MATLAB, part II

Simple data summaries – mean, variance, etc

Built into Matlab are functions for simple data analysis. They include, `mean`, `median`, `var`, `std` (standard deviation), `cov` (covariance matrix), `min`, `max`, etc.

The default for each of these functions is to work columnwise. For example

```
b =

     1     3     5
     2     4     6

>> mean(b)

ans =

    1.5000    3.5000    5.5000
```

These functions also have options to allow the summaries to be calculated for other dimensions. For example

```
>> mean(b,2)

ans =

     3
     4
```

Now for some functions you do need to be a bit careful as the second argument may not be the dimension indicator. One case is with `std`. The second argument is an indicator of whether a divisor of n-1 (0, the default) or n (anything else), should be used.

```
>> std(b)         %  n-1, cols

ans =

    0.7071    0.7071    0.7071

>> std(b,2)       %  n, cols

ans =

    0.5000    0.5000    0.5000

>> std(b,0,2)     % n-1, rows

ans =

     2
     2
```

Most of these functions can also be used on higher way arrays. However there are some exception, such as cov and var, only 1 or 2 dimensional objects can be used.

Missing data

For these functions, if there is any missing data (coded by NaN), the response will be NaN.

```
>> mean(furnace_num)

ans =

NaN  21.9667  38.5667  10.0384  10.8131
```

However many of these functions have a companion function that deletes out these NaN and calculates the summaries and the remaining values. If there is no missing data for a column, the returned answer is the same as the base function

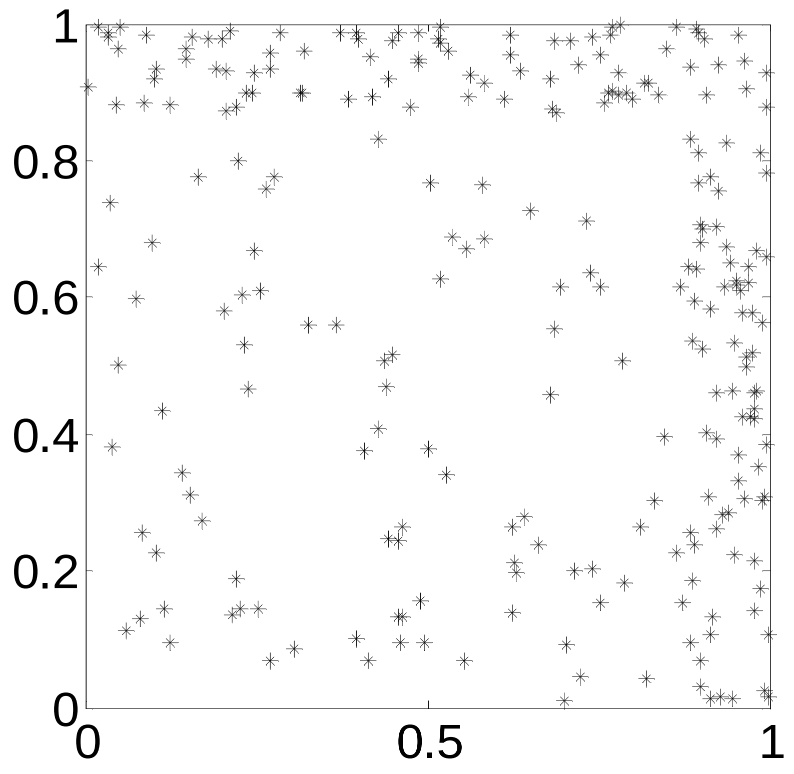
```
>> nanmean(furnace_num)

ans =

62.5618  21.9667  38.5667  10.0384
10.8131
```

`cov` doesn't have the nan version, which is probably the correct thing to do (at least if it acted like the other nan functions). If pairwise deletion (all complete pairs are used) is done, you can end up with a estimated covariance matrix which is not positive semidefinite. Listwise deletion (any NaNs in a row will remove observation from all calcs) does not have this problem. Also nanvar doesn't exist. Use `(nanstd(a)).^2` instead.

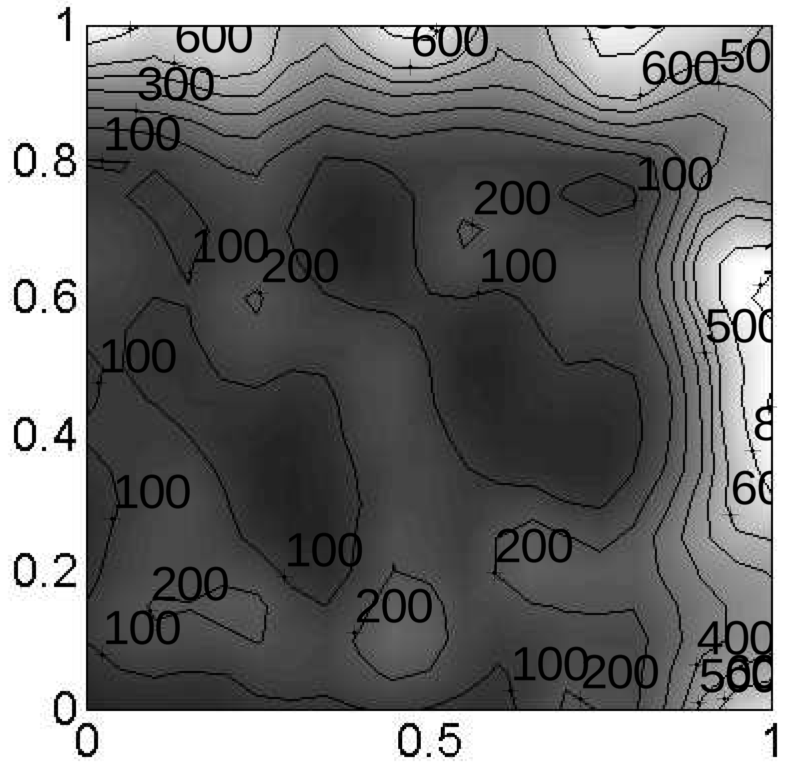
Graphics

Matlab can produce a wide range of graphics, though it can take some work to do it. Just about anything you can do in S you can do in Matlab, though some require an added toolbox (such as the Stat toolbox for doing boxplots). There are also some plots that Matlab will do that I don't think S will, such as a rose plot (angle histogram)

For example here is a scatterplot of points generated by a non-homogeneous Poisson process and a kernel estimator of the intensity function of the process (with the code used to generate the plots)



```
scatter(Locs_high5(:,1), Locs_high5(:,2),'k*','filled')
axis square
box on
set(gca,'FontSize',18)
```

