# Synthesis of Microarray Experiments

Robert Gentleman and Deepayan Sarkar

January 12, 2007

With many different investigators studying the same disease and with a strong commitment to publish supporting data in the scientific community, there are often many different datasets available for any given disease. Hence there is interest in finding methods for combining these datasets to provide better and more detailed understanding of the underlying biology. In this tutorial we will briefly cover $p$-value based approaches to combining multiple studies, and then move on to more general methods which are usually more appropriate for microarray studies.

## Combining $p$-values: an artificial example

We start with an artificial example to illustrate voting and $p$-value based methods. The following code generates some random data that we can work with:

```
> k <- 7   # number of experiments
> n <- 10  # number of samples in each group
> mu1 <- 0; mu2 <- 1; sigma <- 2.5
> x <- matrix(rnorm(k * n, mean = mu1, sd = sigma), n, k)
> y <- matrix(rnorm(k * n, mean = mu2, sd = sigma), n, k)
```

Each paired column in `x` and `y` represent samples from one study. We can perform a $t$-test for the first experiment as follows:

```
> t.test(x[, 1], y[, 1])

        Welch Two Sample t-test

data:  x[, 1] and y[, 1]
t = -0.9671, df = 16.707, p-value = 0.3473
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -2.904582  1.080411
sample estimates:
mean of x mean of y
0.3059694 1.2180550
```

The $p$-values for all `k` experiments (along with direction) can be obtained as a vector:

```
> pvals <- sapply(1:k, function(i) t.test(x[, i], y[, i])$p.value)
> pvals.less <-
      sapply(1:k,
             function(i) t.test(x[, i], y[, i],
                                alternative = "less")$p.value)
> direction <- sapply(1:k, function(i) sign(mean(x[, i]) - mean(y[, i])))
> pvals.less

[1] 0.17364742 0.17237995 0.01836840 0.01152889 0.02551708 0.20729906 0.29754607

> direction

[1] -1 -1 -1 -1 -1 -1 -1
```

The number of significant votes (out of 7) at level 0.05:

```
> sum(pvals.less < 0.05)

[1] 3
```

**Exercise 1** *What do you conclude about the effect? Would it make sense to use* `pvals` *instead of* `pvals.less`*? Repeat the process a few times (regenerating the random numbers).*

Let random variables $U_1, \ldots, U_K$ represent $K$ $p$-values. A new statistic derived from these is

$$U = U_1 \ldots U_K$$

which we expect to be small (i.e. smaller than it would be by chance) when there is a weak signal. We can perform a consensus test if we knew the null distribution of $U$. Here are a couple of facts from probability theory:

- If $X$ has a $\mathcal{U}(0,1)$ distribution, the $-\log(X)$ has an exponential distribution with rate parameter 1.
- If $Y_1, \ldots, Y_n$ are independent exponentials with rate 1, then $\sum Y_i$ has a Gamma distribution with rate 1 and shape parameter $n$

Since under the null of no difference each $U_i \sim \mathcal{U}(0,1)$, $-\log(U) = \sum -\log(U_i) \sim \mathcal{G}amma(1,k)$.

**Exercise 2** *Compute the consensus p-value $P(-\log(U) > -\log(u))$, where $u$ is the product of the observed p-values. Repeat with different parameters.*

See `R` file for solution. In our example, the $p$-value was

```
[1] 0.0007784023
```

# Microarray studies

The usual application of meta-analysis is to analyze a single outcome, or finding, using published data where typically only summary statistics are available. With microarray experiments, we are often in the more fortuitous situation of having the complete set of primary data available, not just the summary statistics. By phrasing the synthesis in terms of standard statistical models, many of the recently developed $p$-value adjustment methods for multiple comparisons can be applied directly.

## The experimental data

We will use three data sets as examples. One is a study of breast cancer reported by West et al. (2001) in which 46 patients were assayed and two phenotypic conditions were made public, the estrogen receptor (ER) status and the lymph node (LN) status. We will refer to this as the Nevins data in the remainder of the text. The samples were arrayed on Affymetrix HuGeneFL GeneChips. ER status was determined by immunohistochemistry and later by a protein immunoblotting assay. We have used 46 samples, of which 4 gave conflicting evidence of ER status depending on the test used. Lymph node status was determined at the time of diagnosis. Tumors were reported as negative when no positive lymph nodes were discovered and as positive when at least three identifiably positive lymph nodes were detected.

