BIOL 708L/NACS 728B Lab #11
Fall 2011

Permutation Test Example

The goal of this lab is to use a permutation test to determine if the means of two distributions are different.

1) Download the Matlab file containing the data from the Lab page of the course website, or directly from < http://terpconnect.umd.edu/~jzsimon/biol708L/lab/permExData.mat >, and load it into Matlab.
   a) Once you have loaded the data, use `whos`, or inspect the Matlab workspace, to see what the new variables are that you have loaded. There should be two, `ILEC_data` and `CLEC_data`. Both data sets are lists of latencies (measured in units of hours).
   b) Put `ILEC_data` into a new variable called `data1`, and `CLEC_data` into a new variable called `data2`. Define `n1` and `n2` to be the lengths of the respective datasets.
   c) Examine histograms of the latency data of `data1`. Use the command `hist()` to do this, using `data1` as its first argument and the number of histogram-bins as its second. Try 10, 20, 30, 40, 100, and possibly others, for the number of bins. The best value is the largest number that best reflects the shape of the probability distribution without giving too much bin-to-bin variability. What is the best value? There is no one answer to this question, but find a number that feels right (try 1000 to see what is definitely too big). Can you see evidence that the latencies are measured in hours?
   d) Repeat the previous step, in a new figure, but for `data2`. The number of latencies in `data2` is much smaller than for `data1`, so you will probably end up with a very different answer for the "best" number of bins. Set the x-axis limits to match those of `data1` so you can compare the two as fairly as possible. Can you argue that the two datasets arose from the same distribution? Or the opposite?
   e) Calculate `m1`, the mean of `data1`, and `m2`, the mean of `data2`, and `dm = m2 - m1`. Would you guess `dm` is statistically significantly positive? At the 0.05 level? at 0.01? In the field that these data were collected in, statistical significance is measured at the 0.01 level. In this example, only positive differences matter, so a one-sided test is appropriate.
   f) Calculate whether the difference of the means is significantly positive according to a t-test:
   ```
   [h, p_ttest]=ttest2(data1, data2, 0.01, 'left');
   ```
   It turns out that the t-test's estimate of p, `p_ttest`, is incorrect (overly small) by a large factor, because the distributions of the latencies are so strongly non-Gaussian.

2) Perform the permutation test.
   a) First, before the test itself, there are some housekeeping details to take care of. We'll use $n_{PT} = 999$ permutations, but you could also try, e.g., 9999.
   ```
   nPT = 1000-1;
   ```
   Then lump the two datasets together into a common pool.
   ```
   datapool = [data1;data2];
   ```
   b) You are about to calculate `nPT` different instances of the difference between the means of the two

populations, each compatible with the permutation null hypothesis. Create the array to put them in:

```
dmPT = nan(1,nPT);
```

c) Make a `for` loop, with `k` running from `1` to `nPT`. Inside this loop perform each of these steps

i) Create a new version of `datapool`, `datapoolPT`, whose labels are randomly permuted:

```
datapoolPT = datapool(randperm(length(datapool)));
```

ii) Assign the first `n1` values of `datapoolPT` to a new version of `data1` called `data1PT`:

```
data1PT = datapoolPT(1:n1);
```

iii) Similarly assign the remaining `n2` values of `datapoolPT` to `data2PT`. Double check that you are indeed using `n2` values (and not more, or less, than `n2`), and that you are not using any values that you already used in `data1PT`.

iv) Calculate the means for both `data1PT` and `data2PT` (just like you did above for `data1` and `data2`), and put their difference (e.g. `m2PT – m1PT`) into `dmPT(k)`.

d) Run the loop. Now the `nPT` elements of `dmPT` (mean differences) form a distribution compatible with the null hypothesis. Calculate the number of elements in `dmPT` whose value is greater or equal to `dm`, that is, `nge = sum(dm>=dmPT)`. The *p* value associated with this distribution is `p_PT_dm = 1 – (nge + 1)/(nPT +1)`. Is it significant at the 0.01 level? How does this compare with the t-test's estimate of *p*? By approximately by what factor is the t-test's value wrong by?

3) Illustrate the null distribution and show where various *p* and α values fall.

a) First sort the elements of `dmPT` and find which elements correspond to the 95th and 99th percentiles:

```
dmsorted = sort(dmPT);
dm95 = dmsorted(0.95*(nPT+1));
dm99 = dmsorted(0.99*(nPT+1));
```

Why does this work?

b) Do the same for the t-test's incorrect value of *p*, to see how far off it was:

```
dmtt = dmsorted(round((1-p_ttest)*(nPT+1)));
```

How far off was the t-tests value?

c) In a new figure, try this Matlab code and describe the information it conveys:

```
hist(dmPT,50)
title('mean difference permutation distribution')
line(dm95*[1 1],[0 nPT/10],'linewidth',1,'color',[1 0 1])
line(dm99*[1 1],[0 nPT/10],'linewidth',1,'color',[1 0 0])
line(dm*[1 1],[0 nPT/10],'linewidth',1,'color',[0 1 0])
line(dmtt*[1 1],[0 nPT/10],'linewidth',1,'color',[0 1 1])
legend('permutation values', '95% significance', '99% significance',...
        'measured value', 't-test equivalent (incorrect)',0)
```