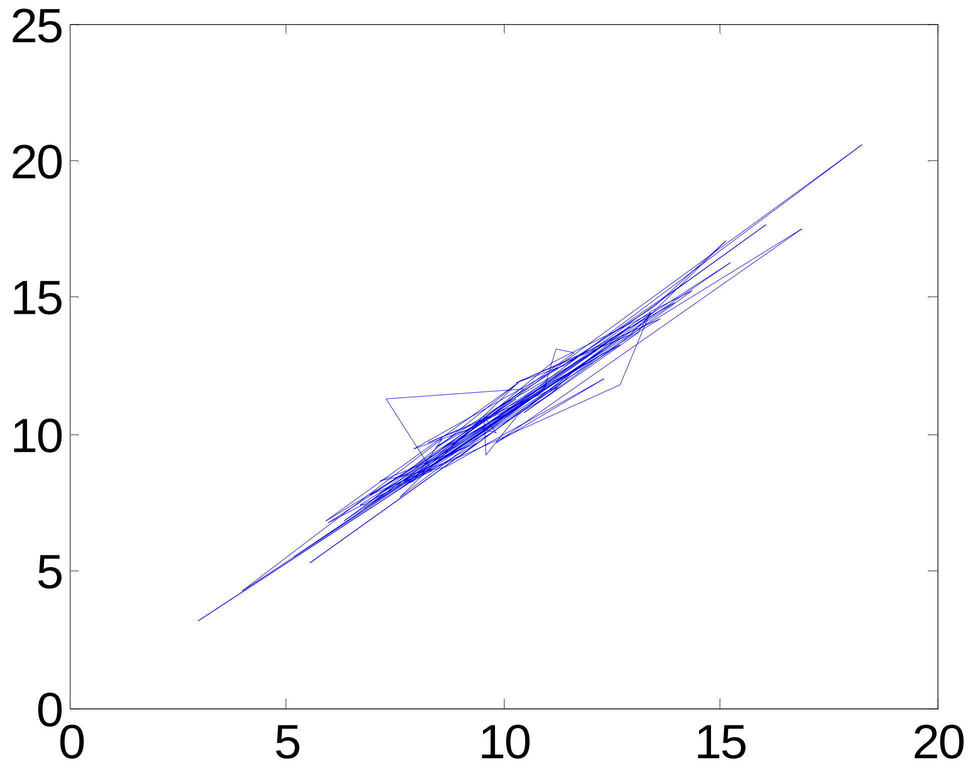


```
pcolor(X,Y,inten_map_high5')
shading interp
set(gca,'FontSize',18)
zlim([minZ maxZ]);
caxis([cminZ cmaxZ]);
axis('square')
colormap gray
hold on
c=contour(X,Y,inten_map_high5',contvec,'k');
clabel(c,'FontSize',18)
hold off
```

Many plots can be done with the `plot` function.

For example

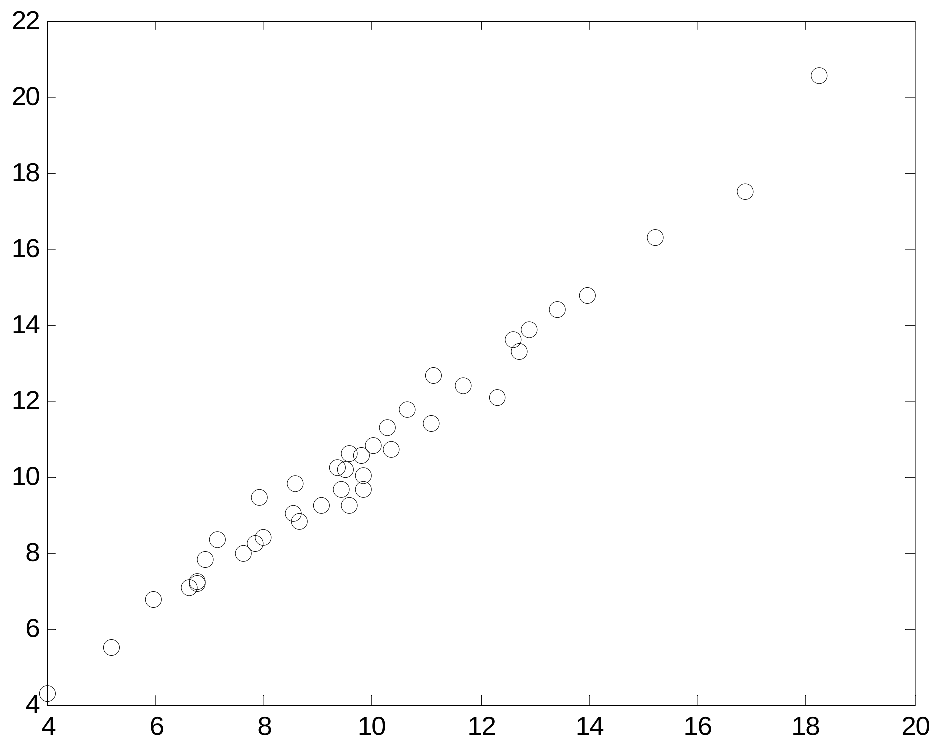
```
>> plot(BTUIn,BTUOut)
```



The default is to do a scatter plot with the points begin joined (in the order they occur in the vectors).

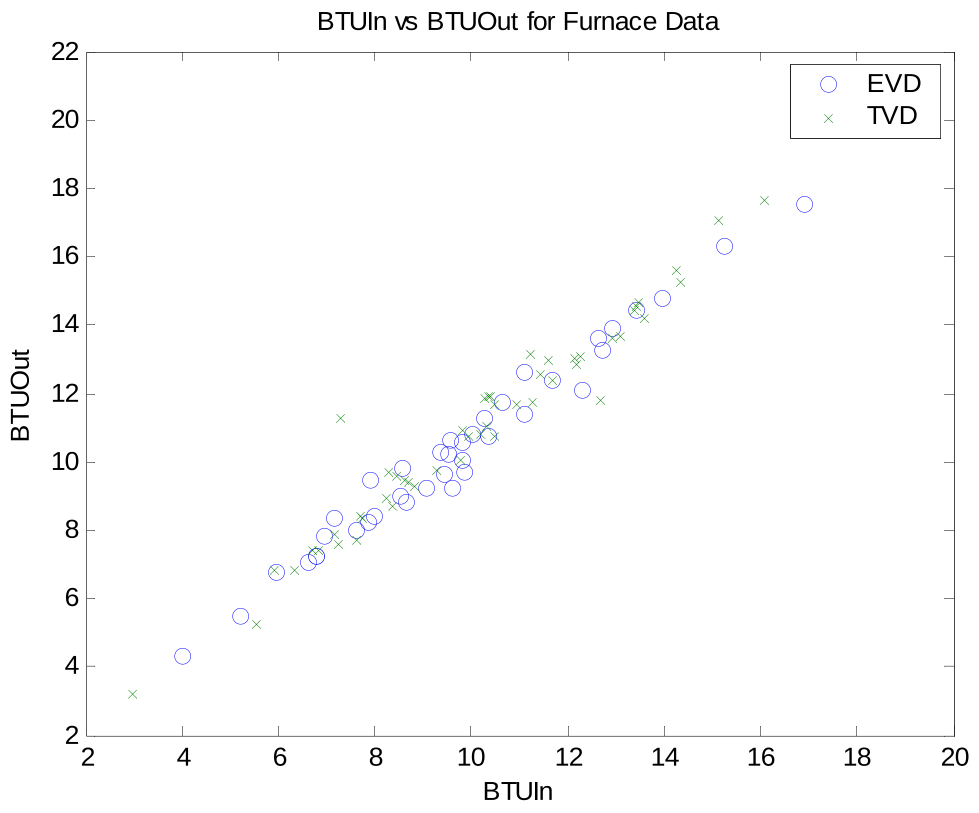
To get what you would really want, do something like

```
>> plot(BTUIn,BTUOut,'ko')
```



The ko options indicate the color (k = black, b = blue, etc) and the symbol (o = circle, x = cross, * = asterisk, etc). There is also an option for line type (- = solid, : = dotted, etc). You can do more complicated things with colors, symbols, and lines with a different approach to giving options.