A second breast cancer data set was made public by van't Veer et al. (2002) in which tumors from 116 patients were assayed on Hu25K long oligomer arrays. Among other covariates the authors published the ER status of the tumors. Their criterion was a negative immunohistochemistry staining, a sample was deemed negative if fewer than 10% of the nuclei showed staining and positive otherwise. We refer to this as the van't Veer data.

The third experiment is one published by Roepman et al. (2005), which assayed patients with primary head and neck squamous cell carcinoma using long oligomer arrays. Lymph node status of the individuals involved was determined by clinical examination followed by computed tomography and/or magnetic resonance imaging. Any nodes that were suspected of having metastatic involvement were aspirated and a patient was classified as lymph node positive if the aspirate yielded any metastatic tumor cells. We refer to this as the Holstege data.

In our first example, the goal is to combine the two breast cancer data sets that report on the estrogen receptor (ER) status. In the second comparison, we combine the Holstege data and the Nevins data on the basis of LN status.

Some of the issues that arise in combining experiments can already be seen. For the comparison on the basis of ER status we see that the two used similar, but different methods for assessing ER status. One might want to revert the Nevins data to the classifications based only on immunohistochemistry staining to increase comparability across the two experiments. This is likely to come at a loss of sensitivity since one presumes that the ultimate (and in four cases different) classification of samples was the correct one.

For the synthesis of experiments on the basis of lymph node status the situation is even more problematic. One might wonder whether approximately the same effort was

expended in determining lymph node status in the two experiments. The value of any synthesis of experiments will have a substantial dependency on the comparability of the patient classifications. If the classifications of samples across experiments are quite different then it is unlikely that the outputs will be scientifically relevant.

The data are available in the compendium package GeneMetaEx.

```
> library("nlme")
> library("GeneMeta")
> library("GeneMetaEx")
> data("NevinsER")
> data("VantER")
> data("NevinsLN")
> data("HolstegeLN")
> ## test to make sure that we have matching data
> stopifnot(all(featureNames(VantER) == featureNames(NevinsER)))
> stopifnot(all(featureNames(NevinsLN) == featureNames(HolstegeLN)))
```

One problem that must be dealt with when combining experiments is the matching problem. For this data, probes were matched on the basis of GenBank or UniGene identifiers. For the Nevins – van't Veer synthesis we have 3988 mRNA targets in common, while for the Nevins – Holstege synthesis there are 3786 common mRNA targets.

## Effect size models

In situations where potentially different scales of measurement have been used it will be necessary to estimate an index of effect magnitude that does not depend on the scaling or units of the variable used. For two-sample problems the scale-free index that is commonly used is the so-called *effect size*, which is the difference in means divided by the pooled estimate of standard deviation (note that this is not the $t$-statistic, which would use the standard error of the mean difference). Other measures include the correlation coefficient and the log odds ratio, but we do not consider them here. Choi et al. (2003) has proposed the use of meta-analytic tools for combining microarray experiments and argued in favor of synthesis on the basis of estimated effects. The `zScores` function from the GeneMeta package can be used to compute various per experiment and combined summaries:

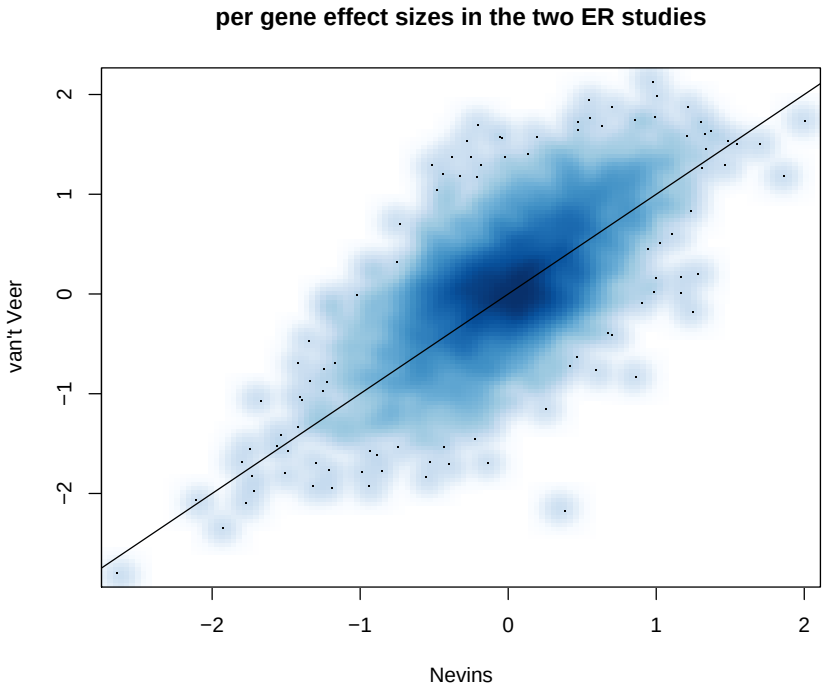
```
> eSER <- list(NevinsER, VantER)
> eCER <- list(NevinsER$ERstatus, VantER$ERstatus)
> eCER <- lapply(eCER, function(x) ifelse(x == "pos", 1, 0))
> wSFEM <- zScores(eSER, eCER, useREM=FALSE)
> wSREM <- zScores(eSER, eCER, useREM=TRUE)
```

