GS01 0163
Analysis of Microarray Data

Keith Baggerly and Kevin Coombes
Section of Bioinformatics
Department of Biostatistics and Applied Mathematics
UT M. D. Anderson Cancer Center
kabagg@mdanderson.org
kcoombes@mdanderson.org

11 October 2007

Lecture 13: Differential Expression, Borrowing, and Modelling

- Testing Redux, and One More
- Comparing three or more groups
- Pairing
- Incorporating covariates
- Models

A Rehash

Comparing two groups:

- t-tests, Wilcoxon tests

Correcting for multiple testing:

- permutation tests
- Bonferroni, BUM and Empirical Bayes

Changing the question:

- Tail Ranks and Biomarkers

One More Difference Measure...

Still looking at one gene, and two groups of measurements for that gene

t-tests let us say “these are different”, but do not necessarily let us say anything about “how different are they?”

We can form confidence intervals corresponding for a given difference (eg, diff in log ratios) and convert that confidence interval into another interval on a scale that is more meaningful to us (such as fold change).

Combining CIs and Criteria

Now, there's a neat trick that can be used here by combining confidence intervals with the quantity of interest.

Our question till now has been "is this gene differentially expressed between the two groups?", but we can expand this to include another criterion by asking "is this gene differentially expressed between the two groups by at least a minimal amount k?"

The dChip Approach

For each group, assemble point estimates of the expression levels. These point estimates are assumed to have normal distributions. We can then form a confidence interval for the ratio, and we can focus our attention just on those genes where the *lower bound* of this confidence interval is more than k-fold. Thus, not only are we pretty sure that the gene is differentially expressed, but we believe that it is different by at least a minimal amount that we can specify.

Is this the way to go?

I don't necessarily think the dChip answers are right, because I think that their model has the wrong error structure, but I do think that the confidence interval idea has some merit.

It has the practical advantage of using more than one filtering criterion to assess "significance".

Applying Bonferroni requires setting a very wide confidence interval. Permutation tests still work.

Expanding our Focus

Say we have data from 3 groups that were run at the same time, as opposed to 2. Does this change the outcome of our initial comparison of two groups?

- Given microarray experiments on
  - N_A sample of type A
  - N_B sample of type B
  - N_C sample of type C
- Decide which of the G genes on the microarray are differentially expressed between groups A and B.

Expanding our Focus

The t-statistic from before

$$t = \frac{\bar{x}_B - \bar{x}_A}{s_P \sqrt{1/N_A + 1/N_B}}.$$

The numerator doesn't change, but what about the denominator?

The pooled estimate of the standard deviation initially includes data from just A and B, but it can be expanded to include data from all of the groups

The Broader Pool...

For two groups:

$$s_P^2 = \frac{(N_A - 1)s_A^2 + (N_B - 1)s_B^2}{N_A + N_B - 2}.$$

For three groups:

$$s_P^2 = \frac{(N_A - 1)s_A^2 + (N_B - 1)s_B^2 + (N_C - 1)s_C^2}{N_A + N_B + N_C - 3}.$$

What Does This Buy Us?

A more precise estimate of the variation gives us more degrees of freedom for the t-test.

More degrees of freedom gives us a more sensitive test.

Extreme case: $N_A = 2, N_B = 2, N_C = 10$.

How many differences do we see?

Some Simulations

No differences in the data...

```
n.genes <- 2000
an <- 2; bn <- 2; cn <- 10
n.samples <- an + bn + cn;
type <- factor(rep(c('A', 'B', 'C'),
                   times=c(an, bn, cn)))
data <- matrix(rnorm(n.genes*n.samples),
  nrow=n.genes)
am <- apply(data[, type=='A'], 1, mean)
bm <- apply(data[, type=='B'], 1, mean)
```

Some Simulations

```
av <- apply(data[, type=='A'], 1, var)
bv <- apply(data[, type=='B'], 1, var)
cv <- apply(data[, type=='C'], 1, var)

sp2.ab  <- ((an-1)*av + (bn-1)*bv)/
              (an+bn-2)
sp2.abc <- ((an-1)*av + (bn-1)*bv +
               (cn-1)*cv)/(an+bn+cn-3)
```

Some Simulations

```
t.stat.ab  <- (bm - am)/
  (sqrt(sp2.ab)*sqrt(1/an+1/bn))
t.stat.abc <- (bm - am)/
  (sqrt(sp2.abc)*sqrt(1/an+1/bn))

p.val.ab <- sapply(t.stat.ab, function(
  tv, df) {
   2*(1-pt(abs(tv), df))
  }, an + bn - 2)
p.val.abc <- sapply(t.stat.abc,...
   , an + bn +cn - 3)
```

p.val.abc ¡- sapply(t.stat.abc, function(tv, df) 2*(1-pt(abs(tv), df)) , an + bn +cn - 3)

What Differences Are There?

None.

Added variability makes it harder to see stuff that is there, but not easier to see stuff that isn't there.

The benefits associated with more precision are linked to increased sensitivity.