It is also possible to have multiple plots in a single figure. There are two basic approaches to this.



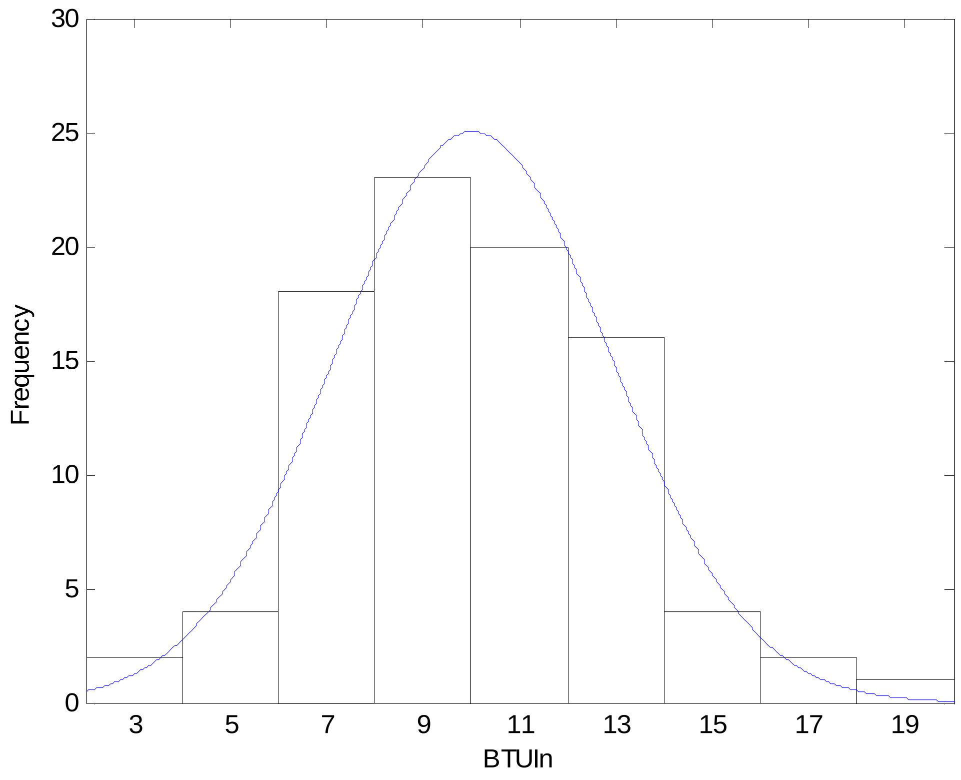
Approach 1:

```
>> plot(BTUIn(damper==1), BTUOut(damper==1), 'o',
BTUIn(damper==2), BTUOut(damper==2),'x')
>> legend('EVD','TVD')
>> xlabel BTUIn
>> ylabel BTUOut
>> title 'BTUIn vs BTUOut for Furnace Data'
```

Approach 2:

```
>> plot(BTUIn(damper==1),BTUOut(damper==1),'o')
>> hold on
>> plot(BTUIn(damper==2),BTUOut(damper==2),'x')
>> legend('EVD','TVD')
>> xlabel BTUIn
>> ylabel BTUOut
>> title 'BTUIn vs BTUOut for Furnace Data'
>> hold off
```

The use of hold on and hold off also allows for different types of plots to be combined, such as historgrams and lines.



```
>> hist(BTUIn,3:2:19)
>> h = findobj(gca,'Type','patch');
>> set(h,'FaceColor','w','EdgeColor','k')
>> hold on
>> plot(2:0.01:20, 180*normpdf(2:0.01:20, mean(BTUIn),
   std(BTUIn))')
>> hold off
>> xlabel BTUIn
>> ylabel Frequency
```

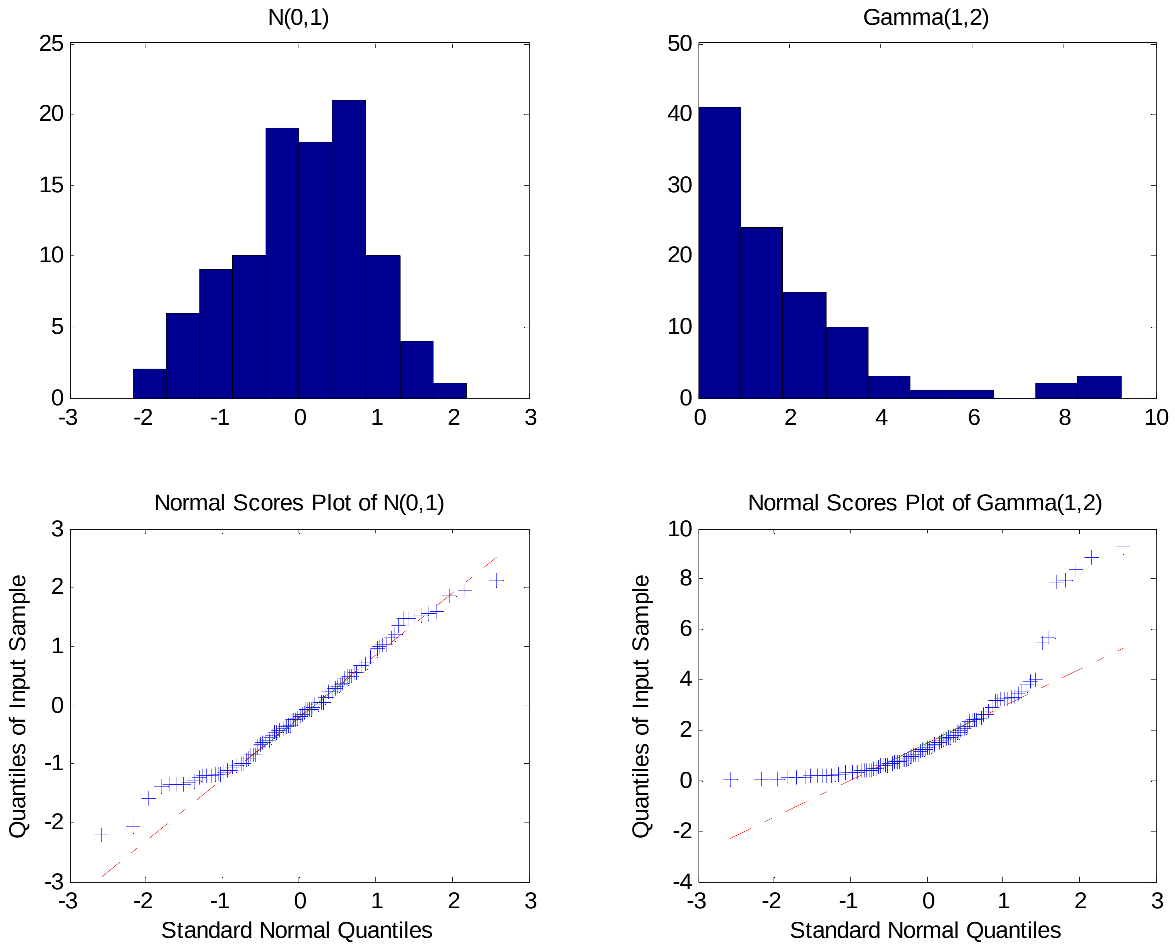
To stop adding new graph elements to a plot, you need to give the `hold off` command. Any new plot commands will generate a new figure. However after giving it you can add axis labels, etc to the figure.

