```
function stats_demo
%Written by Xing 11/6/14.
%Provides examples of how to use Matlab functions to carry out hypothesis
%testing. Structure and format of input & output arguments. This
%demonstrates the most basic capabilities of each function- for more
%options, check the help manual.
%Tests: t-tests, ANOVAs, non-parametric tests, correlations, descriptive stats
%(normality, skewness, kurtosis, scedasticity), chi-squared tests, circular
%statistics.

%Generally speaking,
%h==1: reject the null hypothesis (the confidence interval does not span 0);
%h==0: fail to reject the null hypothesis (the CI spans 0).


%1. t-tests & non-parametric analogs

%Test whether mean of a distribution is significantly different from zero:
mu=10;%mean
sigma=5;%SD
x=normrnd(mu,sigma,1,100);%generate 100 values that form a normal distribution
[h p ci stats]=ttest(x)

%Test whether mean of a distribution is significantly different from a certain value,
m:
mu=10;
sigma=5;
x=normrnd(mu,sigma,1,100);%generate 100 values that form a normal distribution
m=9;%hypothesized mean
[h p ci stats]=ttest(x,m)

%Test whether the means of two differently sized distributions are significantly
different from each other:
mu1=10;
sigma1=5;
mu2=14;
sigma2=5;
x1=normrnd(mu1,sigma1,1,100);%generate 100 values that form the first distribution
x2=normrnd(mu2,sigma2,1,50);%generate 50 values that form the second distribution
[h p ci stats]=ttest2(x1,x2)

%Test whether the means of two distributions are significantly different from each
other:
mu1=10;
sigma1=5;
mu2=12;
sigma2=5;
x1=normrnd(mu1,sigma1,1,100);%generate 100 values that form the first distribution
x2=normrnd(mu2,sigma2,1,100);%generate 100 values that form the second distribution
[h p ci stats]=ttest(x1,x2)%paired t-test
[h p ci stats]=ttest2(x1,x2)%unpaired t-test

%Report the t statistic, the df (in brackets), and the p-value
sprintf(['t(',num2str(stats.df),') = ',num2str(stats.tstat),', p = ',num2str(p)])
sprintf(['t(',num2str(stats.df),') = ',num2str(stats.tstat),', p = %.4f'],p)

%Can also specify whether to perform one-tailed or two-tailed tests

%For non-parametric distributions:

%[p h stats]=signrank(x) performs a two-sided signed rank test of the null
%hypothesis that data in the vector x comes from a continuous, symmetric
%distribution with zero median, against the alternative that the
%distribution does not have zero median. The p-value of the test is
%returned in p.

%[p h stats]=signrank(x,y) performs a paired, two-sided signed rank test of the null
% hypothesis that data in the vector x-y come from a continuous, symmetric
% distribution with zero median, against the alternative that the
% distribution does not have zero median. x and y must have equal lengths.
```

```
% Note that a hypothesis of zero median for x-y is not equivalent to a
% hypothesis of equal median for x and y.

%[p h stats]=ranksum(x,y) performs a two-sided rank sum test of the null hypothesis
% that data in the vectors x and y are independent samples from identical
% continuous distributions with equal medians, against the alternative that
% they do not have equal medians. x and y can have different lengths. The
% p-value of the test is returned in p. The test is equivalent to a
% Mann-Whitney U-test.


%2. correlations and partial correlations

sessions=1:10;%for example, check whether data values change linearly over recording
sessions
data=sessions+(rand(1,10)-0.5)*10;%generate simulated data
plot(sessions,data,'ko','LineStyle','--','MarkerFaceColor','k');
xlim([0 11]);
formattedInput=[sessions' data'];

%Calculate the pairwise linear Pearson's correlation coefficient between two
%variables:
[rho p]=corr(formattedInput);
Rvalue=rho(2)
pval=p(2)
df=length(data)-2
sprintf(['r(',num2str(df),') = ',num2str(Rvalue),', p = %.4f'],pval)

%Calculate the pairwise linear Spearman's correlation coefficient between two
%variables (uses rank order instead of actual values):
[rho p]=corr(formattedInput,'type','Spearman');
Rvalue=rho(2)
pval=p(2)
df=length(data)-2
sprintf(['r(',num2str(df),') = ',num2str(Rvalue),', p = %.4f'],pval)

%Can also specify whether to perform one-tailed or two-tailed tests, e.g.
