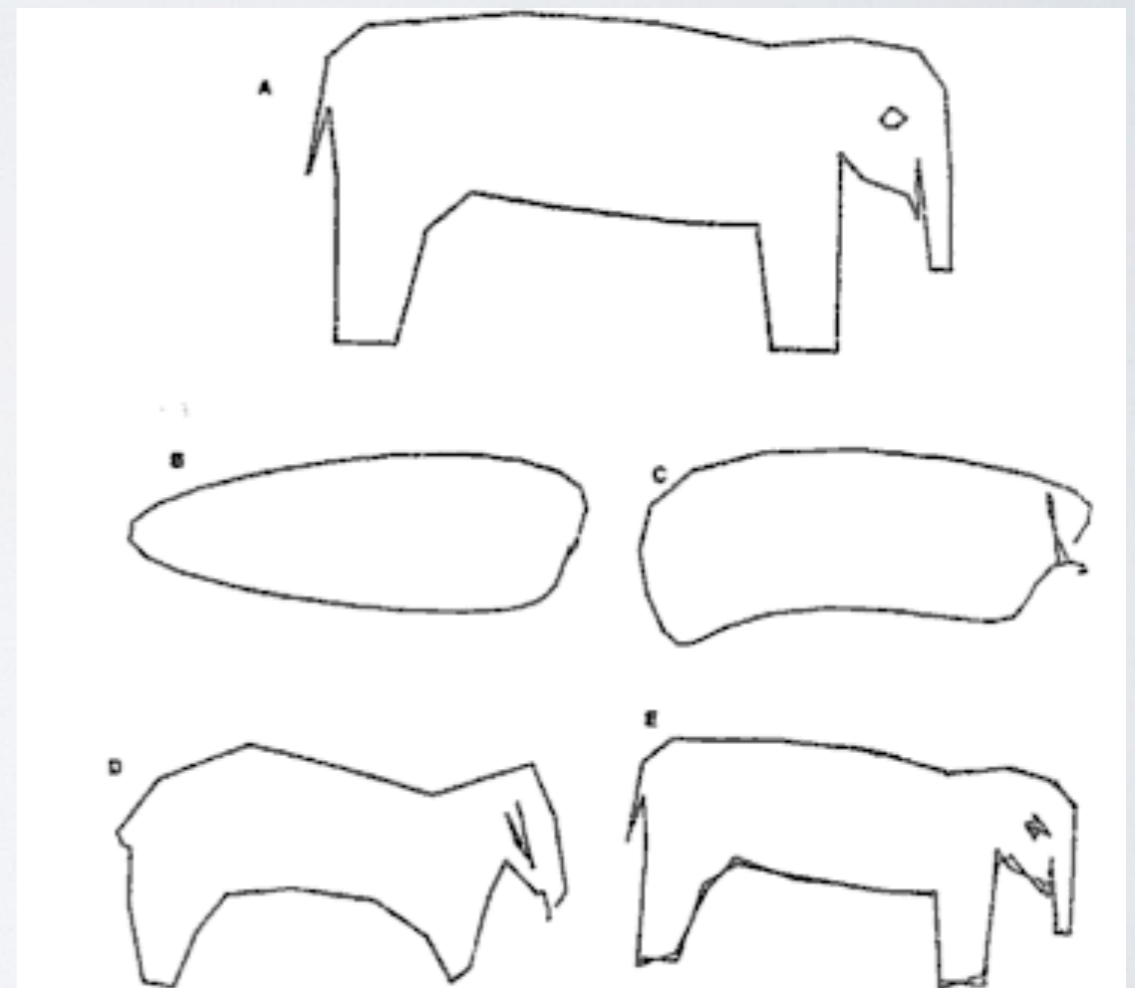


FIGURE 1.2. “How many parameters does it take to fit an elephant?” was answered by Wei (1975). He started with an idealized drawing (A) defined by 36 points and used least squares Fourier sine series fits of the form $x(t) = \alpha_0 + \sum \alpha_i \sin(it\pi/36)$ and $y(t) = \beta_0 + \sum \beta_i \sin(it\pi/36)$ for $i = 1, \dots, N$. He examined fits for $K = 5, 10, 20,$ and 30 (shown in B–E) and stopped with the fit of a 30-term model. He concluded that the 30-term model “may not satisfy the third-grade art teacher, but would carry most chemical engineers into preliminary design.”