Subplots:

It is also possible to have different plots in a figure, similar to what you can get with `par(mfrow)` in S. In Matlab, it is done with the `subplot` function

```
    subplot(nrows, ncols, element)
```

where element = (row-1) × ncol + col



```
>> subplot(2,2,1)
>> hist(z)
>> title 'N(0,1)'
>> subplot(2,2,2)
>> hist(g)
>> title Gamma(1,2)
>> subplot(2,2,3)
>> qqplot(z)
>> title 'Normal Scores Plot of N(0,1)'
>> subplot(2,2,4)
>> qqplot(g)
>> title 'Normal Scores Plot of Gamma(1,2)'
```

Annotating figures.

There are a number of functions useful for annotating figures. The most useful are

`axis`: Sets axis features, such as as axis limits and aspect ratios

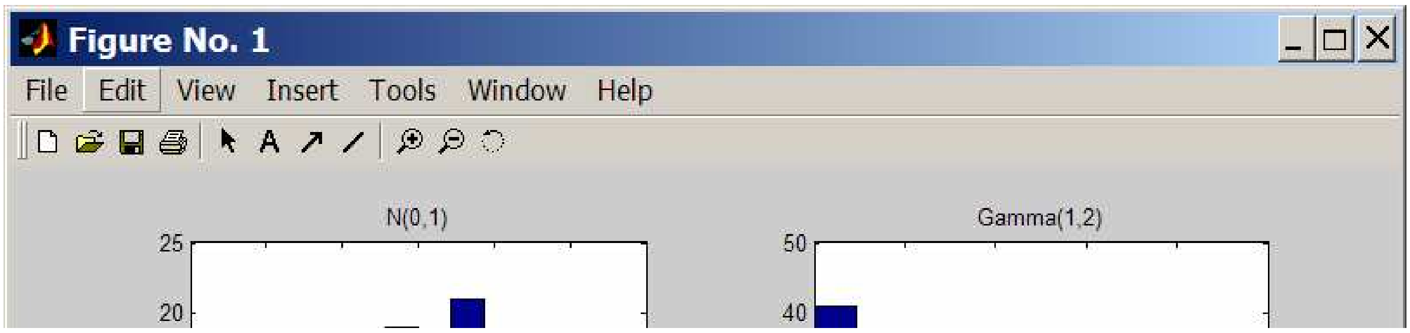
`xlim, ylim`: Sets or gets current axis limits

`xlabel, ylabel`: Sets axis labels on plot

`title`: Sets figure title

`text`: Adds text to a figure at a desired location.

In addition to using these commands you can also do it through the Figure window menus. Many of the things you want to do will be under the Insert menu, though some things you will want to alter will be with Figure properities and Axes properties under the Edit menu.



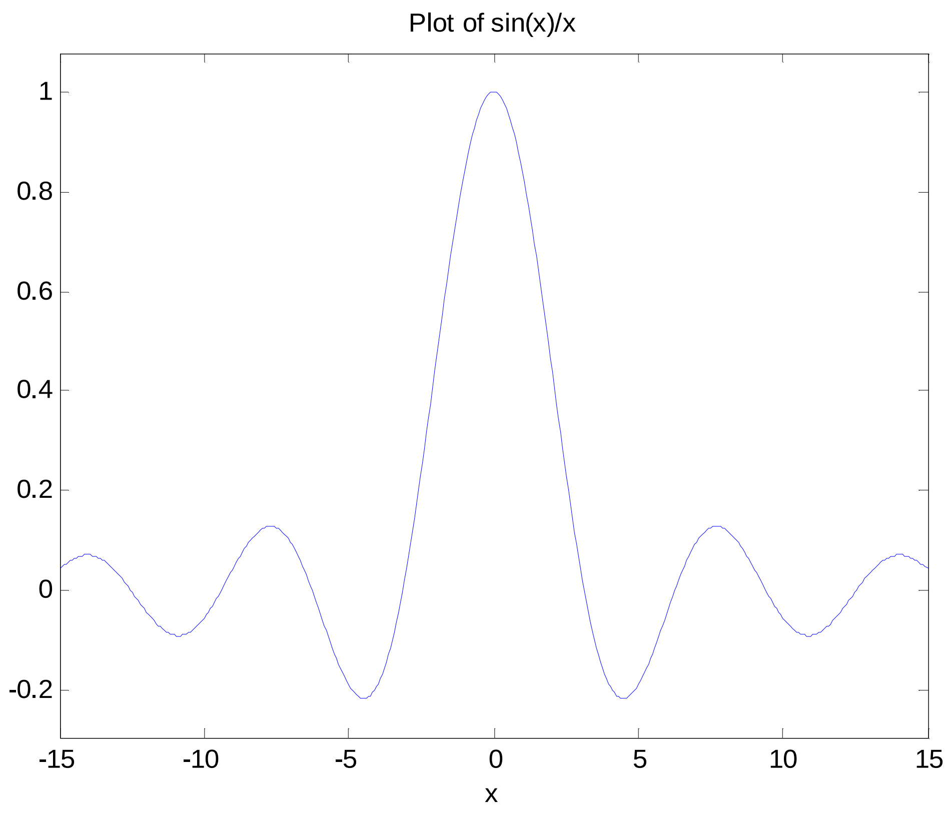
Also when adding text to a figure, such as with title or text, it is possible to do some formatting and to use greek characters. Simple TeX commands can be used. For example, greek characters can be added by adding a \ before the same (e.g. `\mu`). Subscripts (_) and superscripts (^) are also possible. It the text you want to sub/super-script is more than one character, you need to put it in braces. So to set the x-axis to σ_{ij}^2, you can use the command

```
   xaxis('\sigma_{ij}^2)
```

Other useful plots in the basic Matlab setup include scatter plots (`scatter`), pie charts (`pie`), bar charts (`bar` & `barh`), stem plots (`stem`), adding error bars (`errorbar`) and various polar coordinate based plots (`polar`, `compass`, `rose`).

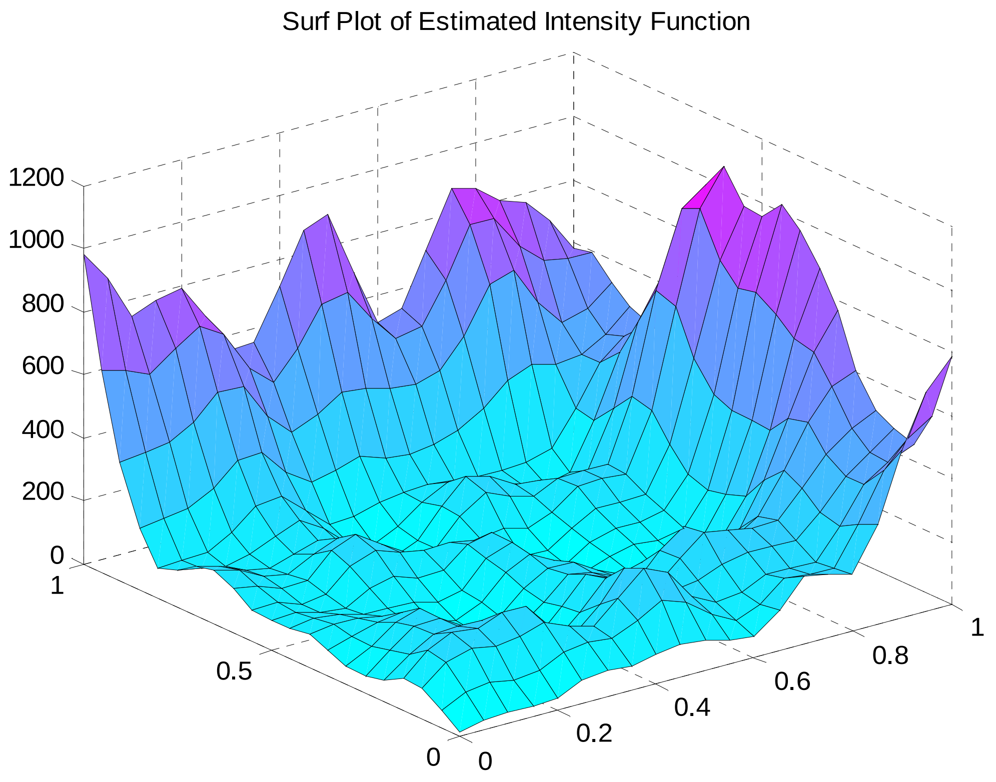
It is also possible to plot functions. For example