Introduce some Differences

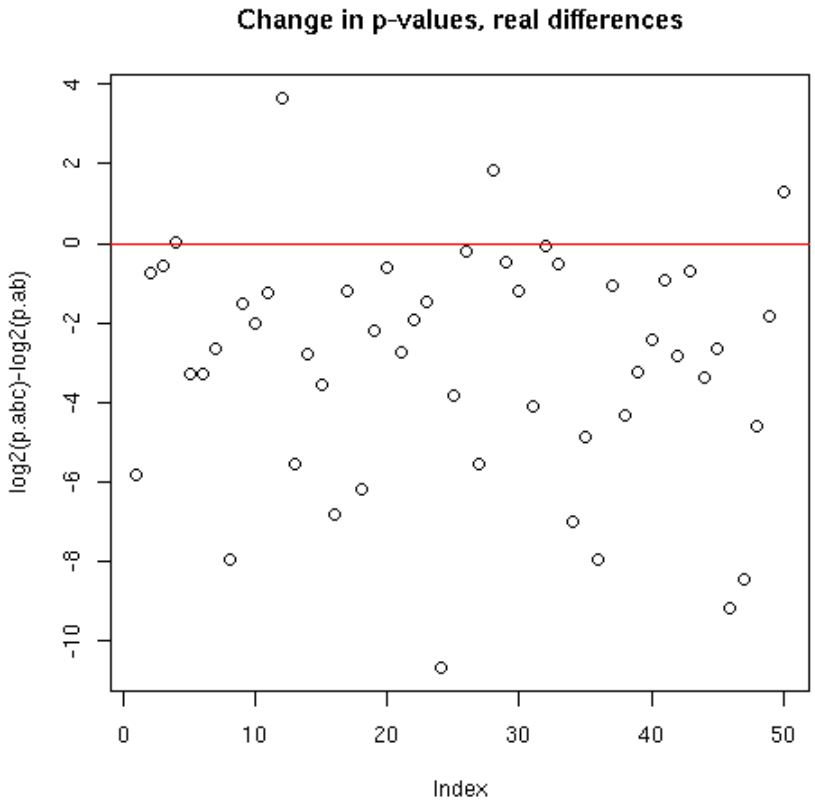
```
data[1:50,type=="A"] <-
    data[1:50,type=="A"] + 3;
```

recompute means, vars, t-values and p-values

```
sum(p.val.ab  < 0.01); # gives 19
sum(p.val.abc < 0.01); # gives 45
sum(p.val.ab[1:50]  < 0.01); # gives  2
sum(p.val.abc[1:50] < 0.01); # gives 21
```

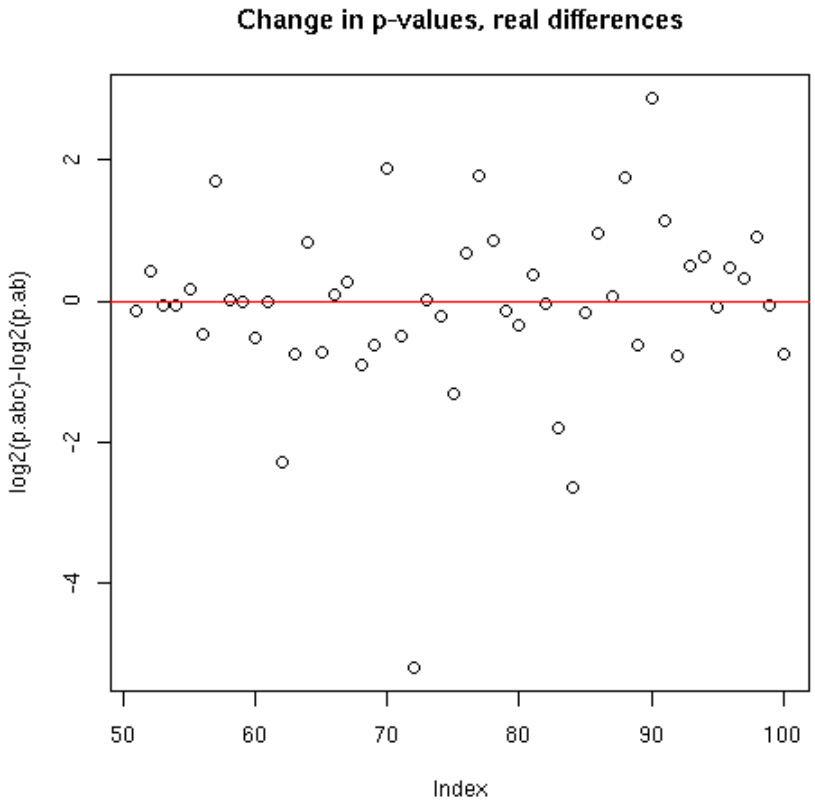
Plot P-Value Differences

```
plot(-log2(p.val.ab[1:50]) +
       log2(p.val.abc[1:50]), ... );
```



Plot P-Value Differences

```
plot(-log2(p.val.ab[51:100]) +
     log2(p.val.abc[51:100]), ... );
```



What Assumptions are We Making?

The variance structures do not change between the three groups (the means can be different).

We are already making this assumption implicitly with the two-sample t-test.

This assumption means that I would restrict the other groups used to those run about the same time, with the same chip lot, etc.

That the data looks approximately normal (work on the log scale).

Corrections

Rank tests also work.

Bonferroni still works just fine.

BUM still works just fine.

Empirical Bayes still works just fine.

Permutations?

Permute residuals from the null model

Another Extension: Chip Lot?

Say we have data from arrays from two different lots, 1 and 2, and that we have samples from groups A and B run on arrays from both lots. How should we look at this?

Well, we can still use a two-sample t-test (assuming run order was randomized), but this might break if there are big differences between lots.

(I'll assume for now that the number of samples in each group/lot combination is the same).

Some more Simulations

```
n.genes <- 2000
a1n <- 2;  b1n <- 2
a2n <- 2;  b2n <- 2
an <- a1n + a2n;  bn <- b1n + b2n;
n.samples <- an + bn;
type <- factor(rep(c('A', 'B'),
     times=c(a1n + a2n, b1n + b2n)))
group <- factor(c(rep(c('G1', 'G2'),
                      times=c(a1n,a2n)),
                      rep(c('G1', 'G2'),
                      times=c(b1n,b2n))));
```

Add Some Big Differences

```
data <- matrix(rnorm(n.genes*n.samples)
  nrow=n.genes)

data[,group=="G2"] <-
    data[,group=="G2"] + 8;

data[1:50,type=="A"] <-
    data[1:50,type=="A"] + 4;
```

Is this realistic? Can groups overshadow types?

How do we fit both type and group?