[rho p]=corr(formattedInput,'tail','right');
Rvalue=rho(2)
pval=p(2)
df=length(data)-2
sprintf(['r(',num2str(df),') = ',num2str(Rvalue),', p = %.4f'],pval)

%Partial correlation to 'partition' out the effects of other variables:
sessions=1:20;%for example, check whether data values change linearly over recording
sessions
data=sessions+(rand(1,20)-0.5)*10;%generate simulated data
otherVariable1=sessions+(rand(1,20)-0.5)*10;%generate simulated data for another
variable, pattern of distribution is similar to that of 'data'
otherVariable2=(rand(1,20)-0.5)*10;%generate simulated data for another variable,
dissimilar to 'data'
[rho p]=partialcorr(data',sessions',[otherVariable1' otherVariable2'])
%see what happens when the patterns underlying BOTH of the 'other variables' are
%similar to that of the original data:
otherVariable2=sessions+(rand(1,20)-0.5)*10;
[rho p]=partialcorr(data',sessions',[otherVariable1' otherVariable2'])
%for the second calculation of rho, when both 'otherVariable1' and
%'otherVariable2' have similar patterns to 'data,' should find that the
%factor 'sessions' is less able to explain patterns in 'data' because effects
%of other variables are now taken into account

%Note that corr is very similar to corrcoef


%3. ANOVA & non-parametric analogs

%1-way ANOVA to check whether means differ between groups:
load hogg
data=hogg;
% hogg =
%group 1     2     3     4     5
```

```
135 %     -------------------------
136 %     24    14    11     7    19
137 %     15     7     9     7    24
138 %     21    12     7     4    19
139 %     27    17    13     7    15
140 %     33    14    12    12    10
141 %     23    16    18    18    20
142 [p table stats] = anova1(data);
143 for columnInd=1:size(data,2)%calculate mean and variance for each group:
144     dataStats(1,columnInd)=mean(data(:,columnInd));
145     dataStats(2,columnInd)=var(data(:,columnInd));
146 end
147 dfBetween=table{2,3};
148 dfWithin=table{3,3};
149 Fstat=table{2,5};
150 [c m h]=multcompare(stats,'dimension',[1 2])%can specify the dimension(s) over which
    to calculate the population marginal means
151 %In figure generated by multcompare, axes are rotated 90 degrees clockwise, relative
    to those generated when calling anona function
152 %Important note: remember that 'anova1' draws boxplots with 25% & 75%
153 %percentiles, whereas multcompare draws graphs with comparison intervals.
154 %Only the latter (comparison intervals) are indicative of significance of
155 %overlap between groups.