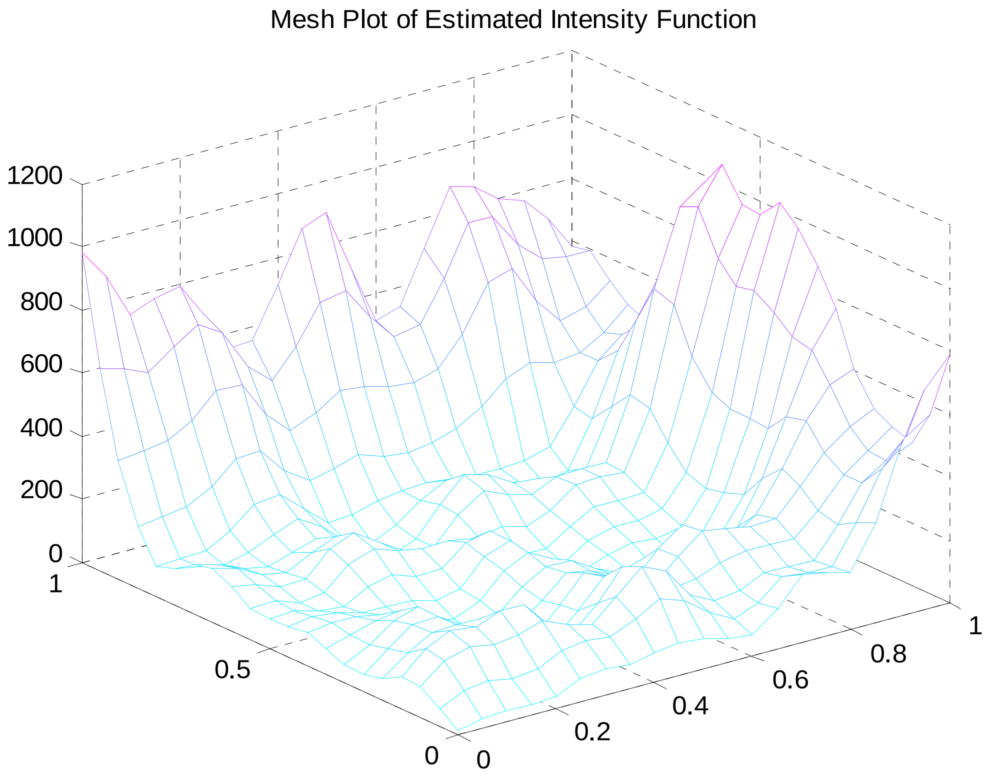
```
>> funct_str = 'sin(x)/x';
>> ezplot(funct_str,[-15 15])
>> title(['Plot of ' funct_str])
```



Also available for doing this are `fplot` (built-in function or function described in .m file) and `ezpolar` (polar coordinates)

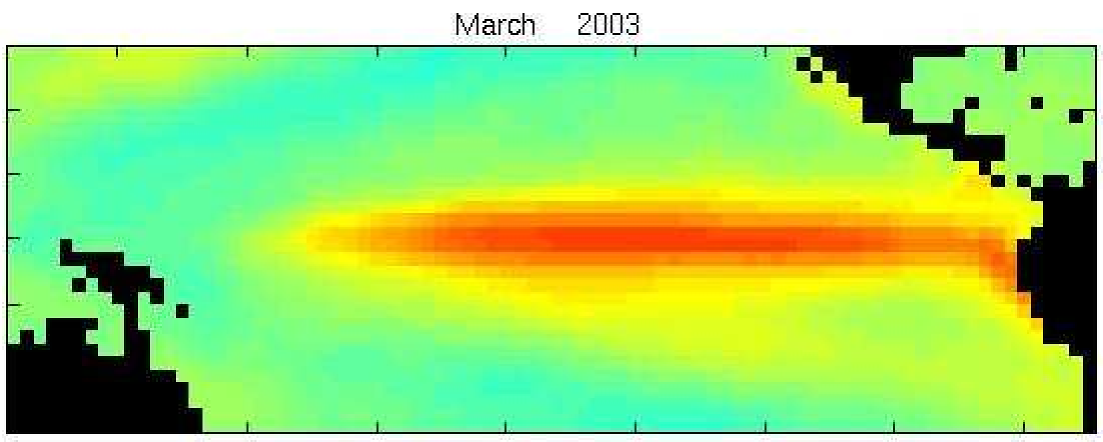
In addition to the 2-D graphics, there are a wide range of 3-D plotting functions. The ones that are most useful are plot3, scatter3, contour (contourf, contour3, pcolor), surf, mesh





In is also possible to plot image files (image, imagesc)

The following is a forecast of tropical Pacific sea surface temperatures based on a Bayesian analysis of a hierarchical model plotted with `imagesc`.



If you want to see it in color, go to
<http://www.stat.ohio-state.edu/~sses/collab_enso_field_forecast.php>

In dealing with these 3-D plots, they tend to be in color. To set the color scheme you can use the colormap function. The map above uses the default colormap (jet: blue to green to yellow to red). The surface and mesh plots earlier use cool (light blue to dark blue to purple). The pcolor plot earlier uses gray (dark to light)

Graphic file formats.

Figures in Matlab can be saved in a number of format. These include Windows Metafiles, postscript, jpeg, png, tiff. Some formats may only be available on certain platforms (wmf?). About the only thing you can't do it gif. You can save figures using Export under the File menu or use a command like

```
   print -depsc2 launcher_low.ps
```

or

```
   print('-djpeg90','test.jpg')
```

The first will save the file as an encapsulated level 2 color postscript and the second will save it as a jpeg will quality level 90.

Statistics Toolbox

This is an add on package from MathWorks. The toolbox has routines that fit into the following categories.

Probability Distributions: Describes the distributions and the distribution-related functions supported by the toolbox.

Descriptive Statistics: Explores toolbox features for working with descriptive statistics such as measures of location and spread, percentile estimates, and data with missing values.

Linear Models: Describes toolbox support for one-way, two-way, and higher-way analysis of variance (ANOVA), analysis of covariance (ANOCOVA), multiple linear regression, stepwise regression, response surface prediction, ridge regression, and one-way multivariate analysis of variance (MANOVA). It also describes support for nonparametric versions of one- and two-way ANOVA, and multiple comparisons of the estimates produced by ANOVA and ANOCOVA functions.