See the help page `?zScores` for the meaning of the columns.

```
> t(head(wSFEM, 4))
```

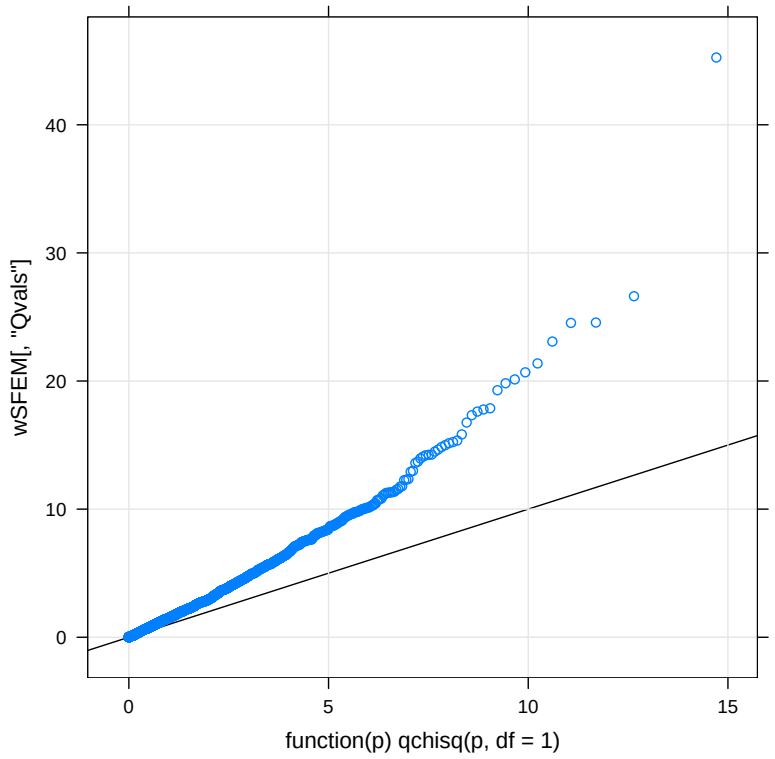
| | NM_003000 | NM_003001 | NM_003003 | NM_003004 |
|---|---|---|---|---|
| zSco_Ex_1 | -0.60545406 | -0.70196788 | 2.41550716 | 1.568831e+00 |
| zSco_Ex_2 | 0.05709760 | 0.09268415 | 1.15929745 | 3.624441e+00 |
| zSco | -0.27918882 | -0.30121285 | 2.26300749 | 3.892129e+00 |
| MUvals | -0.04458474 | -0.04811191 | 0.36534123 | 6.366083e-01 |
| MUsds | 0.15969387 | 0.15972729 | 0.16144057 | 1.635630e-01 |
| Qvals | 0.29188835 | 0.41062008 | 2.05744250 | 4.491378e-01 |
| df | 1.00000000 | 1.00000000 | 1.00000000 | 1.000000e+00 |
| Qpvalues | 0.58901296 | 0.52165495 | 0.15146420 | 5.027447e-01 |
| Chisq | 0.78009992 | 0.76325219 | 0.02363523 | 9.936852e-05 |
| Effect_Ex_1 | -0.17889542 | -0.20755562 | 0.73601566 | 4.689381e-01 |
| Effect_Ex_2 | 0.01083732 | 0.01759197 | 0.22067723 | 7.082711e-01 |
| EffectVar_Ex_1 | 0.08730439 | 0.08742478 | 0.09284477 | 8.934677e-02 |
| EffectVar_Ex_2 | 0.03602535 | 0.03602618 | 0.03623475 | 3.818712e-02 |