156 %Report the F statistic, the between-groups df (the first number in
157 %brackets), the within-groups df (the second number in brackets), and the
158 %p-value
159 sprintf(['F(',num2str(dfBetween),',',num2str(dfWithin),') = ',num2str(Fstat),', p =
    %.4f'],p)
160
161 %N-way ANOVA to check whether means differ between groups, depending on n
162 %factors:
163 load hogg
164 data=hogg;
165 %Let's say that group number is a factor-
166 numValsPerGroup=size(data,1);
167 groupAssignment=[];
168 for groupInd=1:size(data,2)%for each group, create 'coding' index
169     groupAssignment=[groupAssignment zeros(numValsPerGroup,1)+groupInd];
170 end
171 % groupAssignment =
172 %      1     2     3     4     5
173 %      1     2     3     4     5
174 %      1     2     3     4     5
175 %      1     2     3     4     5
176 %      1     2     3     4     5
177 %      1     2     3     4     5
178 %Let's also say that some other independent variable, such as recording
179 %day, is a factor, and that the first 3 rows correspond to Day 1, while the
180 %bottom 3 rows correspond to Day 2-
181 dayAssignment=[];%for each day, create 'coding' index
182 for groupInd=1:size(data,2)
183     dayAssignment=[dayAssignment [1;2;1;2;1;2]];
184 end
185 % dayAssignment =
186 %      1     1     1     1     1
187 %      2     2     2     2     2
188 %      1     1     1     1     1
189 %      2     2     2     2     2
190 %      1     1     1     1     1
191 %      2     2     2     2     2
192 formattedData=data(:);
193 formattedGroupAssignment=groupAssignment(:);
194 formattedDayAssignment=dayAssignment(:);
195 %Run the 2-way ANOVA without examining interaction effects between the two
196 %factors (just examine main effects):
197 [p,table,stats,terms]=anovan(formattedData,{formattedGroupAssignment
    formattedDayAssignment})
198 %Run the 2-way ANOVA with an examination of interactions and main effects:
199 [p,table,stats,terms]=anovan(formattedData,{formattedGroupAssignment
    formattedDayAssignment},'model','full')
200 [p,table,stats,terms]=anovan(formattedData,{formattedGroupAssignment
```

```
    formattedDayAssignment},'model','interaction')
201
202 %Note: for non-parametric data, use rank order instead of actual values,
203 %e.g. a Kruskal-Wallis test (the non-parametric equivalent of a 1-way
204 %ANOVA):
205  p=kruskalwallis(data)%each column in the data corresponds to a group
206  %or a Friedman's test (as a non-parametric version of a 2-way ANOVA):
207  reps=2;%specifies that each pair of rows corresponds to a repetition
208  p=friedman(data,reps)
209
210
211 %4. Lilliefors test for normality:
212
213 %Perform a Lilliefors test of the default null
214 %hypothesis that the sample in vector x comes from a distribution in the
215 %normal family, against the alternative that it does not come from a
216 %normal distribution.
217 mu=10;
218 sigma=5;
219 x=normrnd(mu,sigma,1,100);%generate 100 values that form a normal distribution
220 [h p kstat critval]=lillietest(x)
221
222
223 %5. measure of skewness:
224
225 mu=10;
226 sigma=5;
227 x=normrnd(mu,sigma,1,100);%generate 100 values that form a normal distribution
228 skewnessMeasure=skewness(x)
229 %If skewness is negative, this means that the data are spread out more to
230 %the left of the mean than to the right. If skewness is positive, the data
231 %are spread out more to the right. The skewness of the normal distribution
232 %(or any perfectly symmetric distribution) is zero.
233 xSkewed=x.^2;%create a positively skewed distribution
234 skewnessMeasure=skewness(xSkewed)
235
236
237 %6. measure of kurtosis:
238
239 mu=10;
240 sigma=5;
241 x=normrnd(mu,sigma,1,100);%generate 100 values that form a normal distribution
242 kurtosisMeasure=kurtosis(x)
243 %Kurtosis is a measure of how 'peaked' a distribution is. The kurtosis
244 %of the normal distribution is 3. Distributions that are more outlier-prone
245 %than the normal distribution have kurtosis greater than 3; distributions
246 %that are less outlier-prone have kurtosis less than 3.
247
248
249 %7. tests for scedasticity:
250
251 % Perform a Bartlett test of the null hypothesis that the columns of data
252 % vector x come from normal distributions with the same variance (homoscedastic).
253 %The alternative hypothesis is that not all columns of data have the same
254 % variance heteroscedastic).