Nonlinear Regression Models: Discusses parameter estimation, interactive prediction and visualization of multidimensional nonlinear fits, and confidence intervals for parameters and predicted values.

Hypothesis Tests: Describes support for common tests of hypothesis - t-tests, Z-tests, nonparametric tests, and distribution tests.

Multivariate Statistics: Explores toolbox features that support methods in multivariate statistics, including principal components analysis, factor analysis, one-way multivariate analysis of variance, cluster analysis, and classical multidimensional scaling.

Statistical Plots: Describes box plots, normal probability plots, Weibull probability plots, control charts, and quantile-quantile plots which the toolbox adds to the arsenal of graphs in MATLAB. It also discusses extended support for polynomial curve fitting and prediction, creation of scatter plots or matrices of scatter plots for grouped data, interactive identification of points on such plots, and interactive exploration of a fitted regression model.

Statistical Process Control: Discusses the plotting of common control charts and the performing of process capability studies.

Design of Experiments: Discusses toolbox support for full and fractional factorial designs, response surface designs, and D-optimal designs. It also describes functions for generating designs, augmenting designs, and optimally assigning units with fixed covariates.

Probability distributions

Available distributions:

Continuous (data)	Continuous (statistics)	Discrete
Beta	Chi-square	Binomial
Exponential	Noncentral Chi-square	Discrete Uniform
Gamma	F	Geometric
Lognormal	Noncentral F	Hypergeometric
Normal	t	Negative Binomial
Rayleigh	Noncentral t	Poisson
Uniform		
Weibull		

Routines: available for pdf, cdf, inverse cdf, random number generation, mean & variance calculations, parameter fitting (MLEs), likelihood functions.

For the normal distribution, these functions are

normpdf, normcdf, norminv, normrnd, normstat, normfit, normlike

Parameter fitting.

The parameter fitting routines tend to give Maximum Likelihood Estimates (MLE) for the parameters. However, if a more popular estimate exists, such the sample standard deviation,*s*, as the estimate of the normal standard deviation is used. To get the MLE for the normal, use the `mle` function

```
>> [xbar, s, cimu, cisigma] = normfit(BTUIn,0.05)

xbar =

   10.0384

s =

    2.8680

cimu =
    9.4378
   10.6391

cisigma =
    2.5015
    3.3613

>> [phat, pci]=mle('norm',BTUIn,0.05)

phat =

   10.0384    2.8520

pci =

    9.4370    2.4338
   10.6399    3.2702
```

The functions, as called above, have the option to give confidence intervals for the parameters. Generally these are based on the asymptotic normality of MLEs. However, if a more popular interval is available, such as for the the standard deviation of the normal based on the scaled chi-squared distribution of s^2, it is used instead. For the confidence intervals , you need to specify the error rate (alpha = 1 – C) instead of the confidence level (C).

Descriptive Statistics.

The routines for desciptive stats fall into 6 groups

Measures of central tendency: `geomean`, `harmmean`, `mean`, `median`, `trimmean`
Measures of dispersion: `iqr`, `mad` (Mean Absolute Deviation), `range`, `std`, `var`

Functions for Data with Missing Values (NaNs): `nanmax`, `nanmean`, `nanmedian`, `nanmin`, `nanstd`, `nansum`

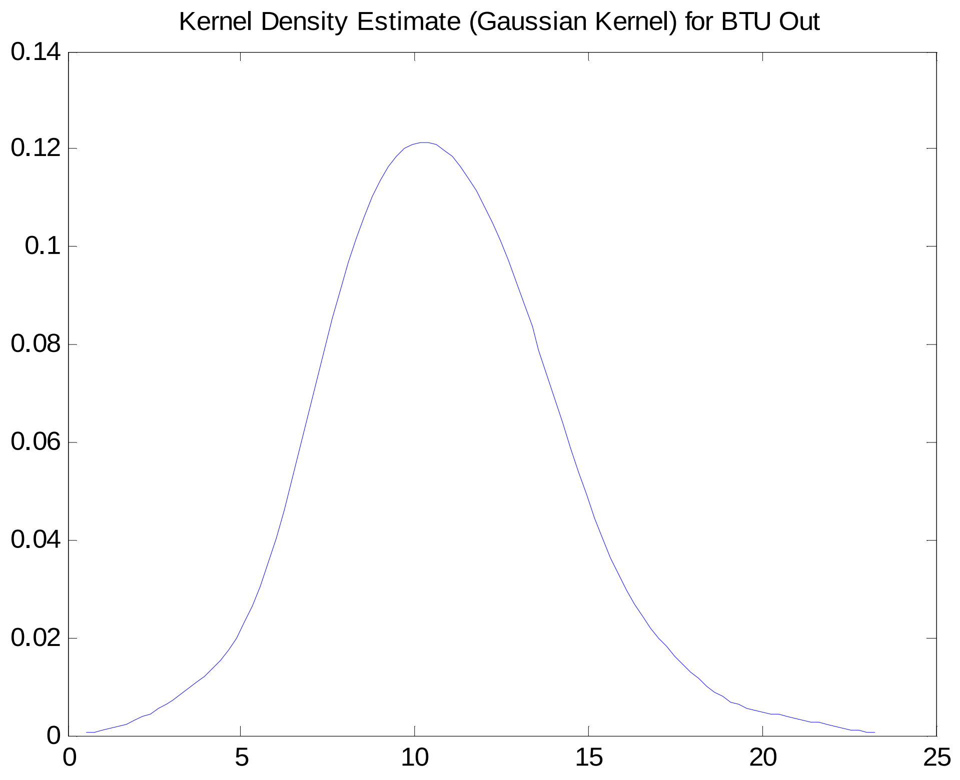
Function for Grouped Data: `grpstat`

Allows for means, and standard errors for the mean ($s/\sqrt{n}$) to be calculated easily for each group. It will also give the counts in each group. Note that more than one grouping variable can be used at a time.

```
>> [means,sem,counts,name] = grpstats(BTUIn,damper)

means =

    9.9078
   10.1430

sem =

    0.4775
    0.3913

counts =

    40
    50

name =

    '1'
    '2'
```

Bootstrapping: A Monte Carlo approach which allows standard errors to be approximated. Useful when normality assumptions for parameters estimates don’t hold.