Start with an overall mean

measure deviations associated with type

measure deviations associated with group

```
mu <- apply(data,1,mean);
delta.type  <- apply(
    data[,type=="A"]-mu,1,mean);
delta.group <- apply(
    data[,group=="G1"]-mu,1,mean);
```

How do we fit both type and group?

fit the data, and sum the squared residuals

```
our.fit <- data;
our.fit[,type=="A" & group=="G1"] <-
  mu + delta.type + delta.group;
our.fit[,type=="A" & group=="G2"] <-
  mu + delta.type - delta.group;
our.fit[,type=="B" & group=="G1"] <-
  mu - delta.type + delta.group;
our.fit[,type=="B" & group=="G2"] <-
  mu - delta.type - delta.group;
```

Some numbers

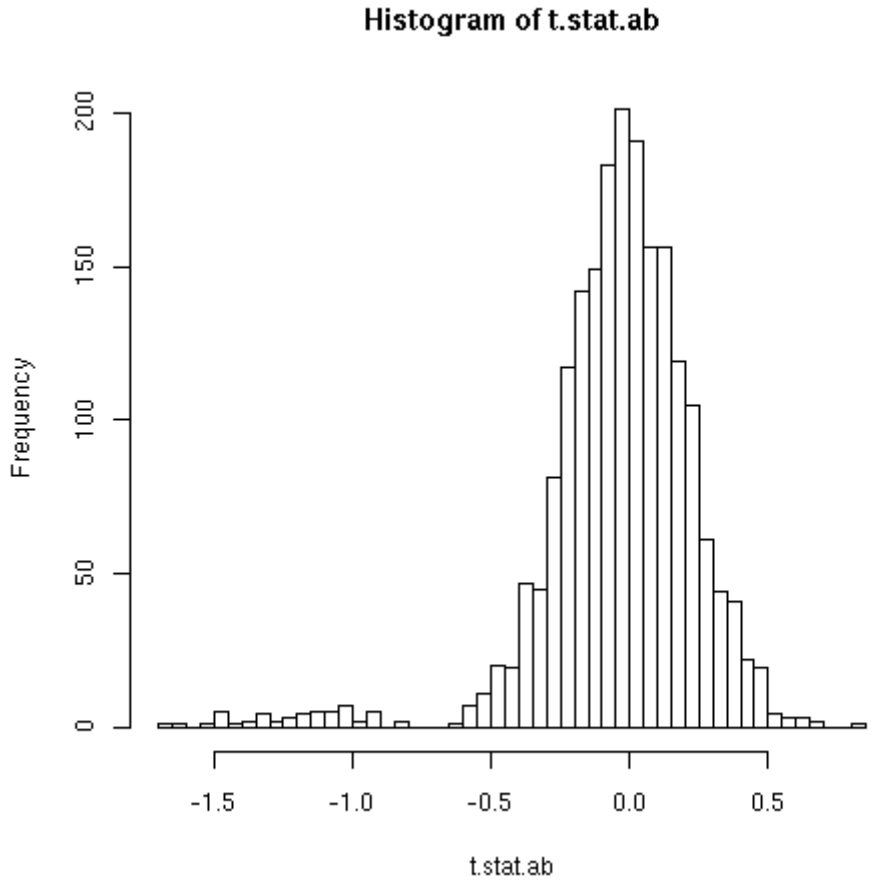
```
> our.resid <- data - our.fit;
> our.se <- sqrt(apply(our.resid^2,
                       1, sum)/5);
> data[1,]
[1]  3.81  3.59 11.11 12.54
[5] -0.67 -0.17  7.05  8.95
> mu[1]
[1] 5.78
> delta.type[1]
[1] 1.99
> delta.group[1]
[1] -4.14
```

Some numbers

```
> our.fit[1,]
[1]  3.63  3.63 11.90 11.90
[5] -0.35 -0.35  7.93  7.93
> our.resid[1,]
[1]  0.18 -0.04 -0.79 0.64
[5] -0.32  0.17 -0.88 1.03
> our.se[1]
[1] 0.79
our.t.type <- delta.type[1]/
              (our.se[1]/sqrt(8));
```

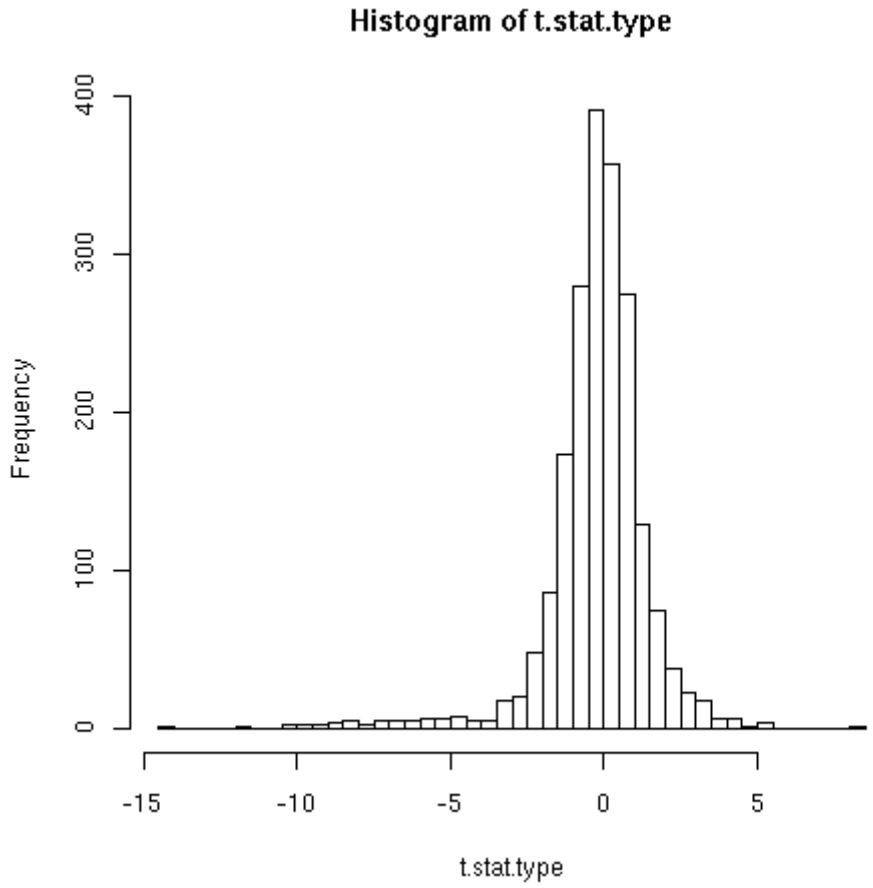
What do the t-stats look like?

```
hist(t.stat.ab,breaks=50);
```