**Exercise 3** *Are the effect sizes similar in the two studies? Would you expect them to be? Code to produce the plot below is given in the* `R` *file. How do you interpret this plot? Repeat this process with the lymph node data sets* `NevinsLN` *and* `HolstegeLN`*. Do you get similar results?*



There are two candidate models, with and without random effects for experiments, to combine the effects for each gene. The usual procedure is to first assess which of the two models is appropriate and to then subsequently fit that model. This determination is often based on Cochran's $Q$ statistic, if the value of this statistic is large then the hypothesis that the per-study measured effects are homogeneous is rejected and a random effects model is needed. The value returned by the `zScores` function includes the values of $Q$. Below we plot a Q-Q plot comparing these $Q$ values to the reference $\chi^2_1$ distribution:

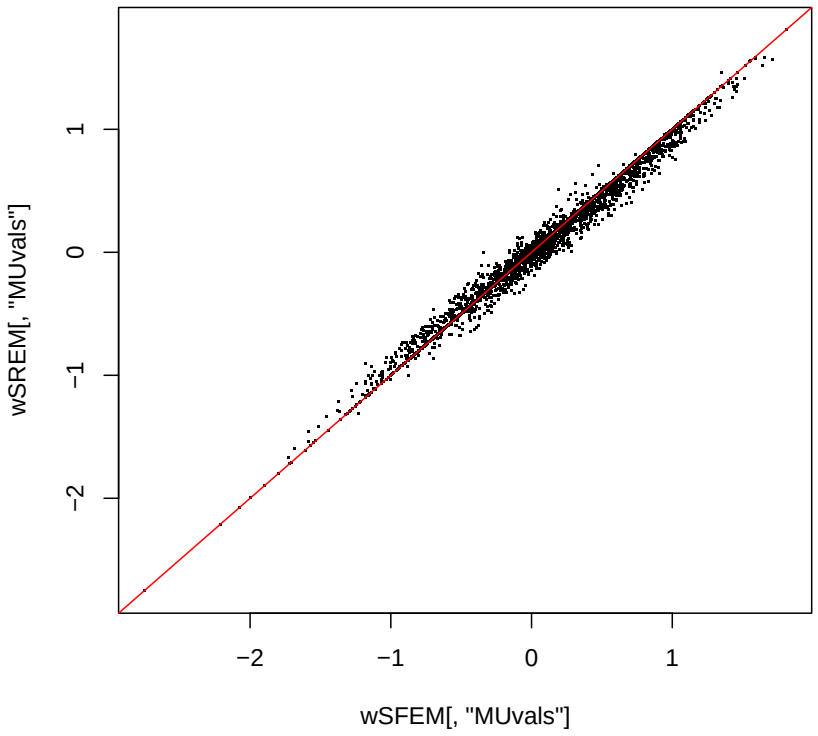
```
> library("lattice")
> plot(qqmath(wSFEM[, "Qvals"],
              type = c("p", "g"),
              distribution = function(p) qchisq(p, df = 1),
              panel = function(...) {
                  panel.abline(0, 1)
                  panel.qqmath(...)
              }))
```




**Exercise 4** *There appears to be a substantial deviation — the observed values are too large. Can we conclude that the random effect model (REM) is required for all genes? Or should we use it only for the ones with a sufficiently low p-value? Does it really matter? Reproduce this plot for the LN data.*

In the following graphic, we plot the estimated effect sizes using the two methods:

```
> plot(wSFEM[, "MUvals"], wSREM[, "MUvals"], pch = ".")
> abline(0, 1, col = "red")
```




**Exercise 5** *This suggests that the estimated combined effect size is not particularly affected by the choice of model. Is the same true for the standardized z-scores? (Hint: replace* `"MUvals"` *in the code with the relevant column name.)*

# Modeling individual observations

We next consider a formal random effects model for each gene comparison. We note that in general the different genes are not independent and hence a *gene at a time* approach will not be optimal. However, in the absence of any knowledge about which genes are correlated with which other genes it is not clear how to approach a genuinely multivariate analysis. Here we describe the gene-at-a-time approach. When the raw data is available, this is the recommended approach as it is easily generalizable to more complex situations.