5. Akaike Information Criterion (AIC)

■ Theory

Information theoretical measure

Estimate of information loss relative to “true” model

Penalizes more parameters and poor fits

$$AIC = 2k - 2\ln(L)$$

$k = \#$ model parameters
 $L =$ likelihood of model given data

For i.i.d. normally distributed errors

$$AIC = 2k + n\ln\left(\frac{RSS}{n}\right)$$

$n = \#$ data points
 $RSS =$ residual sum of squares

Compute AIC for various models and choose $\min(AIC)$

5. Akaike Information Criterion (AIC)

Practice

Use AIC to discriminate between regression models:

$$(i) \quad \sigma = \beta_1 \varepsilon + c$$

$$(ii) \quad \sigma = \beta_2 \varepsilon^2 + \beta_1 \varepsilon + c$$

$$(iii) \quad \sigma = \beta_3 \varepsilon^3 + \beta_2 \varepsilon^2 + \beta_1 \varepsilon + c$$

```
>> y=sigma;
>> X1=[epsilon,ones(length(epsilon),1)];
>> X2=[epsilon.^2,epsilon,ones(length(epsilon),1)];
>> X3=[epsilon.^3,epsilon.^2,epsilon,ones(length(epsilon),1)];
>> [b1,bint1,r1,rint1,stats1] = regress(y,X1);
>> [b2,bint2,r2,rint2,stats2] = regress(y,X2);
>> [b3,bint3,r3,rint3,stats3] = regress(y,X3);
>>
>> n=length(sigma);
>> k=[2 3 4];
>> RSS=[r1'*r1 r2'*r2 r3'*r3];
>> for i=1:3
AIC(i) = 2*k(i) + n*log(RSS(i)/n);
end
>> AIC

AIC =

-83.6249 -82.6224 -85.9937
```

5. Akaike Information Criterion (AIC)

But beware!

AIC measures only **relative**, not **absolute** model quality!

```
>> r2 = [stats1(1) stats2(1) stats3(1)]  
r2 =  
    0.0596    0.1032    0.3056
```

6. Cross Validation

■ Purpose

Empirical assessment of model performance on “new” data
Alternative to AIC for model discrimination
Quantitative assessment of model over-fitting

■ Theory

MSE measured over training data over optimistic prediction of performance on new data - *“in-sample MSE”*

Break data into training and validation sets, the *CV-MSE* or *“out-of-sample MSE”* better measure of model performance

Common splits: **k-fold CV**

leave-one-out CV (LOO-CV)

6. Cross Validation

Practice

Use LOO-CV to discriminate between regression models:

- (i) $\sigma = \beta_1 \varepsilon + c$
- (ii) $\sigma = \beta_2 \varepsilon^2 + \beta_1 \varepsilon + c$
- (iii) $\sigma = \beta_3 \varepsilon^3 + \beta_2 \varepsilon^2 + \beta_1 \varepsilon + c$