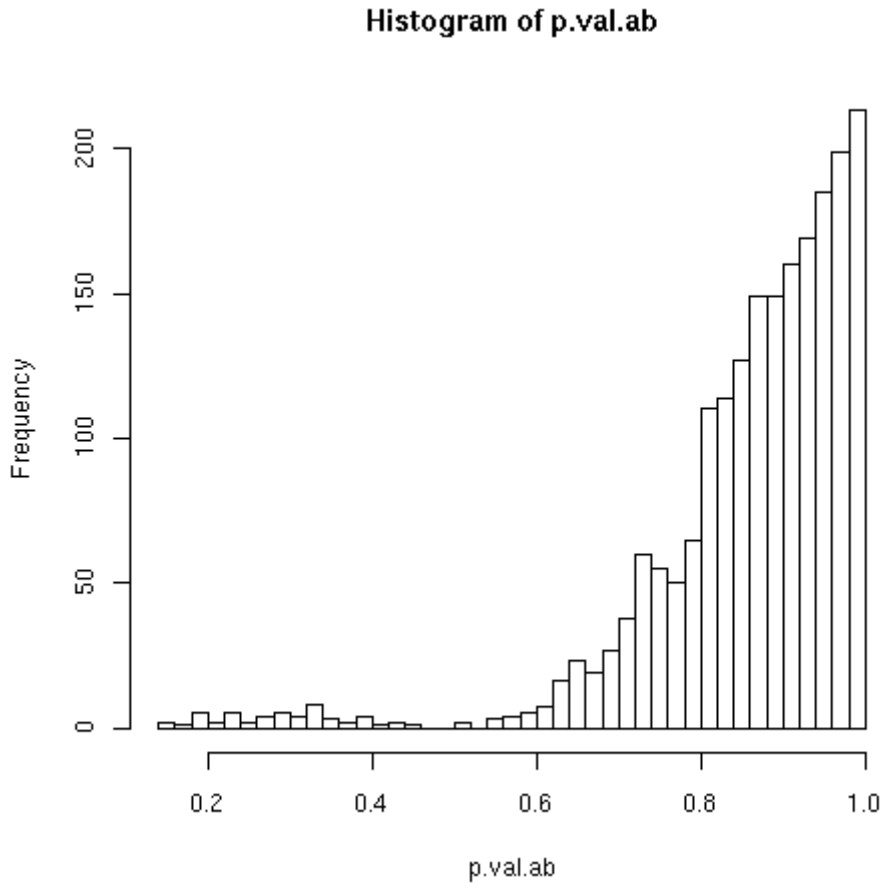
What do the t-stats look like?

```
hist(t.stat.type,breaks=50);
```



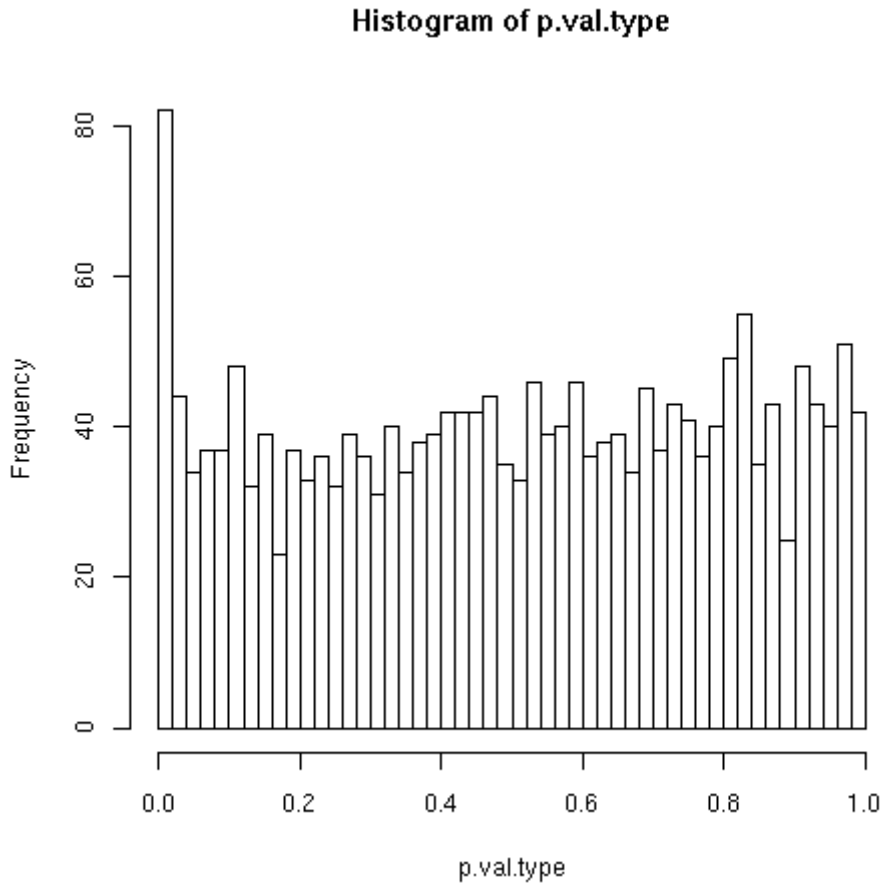
What do the p-values look like?

```
hist(p.val.ab,breaks=50);
```



