Spring 2008 - Stat C141/ Bioeng C141 - Statistics for Bioinformatics

Course Website: http://www.stat.berkeley.edu/users/hhuang/141C-2008.html
Section Website: http://www.stat.berkeley.edu/users/mgoldman

GSI Contact Info:

Megan Goldman
mgoldman@stat.berkeley.edu
Office Hours: 342 Evans M 10-11, Th 3-4, and by appointment

1 Why is multiple testing a problem?

Say you have a set of hypotheses that you wish to test simultaneously. The first idea that might come to mind is to test each hypothesis separately, using some level of significance α. At first blush, this doesn't seem like a bad idea. However, consider a case where you have 20 hypotheses to test, and a significance level of 0.05. What's the probability of observing at least one significant result just due to chance?

$$\begin{aligned}\mathbb{P}(\text{at least one significant result}) &= 1 - \mathbb{P}(\text{no significant results}) \\ &= 1 - (1 - 0.05)^{20} \\ &\approx 0.64\end{aligned}$$

So, with 20 tests being considered, we have a 64% chance of observing at least one significant result, even if all of the tests are actually not significant. In genomics and other biology-related fields, it's not unusual for the number of simultaneous tests to be quite a bit larger than 20... and the probability of getting a significant result simply due to chance keeps going up.

Methods for dealing with multiple testing frequently call for adjusting α in some way, so that the probability of observing at least one significant result due to chance remains below your desired significance level.

2 The Bonferroni correction

The Bonferroni correction sets the significance cut-off at α/n. For example, in the example above, with 20 tests and $\alpha = 0.05$, you'd only reject a null hypothesis if the p-value is less than 0.0025. The Bonferroni correction tends to be a bit too conservative. To demonstrate this, let's calculate the probability of observing at least one significant result when using the correction just described:

$$\begin{aligned}\mathbb{P}(\text{at least one significant result}) &= 1 - \mathbb{P}(\text{no significant results})\\ &= 1 - (1 - 0.0025)^{20}\\ &\approx 0.0488\end{aligned}$$

Here, we're just a shade under our desired 0.05 level. We benefit here from assuming that all tests are independent of each other. In practical applications, that is often not the case. Depending on the correlation structure of the tests, the Bonferroni correction could be extremely conservative, leading to a high rate of false negatives.

3 The False Discovery Rate

In large-scale multiple testing (as often happens in genomics), you may be better served by controlling the false discovery rate (FDR). This is defined as the proportion of false positives among all significant results. The FDR works by estimating some rejection region so that, on average, FDR $< \alpha$.

4 The positive False Discovery Rate

The positive false discovery rate (pFDR) is a bit of a wrinkle on the FDR. Here, you try to control the probability that the null hypothesis is true, given that the test rejected the null. This method works by first fixing the rejection region, then estimating α, which is quite the opposite of how the FDR is handled. For gory levels of detail, see the Storey paper the professor has linked to from the class website.

5 Comparing the three

First, let's make some data. For kicks and grins, we'll use the random normals in such a way that we'll know what the result of each hypothesis test should be.

```
x <- c(rnorm(900), rnorm(100, mean = 3))
p <- pnorm(x, lower.tail = F)
```

These functions should all be familiar from the first few weeks of class. Here, we've made a vector, x, of length 1000. The first 900 entries are random numbers with a standard normal distribution. The last 100 are random numbers from a normal distribution with mean 3 and sd 1. Note that I didn't need to indicated the sd of 1 in the second bit; it's the default value. The second line of code is finding the p-values for a hypothesis test on each value of x. The hypothesis being tested is that the value of x is not different from 0, given the entries are drawn from a standard normal distribution. The alternate is a one-sided test, claiming that the value is larger than 0.

Now, in this case, we know the truth: The first 900 observations should fail to reject the null hypothesis: they are, in fact, drawn from a standard normal distribution and any difference between the observed value and 0 is just due to chance. The last 100 observations should reject the null hypothesis: the difference between these values and 0 is not due to chance alone.

Let's take a look at our p-values, adjust them in various ways, and see what sort of results we get. Note that, since we all will have our own random vectors, your figures will probably be a different from mine. They should be pretty close, however.