Following Cox and Solomon (2003) we write the model for each gene as:

$$Y_{tjs} = \beta_0 + \beta_t + b_j + \xi_{jt} + \epsilon_{tjs}, \tag{1}$$

where $Y_{tjs}$ represents the expression value for the $s^{\text{th}}$ sample in the $j^{\text{th}}$ experiment, which is on treatment $t$. Note that we use the term *treatment* interchangeably with what would be called the disease condition or phenotype in the current application. $\beta_0$ is the overall mean expression, $\beta_t$ is the effect for the $t^{\text{th}}$ treatment, $b_j$ is a random effect characterizing the $j^{\text{th}}$ experiment, $\xi_{jt}$ is a random effect characterizing the treatment by experiment interaction. We assume that the $b_j$ have mean zero and variance $\tau_b$, that the $\xi_{jt}$ have mean zero and variance $\tau_\xi$, and that $\epsilon_{tjs}$ are random variables with mean zero and variance $\tau_\epsilon$ that represent the internal variability. The `lme` function in the nlme package can be used to fit such models.

## Constructing per-gene data frames

Since `lme` is not designed to work with expression sets directly, one needs to construct suitable data frames for each gene. It is useful to write a function that creates such data frames, e.g.

```
> makeDf <- function(i,
                     expr1, expr2, cov1, cov2,
                     experiment.names = c("1", "2")) {
      y <- c(expr1[i, ], expr2[i, ]) # i-th gene
      treatment <- c(as.character(cov1), as.character(cov2))
      experiment <- rep(experiment.names, c(length(cov1), length(cov2)))
      ans <- data.frame(y = y, treatment = factor(treatment),
                        experiment = factor(experiment))
      rownames(ans) <- NULL
      ans
  }
> makeDf.ER <- function(i) {
      makeDf(i,
             exprs(NevinsER), exprs(VantER),
             NevinsER$ERstatus, VantER$ERstatus,
             experiment.names = c("Nevins", "Vant"))
  }
> df3 <- makeDf.ER(3)
> summary(df3)
```

```
       y            treatment  experiment
 Min.   :-0.26300   neg:69    Nevins: 46
 1st Qu.:-0.01775   pos:93    Vant  :116
 Median : 0.07100
 Mean   : 2.52442
 3rd Qu.: 8.58171
 Max.   : 9.43015
```

This is not a particularly useful function, as it can only combine two studies at a time. Here is a more general version that we use below.

```
> makeDf2 <-
      function(i, ...)
  {
      args <- list(...)
      exprlist <- lapply(args, function(x) x$exprs[i, ])
      covlist <- lapply(args, function(x) as.character(x$cov))
      nsamples <- sapply(covlist, length) # number of samples
      experiment.names <- names(args)
      ans <-
          data.frame(y = unlist(exprlist),
                     treatment = factor(unlist(covlist)),
                     experiment = factor(rep(experiment.names, nsamples)))
      rownames(ans) <- NULL
      ans
  }
```

One point to keep in mind is that random effect models assume similar intrinsic variability ($\tau_\epsilon$) in all experiments. If there is no prior reason to believe that this is the case, it is often useful to scale the data beforehand. This can be done globally or on a per-gene basis. In the latter case, any filtering to leave out genes with low variability needs to be done before this step. See R code for examples.

```
> NevinsER.info <- list(exprs = scaled.exprs(NevinsER),
                        cov = NevinsER$ERstatus)
> VantER.info <- list(exprs = scaled.exprs(VantER),
                      cov = VantER$ERstatus)
> df3 <- makeDf2(3, Nevins = NevinsER.info, Vant = VantER.info)

> str(df3)

'data.frame':        162 obs. of  3 variables:
 $ y         : num  -1.611  0.893 -0.283 -1.395 -0.717 ...
 $ treatment : Factor w/ 2 levels "neg","pos": 2 2 2 1 1 1 1 2 2 2 ...
 $ experiment: Factor w/ 2 levels "Nevins","Vant": 1 1 1 1 1 1 1 1 1 1 ...
```