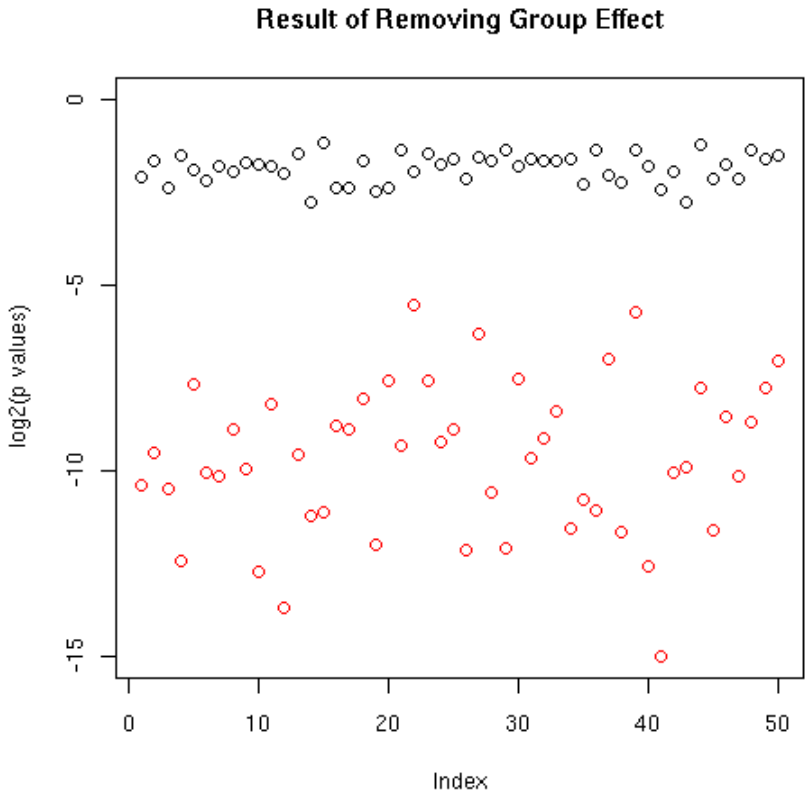
What do the p-values look like?

```
hist(p.val.type,breaks=50);
```



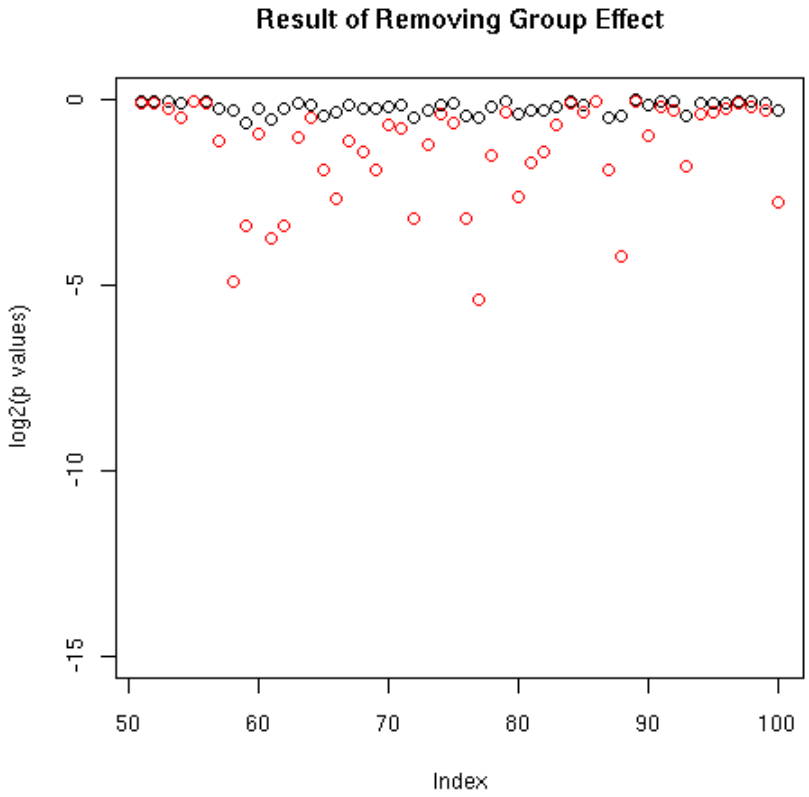
Changes When Different

```
plot(log2(p.val.ab[1:50]),ylim=c(-15,0),...);
points(log2(p.val.type[1:50]),col='red');
```



Changes When Different

```
plot(c(51:100),log2(p.val.ab[1:50]),...);
points(c(51:100),log2(p.val.type[1:50]),...);
```



Partitioning Variance: ANOVA

This general procedure of apportioning the observed variation to the effects that gave rise to it is known as the Analysis of Variance (ANOVA). It was introduced by R.A. Fisher in the 1920s.

Using other groups to stabilize the variance may not be that big a deal. Splitting off variation due to external causes before assessing our effect of interest can be vital.

ANOVA in R

```
our.lm.1 <- lm(data[1,] ~ type + group);
our.anova.1 <- anova(our.lm.1);
our.anova.1
Analysis of Variance Table

Response: data[1, ]
          Df  Sum Sq Mean Sq F value
type       1  31.557  31.557  52.008
                Pr(>F) 0.000799 ***
group      1 136.818 136.818 225.487
                Pr(>F) 2.372e-05 ***
Residuals  5   3.034   0.607
```

Replacing Data with Ranks

```
our.lm.1 <- lm(rank(data[1,]) ~ ...);
our.anova.1 <- anova(our.lm.1);
our.anova.1
Analysis of Variance Table

Response: rank(data[1, ])
          Df Sum Sq Mean Sq F value
type       1    8.0     8.0      20
                Pr(>F) 0.0065663 **
group      1   32.0    32.0      80
                Pr(>F) 0.0002911 ***
Residuals  5    2.0     0.4
```

Is this Legit?

Not really. The rank theory is based on shuffling just those values around, and requires somewhat larger sample sizes to get close to “normal”.

The more standard rank test for any difference at all here is the Kruskal-Wallis test. The p-values for this test are computed by permuting ranks and counting the possible sums.