5.1 No corrections

```
test <- p > 0.05
summary(test[1:900])
summary(test[901:1000])
```

The two summary lines will give me a count of how many p-values where above and below .05. Based on my random vector, I get something that looks like this:

```
> summary(test[1:900])
   Mode   FALSE    TRUE
logical      46     854
> summary(test[901:1000])
   Mode   FALSE    TRUE
logical      88      12
```

The type I error rate (false positives) is 46/900 = 0.0511. The type II error rate (false negatives) is 12/100 = 0.12. Note that the type I error rate is awfully close to our α, 0.05. This isn't a coincidence: α can be thought of as some target type I error rate.

5.2 Bonferroni correction

We have $\alpha = 0.05$, and 1000 tests, so the Bonferroni correction will have us looking for p-values smaller than 0.00005:

```
> bonftest <- p > 0.00005
> summary(bonftest[1:900])
   Mode   FALSE    TRUE
logical       1     899
> summary(bonftest[901:1000])
   Mode   FALSE    TRUE
logical      23      77
```

Here, the type I error rate is 1/900 = 0.0011, but the type II error rate has skyrocketed to 0.77. We've reduced our false positives at the expense of false negatives. Ask yourself: which is worse? False positives or false negatives? Note: there isn't a firm answer. It really depends on the context of the problem and the consequences of each type of error.

5.3 FDR

For the FDR, we want to consider the ordered p-values. We'll see if the kth ordered p-value is larger than $\frac{k \times .05}{1000}$.

```
psort <- sort(p)
fdrtest <- NULL
for (i in 1:1000)
fdrtest <- c(fdrtest, p[i] > match(p[i],psort) * .05/1000)
```

Let's parse this bit of code. I want the string of trues and falses to be in the same order as the original p-values, so we can easily pick off the errors. I start by creating a separate variable, psort, which holds the same values as p, but sorted from smallest to largest. Say I want to test only the first entry of p:

```
> p[1] > match(p[1],psort) * .05/1000
[1] TRUE
```

p[1] picks off the first entry from the vector p. match(p[1],psort) looks through the vector psort, finds the first value that's exactly equal to p[1], and returns which entry of the vector it is. In my random vector, match(p[1], psort) returns 619. That means that, if you put all the p-values in order from smallest to largest, the 619th largest value is the one that appears first in my vector. The value you get might differ pretty wildly in this case. Anyhow, on to see how the errors go:

```
> summary(fdrtest[1:900])
   Mode   FALSE    TRUE
logical       3     897
> summary(fdrtest[901:1000])
   Mode   FALSE    TRUE
logical      70      30
```

Now we have a type I error rate of 3/900 = 0.0033. The type II error rate is now 30/70 = 0.30, a big improvement over the Bonferroni correction!

5.4 pFDR

The pFDR is an awful lot more involved, coding-wise. Mercifully, someone's already written the package for us.

```
> library(qvalue)
> pfdrtest <- qvalue(p)$qvalues > .05
```

The qvalue function returns several things. By using $qvalues after the function, it says we only really want the bit of the output they call ”qvalues”.

```
> summary(pfdrtest[1:900])
   Mode   FALSE    TRUE
logical       3     897
> summary(pfdrtest[901:1000])
   Mode   FALSE    TRUE
logical      70      30
```

I seem to get the same results as in the regular FDR, at least at the 5% level. Let's take a look at the cumulative number of significant calls for various levels of α and the different corrections:

alpha	0.0001	0.001	0.01	0.025	0.05	0.1
Uncorrected	31	57	93	118	134	188
Bonferroni	0	6	13	21	24	31
FDR	0	19	44	63	73	91
pFDR	0	20	48	64	73	93

Here's how the type I errors do:

alpha	0.0001	0.001	0.01	0.025	0.05	0.1
Uncorrected	0.0011	0.0022	0.0144	0.0344	0.0511	0.1056
Bonferroni	0	0	0	0.0011	0.0011	0.0011
FDR	0	0	0.0011	0.0022	0.0033	0.0122
pFDR	0	0	0.0011	0.0022	0.0033	0.0144

... and type II errors:

alpha	0.0001	0.001	0.01	0.025	0.05	0.1
Uncorrected	0.70	0.45	0.20	0.13	0.12	0.07
Bonferroni	1	0.94	0.87	0.80	0.77	0.70
FDR	1	0.81	0.57	0.39	0.30	0.20
pFDR	1	0.80	0.53	0.38	0.30	0.20