## Testing for interaction

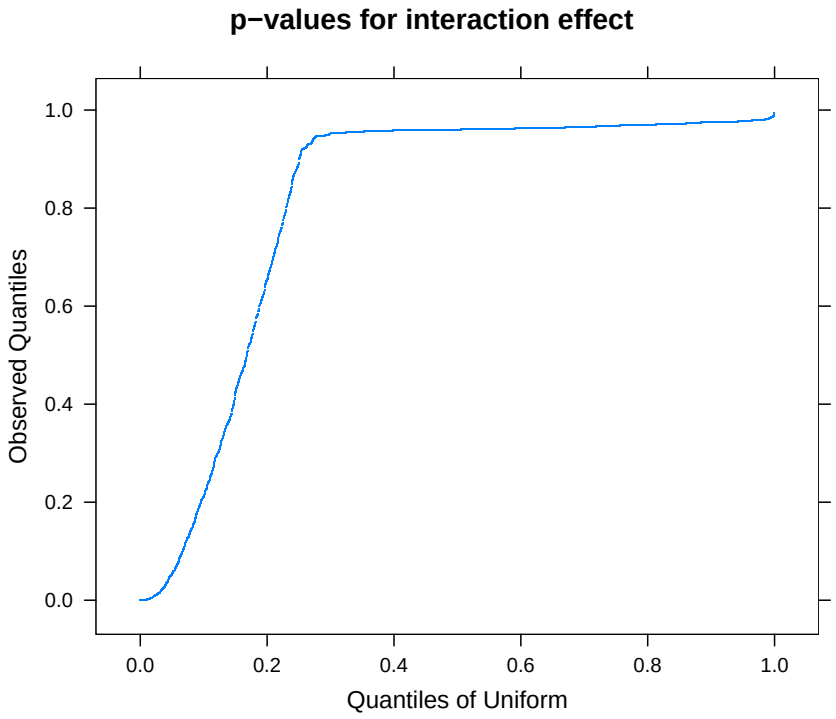
We fit the model in Equation (1) where the treatment effect is a fixed effect and experiment is considered to be a random effect. We also fit a model that includes a treatment by experiment interaction, and test the hypothesis that no interaction term is needed. There are essentially two ways in which the interaction could be important. In one situation the treatment has an opposite effect in the two experiments, we can also detect this by simply comparing the estimated effects for each experiment estimated separately. For such probes, or genes, it would not be appropriate to combine estimates. In the other case, the interaction suggests that the magnitude of the effect is different in one experiment, versus the other. For these probes it may simply be the case that the model is incorrect. For example, we might be looking for a change in mean abundance while the magnitude of the effect is a function of the abundance, and hence in samples where the abundance of mRNA transcript is larger a larger effect is observed. In cases where an interaction is absent, the model without an interaction will have more power to detect a treatment effect.

The following code snippet can fit these models, one gene at a time, and store the $p$-value for a likelihood ratio test:

```
> ## ngenes <- nrow(exprs(NevinsER)) # takes very long
> ngenes <- 5
> ERpvals <- numeric(ngenes)
> for(i in 1:ngenes)
  {
      cat(sprintf("\r%5g / %5g", i, ngenes))
      dfi <- makeDf2(i, Nevins = NevinsER.info, Vant = VantER.info)
      fm.null <-
          lme(y ~ 1 + treatment, data = dfi,
              random = ~ 1 | experiment,
              method = "ML")
      fm.full <-
          lme(y ~ 1 + treatment, data = dfi,
              random = ~ 1 | experiment/treatment,
              method = "ML")
      ERpvals[i] <- anova(fm.null, fm.full)[["p-value"]][2]
  }
```

This will take a fairly long while to run for all genes, but the results are already available (from slightly different fits) in the GeneMetaEx package. Next we plot these $p$-values in a Q-Q plot against the uniform distribution as reference.

```
> data("ERpvs")
> ERpvals <- unlist(eapply(ERpvs, function(x) x[2]))
> plot(qqmath(~ ERpvals,
              outer = TRUE,
              main = "p-values for interaction effect",
              xlab = "Quantiles of Uniform",
              ylab = "Observed Quantiles",
              distribution = qunif, pch = "."))
```




Perhaps the most striking feature in this plot is the very large number of $p$-values close to 1. This is a reflection of the fact that the hypothesis test here is being performed under non-standard conditions. The test is that the variance of the random effect is zero, and hence is on the boundary of the parameter space. In this case the asymptotics can be delicate (Crainiceanu and Ruppert, 2004) and further study is needed to fully interpret the output.