255 mu1=10;%mean
256 sigma1=5;%SD
257 mu2=14;
258 sigma2=10;%try it with an SD that is closer to the SD of the first distribution, e.g.
    sigma2=6;
259 x1=normrnd(mu1,sigma1,100,1);%generate 100 values that form the first distribution
260 x2=normrnd(mu2,sigma2,100,1);%generate 100 values that form the second distribution
261 formattedInput=[x1 x2];%different groups separated by column
262 [p stats]=vartestn(formattedInput)
263 %Perform a Levene's test for equal variances:
264 p = vartestn(formattedInput,[],'on','robust')
265 load carsmall;
266 p = vartestn(MPG,Model_Year,'TestType','LeveneAbsolute')%newer versions of Matlab
267 p = vartestn(MPG,Model_Year,'on','robust')%older versions of Matlab
268 %Perform a Brown-Forsythe test for equal variances:
269 load examgrades;
```

```
270 [p,stats] = vartestn(grades,'TestType','BrownForsythe','Display','on')%newer versions
    of Matlab
271
272
273 %8. Chi-square test of variance:
274 mu=10;%mean
275 sigma=5;%standard deviation
276 x=normrnd(mu,sigma,1,100);%generate 100 values that form a normal distribution
277 v=sigma^2;%hypothesized variance (variance=SD^2)
278 [h p ci stats]=vartest(x,v)
279
280
281 %9. Chi-square goodness of fit test:
282 %Perform a chi-square goodness-of-fit test of the default
283 %null hypothesis that the data in vector x are a random sample from a
284 %normal distribution with mean and variance estimated from x, against the
285 %alternative that the data are not normally distributed with the estimated
286 %mean and variance.
287 [h p ci stats]=chi2gof(x)
288 %for example:
289 observedOrientations=rand(1000,1)*180;%create simulated data consisting of 1000
    values, from a uniform distribution with a range of 0 to 180
290 edges=linspace(0,180,4)%create a certain number of bins (specify number of edges)
291 expectedCounts=zeros(1,length(edges)-1)+length(observedOrientations)/(length(edges)-1)
292 [h p stats]=chi2gof(observedOrientations,'edges',edges,'expected',expectedCounts)
293 %Report the degrees of freedom, the sample size, the Chi-square value, and the
    p-value:
294 sprintf(['X(',num2str(stats.df),',',num2str(length(observedOrientations)),') =
    ',num2str(stats.chi2stat),', p = %.4f'],p)
295 %another example, this time using a non-uniform distribution:
296 mu=1;%mean
297 sigma=2;%SD
298 observedOrientations=normrnd(mu,sigma,1000,1)*180;%create simulated data consisting
    of 1000 values, from a normal (i.e. non-uniform) distribution with a range of 0 to 180
299 edges=linspace(0,180,4)%create a certain number of bins (specify number of edges)
300 expectedCounts=zeros(1,length(edges)-1)+length(observedOrientations)/(length(edges)-1)
301 [h p stats]=chi2gof(observedOrientations,'edges',edges,'expected',expectedCounts)
302 %Report the degrees of freedom, the sample size, the Chi-square value, and the
    p-value:
303 sprintf(['X(',num2str(stats.df),',',num2str(length(observedOrientations)),') =
    ',num2str(stats.chi2stat),', p = %.4f'],p)
304
305
306 %10. Pearson's Chi-square test of independence:
307
308 %table = crosstab(x1,x2) returns a cross-tabulation table of two vectors of
309 %the same length x1 and x2. table is m-by-n, where m is the number of
310 %distinct values in x1 and n is the number of distinct values in x2.
311 %x1 and x2 are grouping variables.
312 %table(i,j) is a count of indices where grp2idx(x1) is i and grp2idx(x2) is
313 %j. The numerical order of grp2idx(x1) and grp2idx(x2) order rows and columns of
    table, respectively.
314 %[table,chi2,p] = crosstab(x1,...,xn) also returns the chi-square statistic chi2 and
    its p value p for a test that table is independent in each dimension. The null
    hypothesis is that the proportion in any entry of table is the product of the
    proportions in each dimension.
315 [table chi2 p]=crosstab(x1,x2)
316
317
318 %11. tests for circular statisticss (e.g. a von Mises distribution)
319
320 %Download circstat toolbox and add paths to the folder containing the
321 %circstats functions. This demontrates a few handy functions; there are
322 %many more.