Kruskal-Wallis Results

```
our.kw <- kruskal.test(rank(data[1,]) ~
                         type + group)
Kruskal-Wallis rank sum test

data:  rank(data[1, ]) by type by group
Kruskal-Wallis
   chi-squared = 1.3333, df = 1,
   p-value = 0.2482
```

Not enough samples here, so we need some assumptions.

An Extreme Case: Pairing

In many cases, we have data that are paired: treated/untreated, before/after, primary/metastasis (same patient), or case/control studies matched on a variety of factors.

In this case the math simplifies rather considerably, and we can use a simple one-sample t-test applied to the paired differences:

$$\frac{\bar{x}_A - \bar{x}_B}{sqrt(var(data[A] - data[B])/(n_A - 1))}$$

The Rank Equivalent: Signed Rank Tests

As with the ANOVA table discussed above, the paired t-test also has a rank analog, arrived at by ranking the differences and applying a sign as A is greater than B or vice-versa. The sum of the positive ranks gives the test statistic.

```
wilcox.test(data[A],data[B],paired=TRUE);
```

Three Groups, Two Lots?

What if we have both scenarios at once?

Multiple groups, and known external factors?

What is the general rule?

Including Covariates

The general extension of ANOVA is supplied by the linear model and regression. This was actually used above:

```
our.lm.1 <- lm(data[1,] ~ type + group);
```

where we are fitting the response (data[1,]) as a function of the covariates at hand (type and group). The final significance value is that associated with the effect of interest in the full model.

Some Standard Factors

What things might we include as explanatory covariates?

- chip lot
- chip
- dye
- run date/order

The Broader Theme: Modelling

If we know that effects other than the ones we're interested in are likely to be present, it is generally worthwhile to recast our test to explicitly incorporate (and hopefully factor out) these other effects.

This is the idea of modelling the data.

Of course, we can't model everything. When we can't model it, randomize to balance it!

Differential Expression and Borrowing

- Modelling Redux
- Borrowing Strength Across Genes
- Combining Borrowing with Ranks
- Borrowing Tail Ranks
- Conditioning on Biology

The Modelling Punchline

Incorporating external information can help sharpen our inferences.

Incorporating such information often goes by the name of modelling, but it can also be viewed as “conditioning on relevant subsets of information”.

The crux of the problem is defining precisely what constitutes a “relevant subset”, which includes what we mean by “relevant”.

Types of Conditioning

One of the more common types of conditioning is to assume that some other quantity being measured shares some distributional characteristics with measurements of the quantity of interest.

In shorter words, we can use other data to give us better estimates of standard deviations, or the shape of the distribution, or so on. We saw this earlier with the use of a third group of microarray measurements to sharpen inferences about differences between the first two.

Are Other Genes Relevant?

Are there similar characteristics to microarray measurements of different genes?

If there are, how can we use them?

Most frequently, the answer to the first question is assumed to be yes based on empirical observations. Occasionally, a modelling of the underlying physical processes can further suggest the nature of the similarity.

An Example

Our first example: normalization.

This can assume either that “most genes don’t change” (single scaling factor normalization) or, more stringently, that “the quantiles of the intensity distributions should be about the same” (loess normalization).

Are the Assumptions Valid Here?

In general, yes. In checking normalization methods, people have produced some nice-looking smooth curves, but the latter in particular are working under the assumption that if we start with genes of the same rough level of expression, the distributions of values when nothing is going on will be about the same.

Other Extensions of Borrowing

borrowing strength on the p-value scale.

BUM

Empirical Bayes

Extending this idea to diff. e.

What can we do here?

Say that we have our standard question of trying to compare the levels of a given gene in two different groups, A and B.

How can we change the t statistic?

As before, our best guess about the central value of the gene in each of the groups is driven by the observed values for that gene:

$\bar{x}_A - \bar{x}_B$ is unchanged.

Pooling variance estimates

What can use to improve our estimate of the variance?

How about the variance of all of the genes?

This is likely to be too much.

What if we just use the genes that are close by in terms of overall (average) intensity?

This type of procedure makes some of the same underlying assumptions as the loess normalization, which also works with “locally similar” data.

What does this produce?

a stabilized variance and a “smooth” t-test.

This idea has been independently reintroduced in several forms.

Baldi and Long (2001) use a Bayesian approach to trade off between the sample variance for the gene of interest and the pooled variance estimate. This is known as a “shrinkage” estimate.

Newton et al (2001) use a Gamma-Poisson model which achieves the same effect.

Some More Papers

The “fudge factor” in the denominator of SAM is of this variety.

Baggerly et al (2001) use a Beta-binomial model based on the use of variance derived from replicate spottings to derive the the locally pooled variance estimate; there is no weighting tradeoff with the actual variance observed.

Are We Using It?

This last paper is the basis for some of the "standard analyses" done at MD Anderson. All of the above tests were developed in the context of cDNA microarrays.

We've also used it to analyze data from nylon membrane arrays (Coombes, 2001).

Why might the assumption be valid here?

There are plausible reasons why the variance of microarray readings should change in a smooth fashion as the overall intensity increases.

These have to do with lognormal expression values, background subtraction, and thresholding.

But we're implicitly assuming that "most genes aren't too correlated" so a variance estimate derived from several genes will be close.

Implications of Independence

We note that this assumption of independence means that in terms of trying to define the overall variance distribution, it is not a good idea to choose a bunch of genes known to be biologically related as our relevant subset. It is interesting to explore these connections, but here we are seeking reinforcement of a story by looking for groups of genes having similar expression patterns.

Does this help a great deal?

In our earlier discussions, we noted that better characterization of the variability did improve things, but maybe not so much.

However, that assessment was predicated on our having a good idea of the underlying distribution to begin with. If the data are skewed or subject to frequent outliers, things can get worse.

Skewness, Outliers, and Bears?

This, of course, is why we often shift to rank tests which don't depend on the particular shape of the distribution. But, as we saw last time, the discrete nature of the ranks may preclude a rank test from yielding a small p-value even when something extreme is going on.