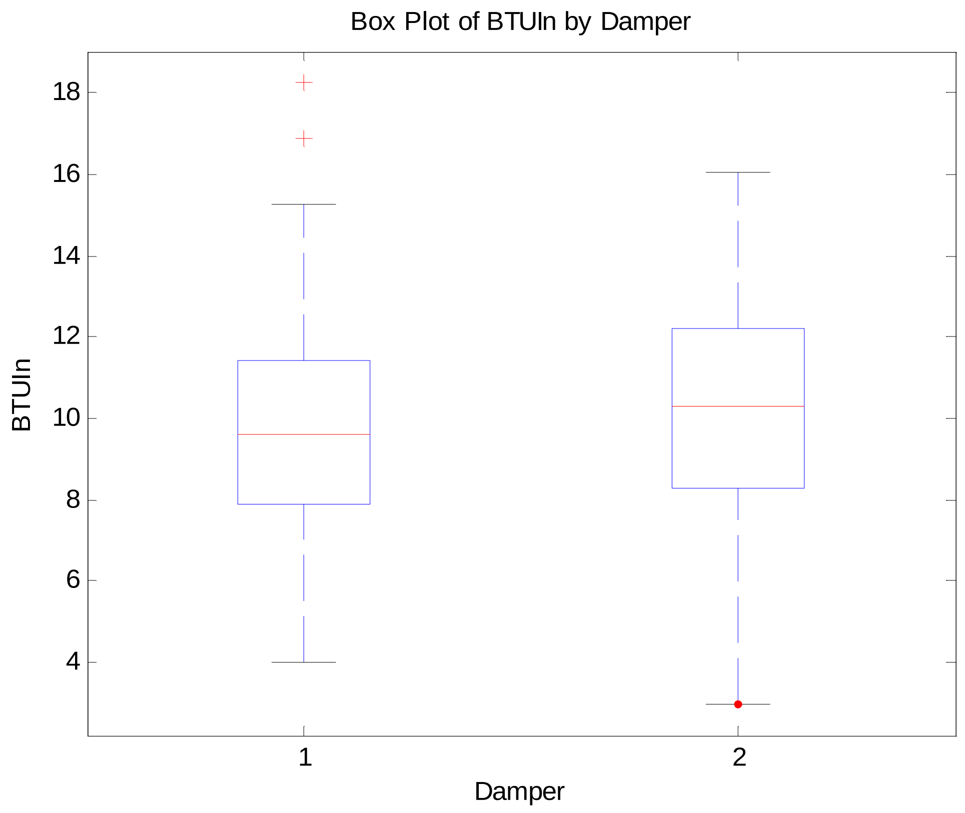
Percentiles and Graphical Descriptions: This included percentiles, kernel density estimates, and empirical cdfs, with the last two being graphical.



```
>> [f,x] = ksdensity(BTUOut)
>> plot(x,f)
>> title('Kernel Density Estimate (Gaussian Kernel)
for BTU Out')
```

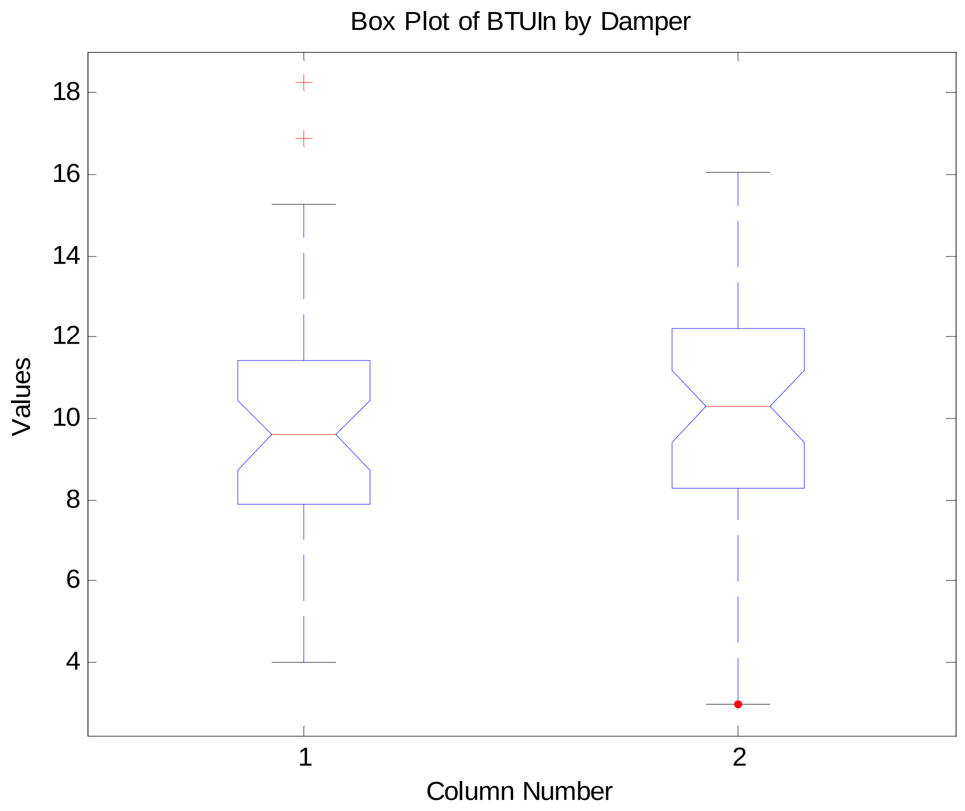
Statistical Plots:

Even though there is a wide range of plotting routines in the base Matlab product, the Stat Toolbox adds a few more. The include Boxplots, distribution plots (normal scores, qqplot, etc) and grouped scatter plots.



```
BTUin = [[BTUIn(damper==1); NaN*zeros(10,1)]
BTUIn(damper==2)]
>> boxplot(BTUin)
>> xlabel 'Damper'
>> ylabel 'BTUIn'
>> title 'Box Plot of BTUIn by Damper'
```

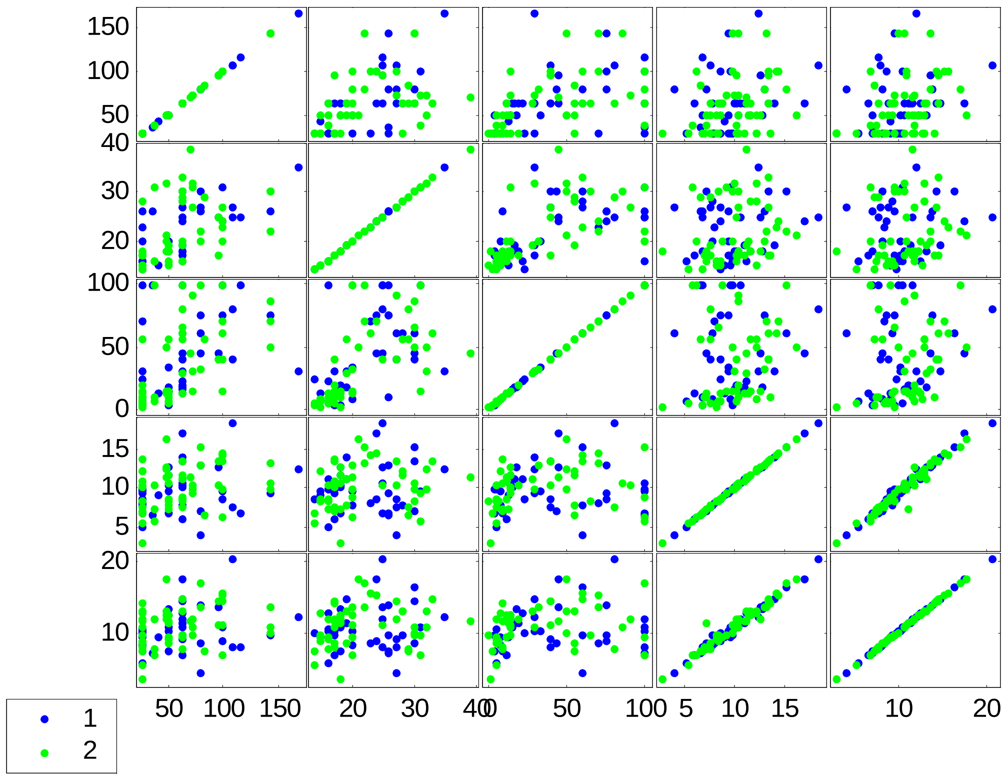
You also have options for doing notches, changing the outlier symbols, the outlier rule (c × IQR), and whether you want to have things vertical or horizontal. Notches are used to indicate confidence intervals for the population median. I think the help page documentation is wrong with the statement “Notches graph a robust estimate of the uncertainty about the means for box-to-box comparison.”



```
>> boxplot(BTUin,1)
```

The distribution plots in the toolbox include normplot (Normal Scores plot), weibplot (Weibull Scores plot), qqplot (comparing two samples).

