According to Prof. Adamic: "Please note that all these scripts were written by me, in a rushed fashion, and so are pretty breakable and quite possibly faulty."

In order to run them you either need to add the directory where you have saved the scripts to your matlab search path, or cd into the directory using

> cd c:/dir1/dirwherescriptsare/

1. Generating a power law integer distribution

Use the script randompowerlawints.m as follows

At the Matlab prompt, type
> x = randompowerlawints(numsamples,alpha)

where numsamples is an integer value specifying the number of integer samples you would like, and alpha is the power law exponent in the range from 1 to very big, although it gets boring for alpha > 3.5

2. Binning a sample (works for any distribution, not just power law)

Use the script binvector.m

At the Matlab command prompt type:
> [xlinbin, ylinbin] = binvector(x)

If x is not already an integer array, it will round each value to the nearest integer, and then count how many times an integer occurs. This will be a histogram of your data.
xlinbin is the list ofvalues, and ylinbin is the list of corresponding frequencies

3. Plotting a histogram

For a linear scale
> plot(xlinbin,ylinbin,'ro')

here I've told it with 'ro' to mark the points with red circles

For a semilog or log log plot, use the Matlab commands semilogy or loglog instead of plot

4. Fitting a power law distribution by doing a linear regression on the logarithm of the x and y values

Use the Matlab script fitlineonloglog.m
> alpha = fitlineonloglog(x,y, x_{min})

Note that the xmin argument is optional. If you just call fitlineonloglog(x,y), x_{min} will be set to 0.

The function will return the value of the power-law exponent*(-1) and show the fitted line on the log log plot. There is a factor of -1 because we are fitting x^{-alpha} and the function is reporting on the slope, which is (-alpha).

5. Binning the data in logarithmic bins

First bin the data linearly using binvector (see above), then
use the matlab script logbinfromlinbin.m

> [xlogbin,ylogbin] = logbinfromlinbin(xlinbin,ylinbin,logbase)

logbase determines the width of the bin. If you don't specify it, it will be set to 2. The bins will be of widths, 1, 2, 4, 8, ..
If you would like the bins spaced more closely use a value between 1 and 2..

6. Calculating the cumulative distribution

Get your original data sample x and use the script cumulativecounts.m
> [xlincumulative,ylincumulative] = cumulativecounts(x)

7. Normalizing your distribution

If you want the proportion of time x takes a certain value, then you need to normalize the histograms. In Matlab, normalizing a vector goes as follows:

> y = y/sum(y)

8. Computing the max-likelihood estimate of the power law exponent

Take you data sample x and run the script PLmaxlikelihood.m **If you don't set x_{min}, or if it is set to something small (< 5 or <10), the fit will only be accurate if your data sample is continuous and not integer!**

> alpha = PLmaxlikelihood(x, x_{min})

Note that this will not give you a goodness of fit estimate. If x_{min} is not specified, it will be set to 1.

9. Loading a Pajek partition file into Matlab

Use the hdrload.m script like so:

```
> [header,mydata] = hdrload('../mypajekpartition.clu')
```

Above I assumed that the Pajek data file is one level up in the directory hierarchy, but you can specify the full path. What this script does is it essentially ignores the first line of the file if it contains text. Pajek for example will say something about '* Vertices 100' or something similar.