Linking Borrowing and Ranks

Small p-values, however, can be obtained if we have more "effective samples" with which to characterize the underlying distribution, leading us to combine the idea of borrowing strength across genes with the idea of using ranks to remain less sensitive to the particular shape of the underlying distribution.

The Relative Rank Test

Oddly enough, we haven't seen that much written about borrowing with ranks, but here goes.

Assume that we are interested in deciding if the levels of gene g are different between two groups A and B, and that g is for the most part contained within a set of genes G having similar null distributions.

The standard procedure (Wilcoxon) is to rank the $n_A + n_B$ values of g and sum the ranks of those in one of the groups (say A). The p-values are then computed by permutation and counting arguments.

The Relative Rank Test

For Wilcoxon, we note that we could just as easily have worked with the difference in average ranks for A and B, respectively, as the total must stay fixed.

Here, we rank all $G * (n_A + n_B)$ expression values within the "relevant set" G, and focus on the difference between the average ranks for gene g in groups A and B. Here, the choice of just one sum (say the A ranks) or the difference does matter because there are intervening values present.

What does this potentially buy us?

The ability to get small p-values

The ability to get large p-values

The ability to differentiate between “extreme cases”

Some robustness against outliers (we lose some of this relative to the Wilcoxon test, however) or different distributions

What does this potentially cost us?

accuracy, if the distributions are starkly different (eg, including high real variability genes with low real variability genes).

The traditional borrowing of strength focuses on a single number (such as the variance) and presumes that will be stable. Rank sharing makes stronger distributional assumptions.

Some Math

What can we say about the distribution of the difference $\bar{r}_A - \bar{r}_B$?

Well, if G is large, then we can effectively ignore the discrete nature of the rank distribution. To make things easier (on me), let's divide the ranks by $G * (n_A + n_B)$ so that we have values ranging from 0 to 1.

Some Math

When nothing is going on, the expected difference in average ranks is 0. The variance of a single draw from a uniform distribution is 1/12, so the variance of the difference is

$$\frac{1}{12}\left(\frac{1}{n_A}+\frac{1}{n_B}\right).$$

Approximate normality kicks in fairly quickly, and for finite samples the shape involves the repeated convolution of uniforms (giving B-splines).

Some Outcomes

So, what do the results of using this test look like?

Looking at the prostate cancer data, the values returned by the relative rank test look intermediate between those of Wilcoxon and t-tests. By using multiple genes to more finely partition the ranks, we recapture some of the parametric sensitivity of the t-test. Here, the data were approximately normal to begin with.

Relative Tails?

Can the relative rank approach be used to help with the tail rank test for biomarkers?

Well, in the description of the tail rank test given earlier, it was stated that we needed to specify two things before using the test:

ψ, the desired specificity of the biomarker, and

γ, the bound on the FWER.

The way that the relative rank approach can help is hidden in the way the value of ψ is used.

Defining Quantiles

Specifically, in order to use the tail rank statistic we need to estimate, for each gene g, a threshold value τ_g such that $P(X_g < \tau_g) = \psi$

The difficulty is that τ_g represents a tail quantile of a distribution, because we want ψ to be close to 1. Tail quantiles are hard to estimate well unless (a) you have lots of samples (which we typically won't) or (b) you have some knowledge of the parametric form of the distribution of X_g.

Tradeoffs

The question then becomes one of which assumption is more plausible:

that we know a parametric form well enough to characterize tails,

or

that the distribution of expression values in a given intensity range when nothing is going on may be similar enough across genes for them to be productively combined.

The Upside

If we collect the ranks for G genes as above, and focus on the results in the control samples, then our "effective sample size" will increase, typically to the point where we can get point estimates of some extreme quantiles (such as 99%).

Further Extensions

What other ways can we use the relative rank approach?

Kruskal-Wallis can be revisited.

Is there some way to build sensitivity into the tail rank procedure? Probably not, since we're assuming that the behavior of the biomarker is "atypical" for the subset that it flags as interesting.

Sensitivity and Biomarkers

There is an asymmetry here, which reflects the asymmetry in the question we're asking.

For good biomarkers, we want the specificity to be high, but we're willing to live with low sensitivity.

The rationale for this is that the heterogeneity of the disease suggests that if markers are to be productively used, this should be as part of a panel.

We don't yet know how to assemble a good and parsimonious panel.

We may be able to assemble a good panel.

Conditioning on Biology

Above, we've tended to group probes as "similar" based on their observed expression values, giving intensity-dependent variance estimates, normalization, and so on.