	<u>MSE</u>	<u>LOOCV-MSE</u>
(i)	0.0154	0.0213
(ii)	0.0147	0.0731
(iii)	0.0114	0.147

```
>> n=length(sigma);
y=sigma;

% standard OLSR
X=[epsilon,ones(length(epsilon),1)];
[b,~,r,~,stats]=regress(y,X);
MSE=mean(r.^2);

% LOO-CV
CV_MSE_array=nan(n,1);

for p=1:n % running over LOO

    % building LOO independent variables
    idx=setdiff(1:n,p);
    X=[epsilon(idx),ones(length(idx),1)];

    % performing regression
    [b,~,r,~,stats]=regress(y(idx),X);

    % computing CV_MSE
    X_solo=[epsilon(p),1];
    CV_MSE_array(p)=(y(p)-X_solo*b)^2;

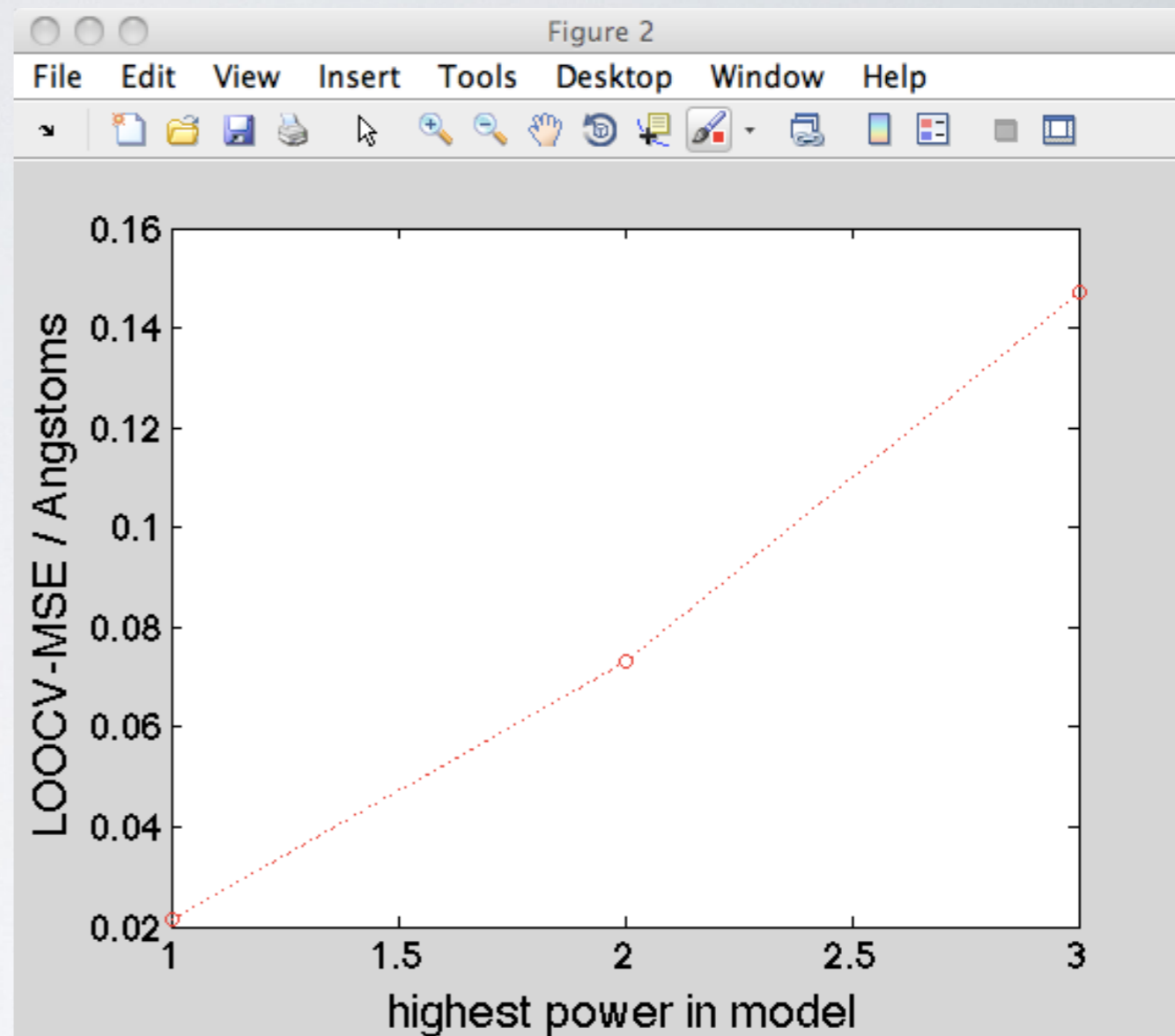
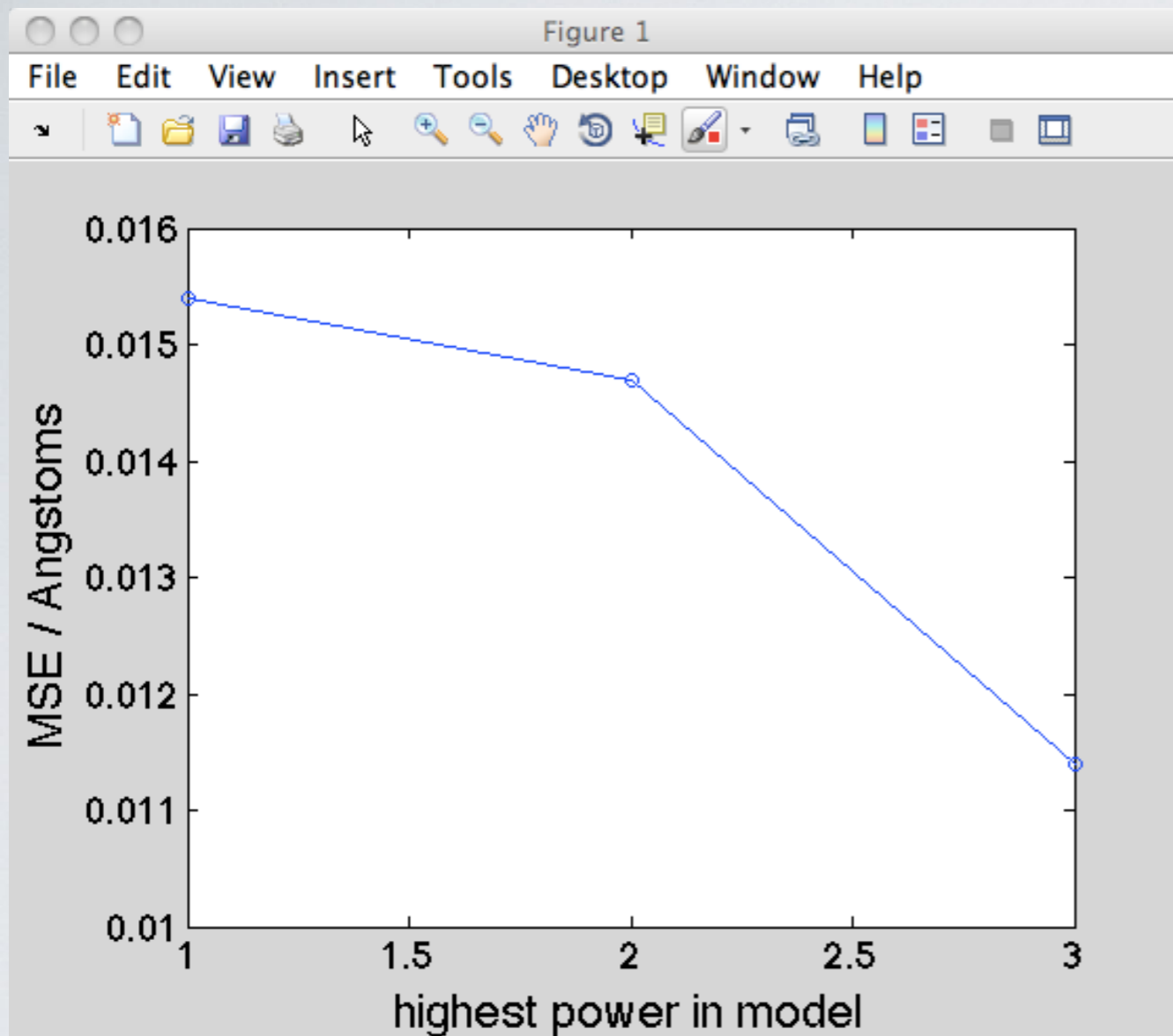
end

CV_MSE=mean(CV_MSE_array);

>> fprintf('MSE = %f, LOOCV-MSE = %f\n',MSE,CV_MSE)
MSE = 0.015412, LOOCV-MSE = 0.021340
```