323 observedOrientations=rand(100,1)*360;%create simulated data from a uniform
    distribution with a range of 0 to 360
324 convertedOri=observedOrientations/180*pi;%convert to radians
325 %Plot data:
326 figure
327 rho=ones(length(convertedOri),1)-0.25;%slightly shorter stem
328 [x,y] = pol2cart(convertedOri,rho);
```

```
329 handle=compass(x,y); hold on
330 set(handle,{'Color'},{[1 0 0]},'LineWidth',2)
331
332 %Rayleigh test to identify unimodal deviation from uniformity:
333 pRayleigh=circ_rtest(convertedOri);
334
335 %Omnibus test to identify multimodal deviations from uniformity:
336 pOmnibus=circ_otest(convertedOri);
337
338 %Watson-Williams test to compare means of two samples. (N.A. if not enough
339 %data present). Performs like a one-way ANOVA test for circular data
340 observedOrientations2=rand(20,1)*180;%create simulated data from a uniform
    distribution with a range of 0 to 360
341 convertedOri2=observedOrientations2/180*pi;%convert to radians
342 [p stats]=circ_wwtest(convertedOri',convertedOri2');
343 %Plot data:
344 shortenStemFactor=2;%set to two to make arrows half the length
345 rho=ones(length(convertedOri2),1)-0.25*shortenStemFactor;%slightly shorter stem
346 [x,y] = pol2cart(convertedOri2,rho);
347 handle=compass(x,y); hold on
348 set(handle,{'Color'},{[0 0 1]},'LineWidth',2)
349
350
351 %12. test goodness-of-fit for regression:
352 figure
353 numVals=20;
354 x=[1:numVals]';
355 y=x+normrnd(1,2,numVals,1);
356 %Introduce an outlier to examine how robust fitting compares with ordniary
357 %least squares:
358 y(numVals)=y(1);
359 scatter(x,y,'filled','r');
360 hold on
361 %To perform a multilinear regression which is robust to outliers:
362 %B = ROBUSTFIT(X,Y) returns the vector B of regression coefficients,
363 %   obtained by performing robust regression to estimate the linear model
364 %   Y = Xb.  X is an n-by-p matrix of predictor variables, and Y is an
365 %   n-by-1 vector of observations.
366 [brob stats]=robustfit(x,y)%brob(1) is the intercept of the best-fit line; brob(2) is
    the slope
367 stats.p%returns p-values for each coefficient in the model, indicating
368 %whether a given coefficient differes significantly from zero.
369 stats.p(2)%Good for checking whether the slope of a regression line is
370 %significantly different from zero.
371 plot(x,brob(1)+brob(2)*x,'k','LineWidth',2)
372 %To perform a multiple linear regression using least squares:
373 %   B = REGRESS(Y,X) returns the vector B of regression coefficients in the
374 %   linear model Y = X*B.  X is an n-by-p design matrix, with rows
375 %   corresponding to observations and columns to predictor variables.  Y is
376 %   an n-by-1 vector of response observations.
377 [bls bint r rint stats]=regress(y,[ones(numVals,1) x])
378 %The vector STATS contains, in the following order: the R-square statistic,
379 %the F statistic and p value for the full model, and an estimate of the error
    variance.
380 stats(3)%This is the p-value for the whole model (from the F test for
381 %whether the model fits significantly better than the null model,
382 %'y = constant'- as opposed to the current model with a slope and an
383 %intercept, 'y = b + ax').
384 plot(x,bls(1)+bls(2)*x,'b','LineWidth',2);
385 %Notice how the outlier affects curve fitting when using an ordinary least
386 %squares approach, whereas robustfit remains unaffected.
387
388 %Alternatively, use the fitlm function:
389 load hospital
390 y = hospital.BloodPressure(:,1);
391 X = double(hospital(:,2:5));
392 mdl = fitlm(X,y)%Requires newer version of Matlab
393 mdl.Rsquared.Ordinary
394 mdl.Rsquared.Adjusted
395
396 close all hidden%command used to close statistical tables as well as ordinary figures
```