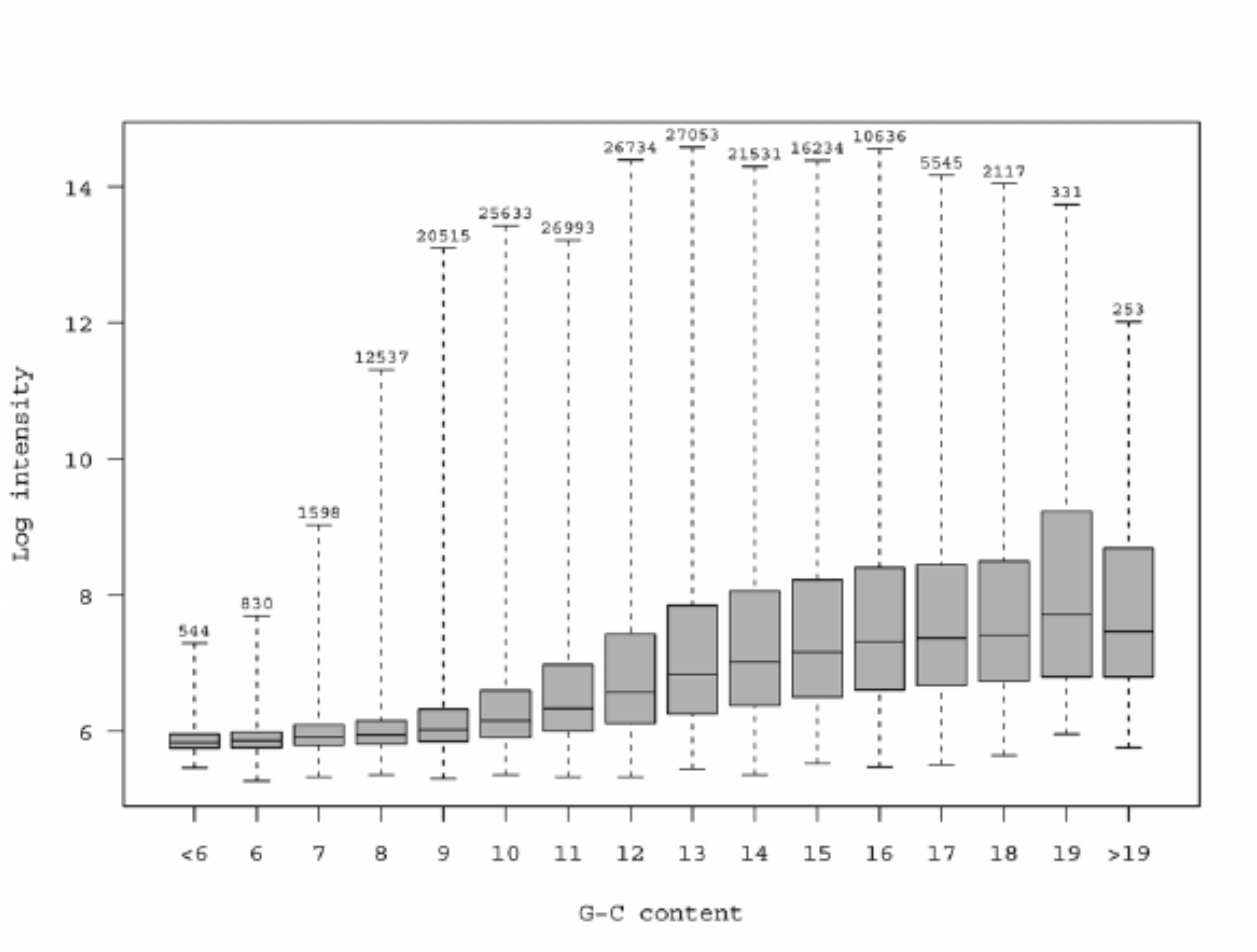
Are there other ways of gathering probes or probesets into groups that we might expect to have similar behavior with respect to baseline brightness and variability?

Exploiting Sequence: PDNN and GCRMA

Some of the more effective methods work by grouping probes in part based on their sequence.

What properties of a probe sequence might make it better at binding, or make the bonds it does produce stronger?

Intensity by GC Count



G-C has 3 hydrogen bonds, A-T has 2

Defining Affinity

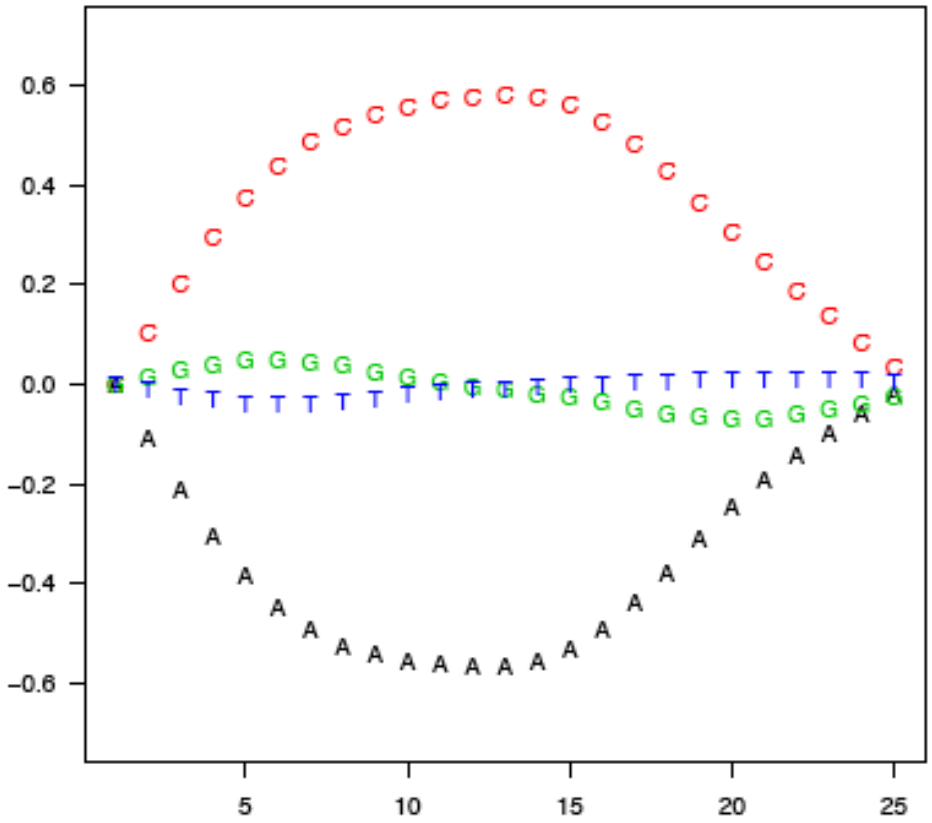
A model for intensity

$$log(PM_{ij}) = \Theta_i + \alpha_j + \epsilon_{ij}$$

A formula for affinity (Naef and Magnasco, 03, Phys Rev E v68)

$$\alpha = \sum_{k=1}^{25} \sum_{j \in \{A,T,G,C\}} \mu_{jk} 1_{b_k=j}, \quad \mu_{jk} = \sum_{m=0}^{3} \beta_{jm} x^m$$

Probe Affinity by Position



Why the asymmetry?

Invoking GCRMA in R

uses some information from the MM values! Follow outline in BioC...

```
library(gcrma)
library(hgu95av2probe)
library(hgu95av2cdf)
library(affydata)
data(Dilution)
ai <- compute.affinities(cdfName(Dilution))
Dil.expr <- gcrma(Dilution,affinity.info=ai)
```