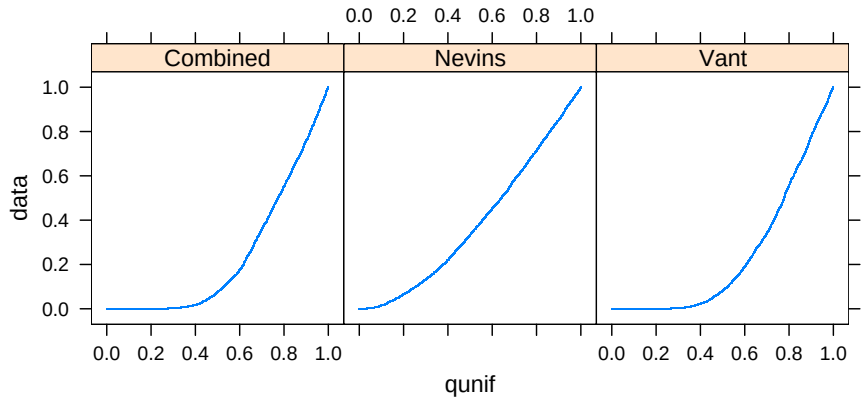
## Combined estimates of difference

The following code fits the combined mixed effect model as well as models (using `lm`) for the individual experiments.

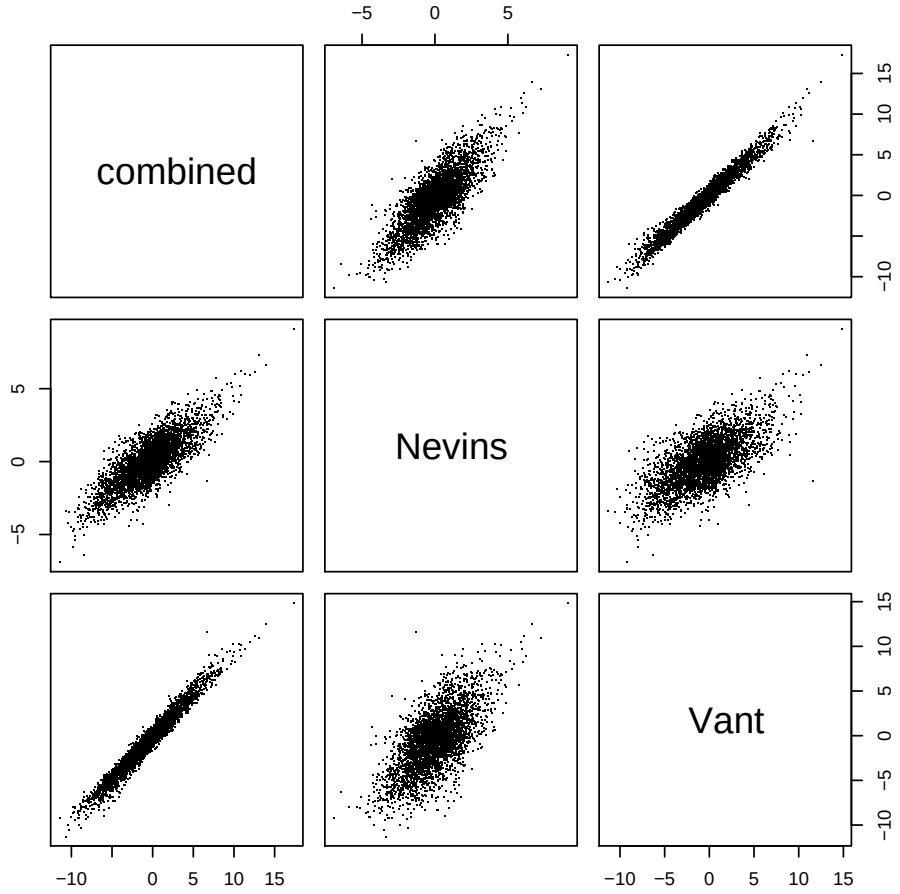
```
> ## ngenes <- nrow(exprs(NevinsER)) # slow
> ngenes <- 20
> ER.meandiff.combined <- data.frame(matrix(NA, nrow = ngenes, ncol = 4))
> ER.meandiff.Nevins <- data.frame(matrix(NA, nrow = ngenes, ncol = 4))
> ER.meandiff.Vant <- data.frame(matrix(NA, nrow = ngenes, ncol = 4))
> colnames(ER.meandiff.combined) <-
      colnames(ER.meandiff.Nevins) <-
      colnames(ER.meandiff.Vant) <-
      c("Value", "Std.Error", "t.value", "p.value")
> for(i in 1:ngenes)
  {
      cat(sprintf("\r%5g / %5g", i, ngenes))
      dfi <- makeDf2(i, Nevins = NevinsER.info, Vant = VantER.info)
      fm.lme <-
          lme(y ~ 1 + treatment, data = dfi,
              random = ~ 1 | experiment,
              method = "ML")
      fm.Nevins <-
          lm(y ~ 1 + treatment, data = dfi,
             subset = (experiment == "Nevins"))
      fm.Vant <-
          lm(y ~ 1 + treatment, data = dfi,
             subset = (experiment == "Vant"))

      ER.meandiff.combined[i, ] <- summary(fm.lme)$tTable[2, -3]
      ER.meandiff.Nevins[i, ] <- summary(fm.Nevins)$coefficients[2, ]
      ER.meandiff.Vant[i, ] <- summary(fm.Vant)$coefficients[2, ]
  }
```