(i)

6. Cross Validation



The models immediately begin to overfit with increasing complexity (indicative of a poor modeling paradigm)

Select model using **minimum** or **knee** in MSE and CV-MSE curves

7. Student's t-test

■ Purpose

Are two data sets significantly different?

or *Are two data sets drawn from same underlying dist'n?*

■ Theory

Assumes:

- two data sets are independent
- each set normally distributed if scaling term were known
- small ($n < 30$) sample sizes

For large sample sizes, use **Z-test**

For non-normally distributed data use **Mann-Whitney**

7. Student's t-test

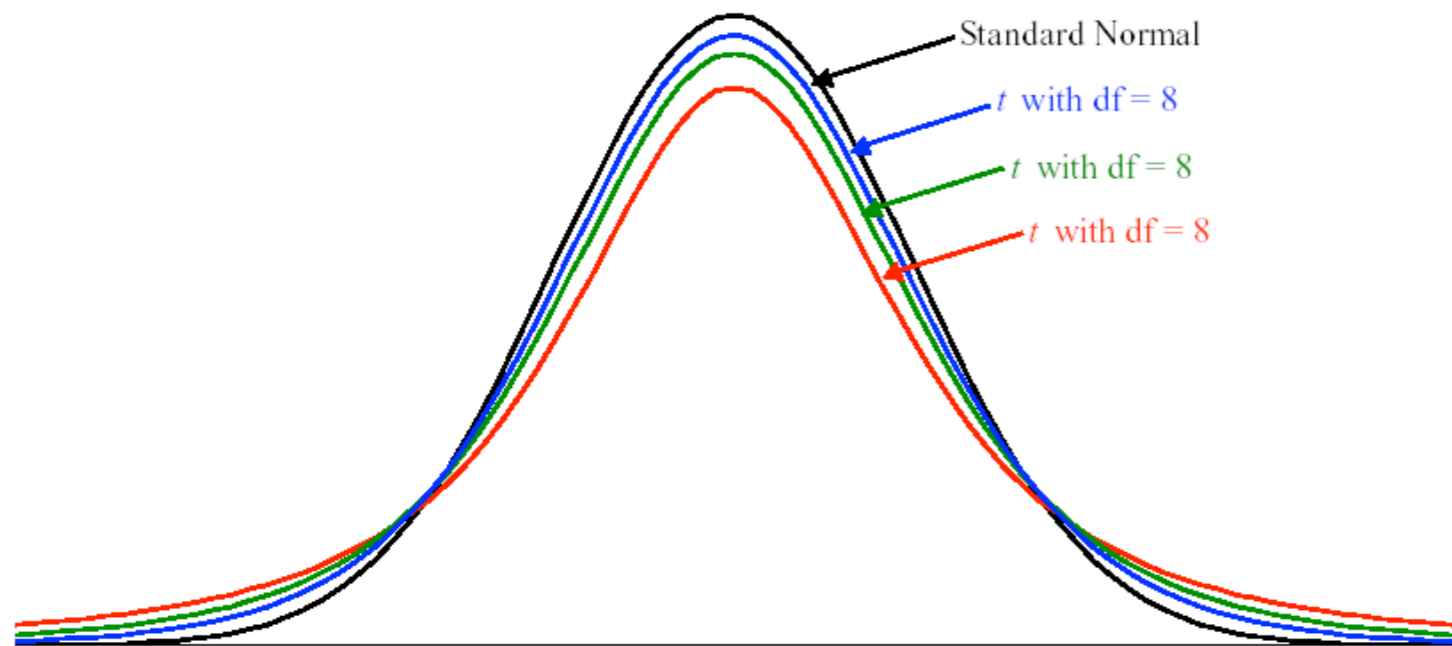
Theory

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

← test statistic

$$dof = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{\left(\frac{s_1^2}{n_1}\right)^2}{n_1 - 1} + \frac{\left(\frac{s_2^2}{n_2}\right)^2}{n_2 - 1}}$$

← degrees of freedom



← determine significance at level α from Student's t-distribution with dof

7. Student's t-test

Practice

Determine if TIP4P water model sigma parameters follow a different distribution from the rest of the models.

```
>> sigma_TIP4P = sigma(8:13);
>> sigma_rest = sigma(setdiff(1:length(sigma),8:13));
>> alpha = 0.05;
>> [h,p,ci,stats] = ttest2(sigma_TIP4P,sigma_rest,alpha,'both','unequal')

h =
    0

p =
    0.2043

ci =
   -0.1384
    0.0324

stats =
    tstat: -1.3311
         df: 14.0976
         sd: [2x1 double]

>> t_crit = tinv(alpha/2,stats.df)

t_crit =
   -2.1434
```

← split data & specify significance

← perform t-test

- ttest2 = 2 independent samples
- 'both' = means not equal
- 'unequal' = variances not equal

- h = 1 => reject null hypothesis
that same dist'n

= 0 => accept null hypothesis

- p = p-value under null
hypothesis of observed or
more extreme t-value

- ci = confidence interval at alpha

← critical t-value to reject null
hypothesis