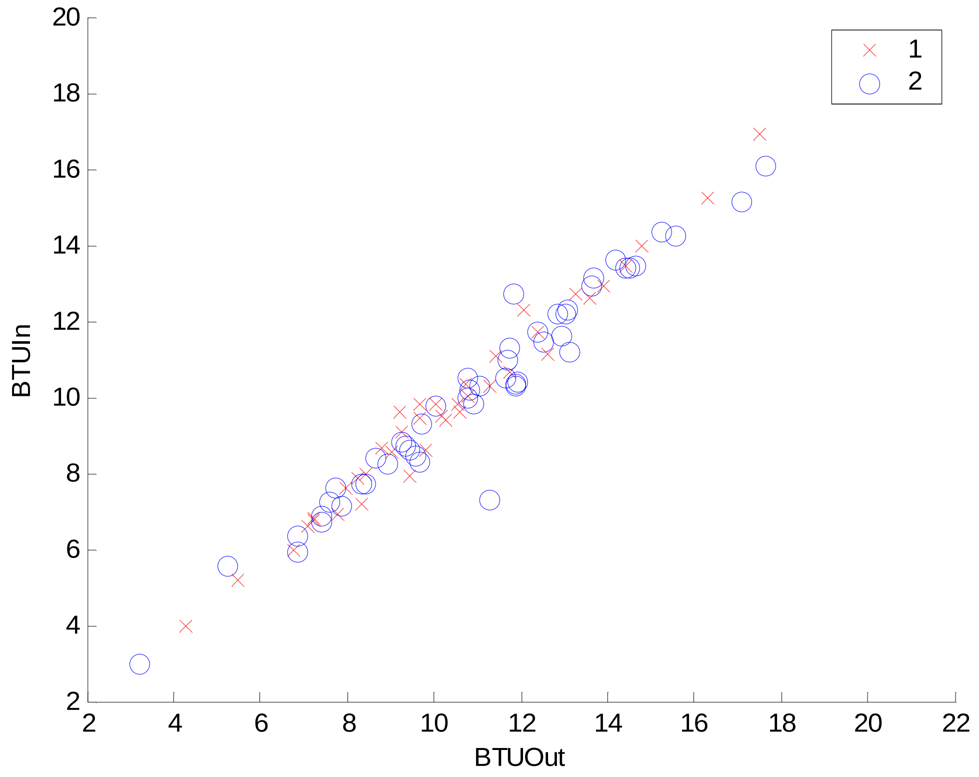
The grouped scatter plots are extentions of scatter (gscatter), and plotmatrix (gplotmatrix).



```
>> gplotmatrix(furnace_num,furnace_num,damper)
```

The plotmatrix commands in Matlab are nice in that you don't have to use the same variables on the xaxis of the plot as the yaxis.

You also have great control of color, symbols, etc with both grouped plots. For example, with gscatter, I can set the color, symbol, and size for each symbol and where I want the legend to be (on or off the plot).

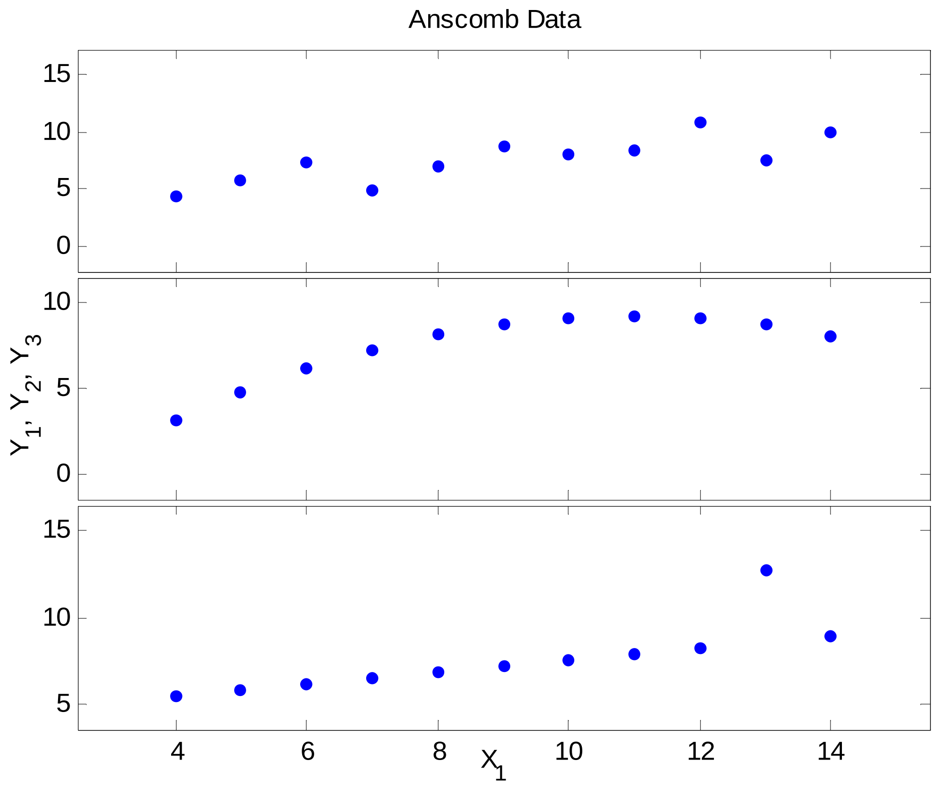


```
>> gscatter(BTUOut, BTUIn, damper, 'rb', 'xo',8,'on')
```

Linear Models.

The Stat Toolbox has functions for dealing with linear models. However it is not as nice as `lm` in S. First the are different routines for different models (`regress`, `anova1`, `anova2`, `anovan`, etc) instead of a single function. Also the functions are based on the matrix formulation of regression, so you need to create your matrices (X in particular) yourself, instead of having the program doing it for you.

Anscomb Data:



```
>> x1mat =[ones(11,1) anscomb(:,1)]
>> [b1,bint1,r,rint,stats1] = regress(anscomb(:,1),x1mat);
>> [b2,bint2,r,rint,stats2] = regress(anscomb(:,3),x1mat);
>> [b3,bint3,r,rint,stats3] = regress(anscomb(:,4),x1mat);

>> [b1 b2 b3]

ans =

    3.0001    3.0009    3.0025
    0.5001    0.5000    0.4997

>> [stats1' stats2' stats3']

ans =

    0.6665    0.6662    0.6663
   17.9899   17.9656   17.9723
    0.0022    0.0022    0.0022
```

Output for `regress`

`b`:	parameter estimates, in order of columns of X
`bint`:	confidence intervals for β
`r`:	residuals
`rint`:	confidence intervals for ε_i
`stats`:	summary of regression – R^2, F, p-value for F test.

This data set was constructed by Frank Anscombe (1973, American Statistician 27, 17-21) to exhibit problems in just blindly accepting regression output and not checking the residuals, etc when performing a regression. The data set has data for 4 regression where the parameters estimates, test statistics, etc are about the same. However only in one case is a linear regression model reasonable. The reason that the all the regression output is the same is that the 5 summary statistics $(\bar{x}, \bar{y}, s_x, s_y, r)$ are all the same.