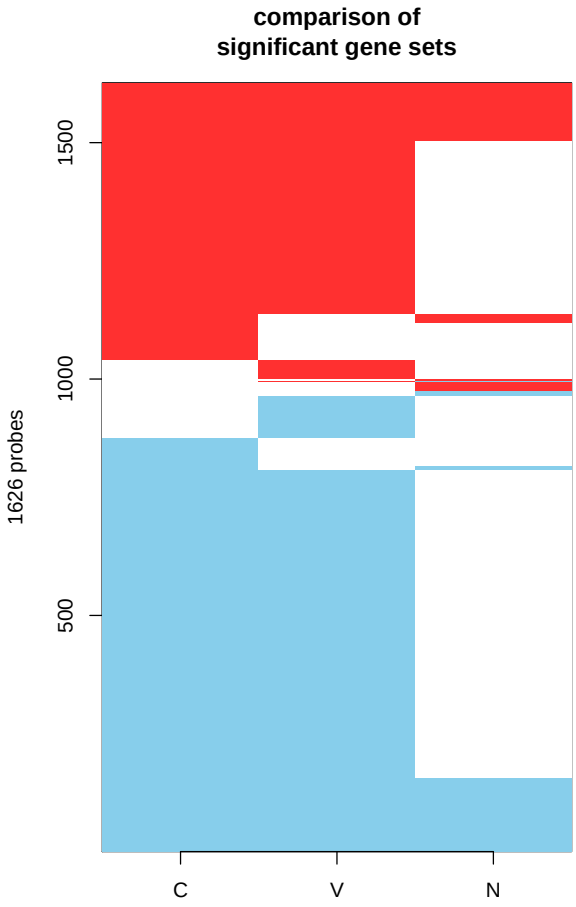
This takes a fairly long time as well (though less than before), and the results are available in a supplied R data file. We use this data to draw a Q-Q plot of the siginificance of the treatment effects in the three models:




We can also compare the estimated $t$-statistics pairwise:



Finally, we can study how many new (integration-driven) discoveries were made by looking at genes that were significant in the combined but not the individual studies.




**Exercise 6** *How would you interpret these results? Perform the similar analysis for the LN data sets. How do the two analyses compare?*

For a more complete discussion of these issues, see the GeneMetaEx vignette:

```
> openVignette("GeneMetaEx")
```

The standard reference for fitting mixed effect models using the nlme package is Pinheiro and Bates (2000). The next generation software being developed for such models is available in the lme4 package.

# References

J. K. Choi, U. Yu, S. Kim, et al. Combining multiple microarray studies and modeling interstudy variation. *Bioinformatics*, 19, Suppl. 1:i84–i90, 2003.

D.R. Cox and P. J. Solomon. *Components of Variance*. Chapman and Hall, New York, 2003.

C. M. Crainiceanu and D. Ruppert. Likelihood ratio tests in linear mixed models with one variance component. *JRSS, B*, 66:165–185, 2004.

J. C. Pinheiro and D. M. Bates. *Mixed-effects models in S and S-PLUS*. Springer, New York, 2000.

P. Roepman, LFA. Wessels, N. Kettelarij, et al. An expression profile for diagnosis of lymph node metastases from primary head and neck squamous cell carcinomas. *Nat Genet*, 37 (2):182–186, Feb 2005.

L. van't Veer, H. Dai, MJ. van de Vijver, et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415:530–536, 2002.

M. West, C. Blanchette, H. Dressman, et al. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proc Natl Acad Sci U S A*, 98(20):11462–11467, 2001.