Evaluation of VST algorithm in lumi package

Pan Du[1]*, Simon Lin[1]†, Wolfgang Huber[2]‡, Warrren A. Kibbe[1]§

March 31, 2012

[1]Robert H. Lurie Comprehensive Cancer Center
Northwestern University, Chicago, IL, 60611, USA

[2]EBI/EMBL, Cambridge, UK

Contents

1 Introduction

Variance stabilization is critical for the subsequent statistical inference to identify differentially expressed genes from microarray data. We devised a variance-stabilizing transformation (VST) by taking advantages of larger number of technical replicates available on the Illumina microarray. Here we use the Barnes data set, which has been packaged as lumiBarnes data package at the Bioconductor Experiment Data web page, to evaluate the VST algorithm. We will compare VST with popular base-2 logarithm transform and VSN method. To

*dupan@northwestern.edu
†s-lin2@northwestern.edu
‡huber@ebi.ac.uk
§wakibbe@northwestern.edu

facilitate the comparison, we used popular quantile normalization for both VST and log2 transformed data.

2 Required packages and data preprocessing

The evaluation requires the users to install packages: *lumi*, *vsn*, *genefilter*, *limma* and *lumiBarnes* (Experiment Data package). First, we need to load these packages:

```
> library("lumi")
> library("vsn")
> library("genefilter")
> library("RColorBrewer")
> library("limma")
> library("lumiBarnes")
> set.seed(0xbadbeef)
> ## Load the Barnes data set
> data("lumiBarnes")
```

We select the Barnes data [2] as the evaluation data set. For convenience, we created a Bioconductor experiment data package *lumiBarnes*. The data is kept in a LumiBatch Object. Because the Barnes data utilized the pre-released version of HumanRef-8 version 1 BeadChip, some probes on the chip do not exist in the public released HumanRef-8 version 1 BeadChip. For annotation consistence, these probes was removed in the *lumiBarnes* package. For the interested users, the raw data can be downloaded from the paper companion website: `http://www.bioinformatics.ubc.ca/pavlidis/lab/platformCompare/`.

Before preprocessing the data, we first compare the methods of fitting the relations between probe standard deviation and mean. The detailed implementation of methods is described in [1]. The results of using 'linear' and 'quadratic' method are shown in Figure 1 and Figure 2 respectively. Compare Figure 1 and Figure 2, we can see the 'quadratic' method over-fits the relations in the high expression range. As a result, VST uses 'linear' method by default to get more robust results.

```
> ## Select the blood and placenta samples
> selChip = !is.na(lumiBarnes$pctBlood)
> x.lumi <- lumiBarnes[, selChip]
> presentCount <- detectionCall(x.lumi)
> ## Since the Barnes data was not background removed, we will do background adjustment fi
> ## The background estimation will be based on the control probe information.
> ##  As the old version lumiBarnes library does not include controlData slot, we will che
> if (nrow(x.lumi@controlData) == 0) {
+         ## We will use the control probe information in the example.lumi in the updated
+         data(example.lumi)
+         x.lumi@controlData <- example.lumi@controlData
+ }
> x.lumi <- lumiB(x.lumi, method='bgAdjust')
> repl1 <- which(x.lumi$replicate=="A")
> repl2 <- which(x.lumi$replicate=="B")
> stopifnot(sum(selChip)==12L, length(repl1)==6L, length(repl2)==6L)
```

```
> temp <- lumiT(lumiBarnes[,1], fitMethod='linear', ifPlot=TRUE)

Perform vst transformation ...
2012-03-31 00:10:22 , processing array  1
```

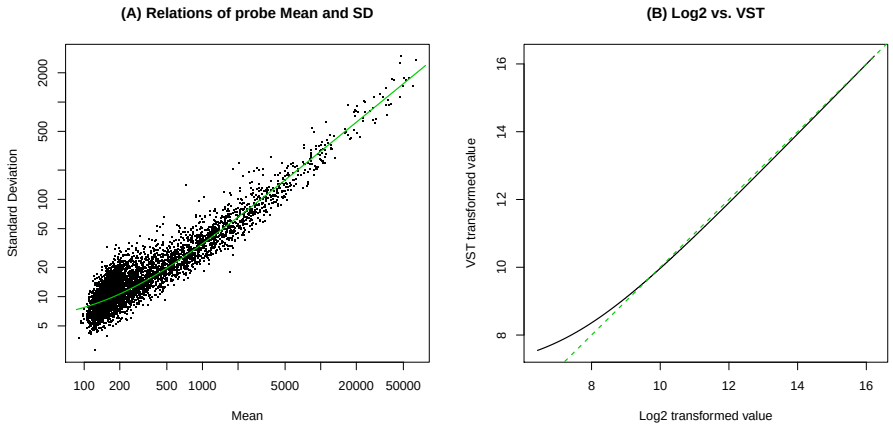

Figure 1: (A) The relations between probe standard deviation and mean by linear fitting. (B) Log2 vs. VST transformed values. The green line in figure A is the fitted curve; the green dotted line in figure B represents Log2 = VST.

```
> temp <- lumiT(lumiBarnes[,1], fitMethod='quadratic', ifPlot=TRUE)

Perform vst transformation ...
2012-03-31 00:10:23 , processing array  1
```

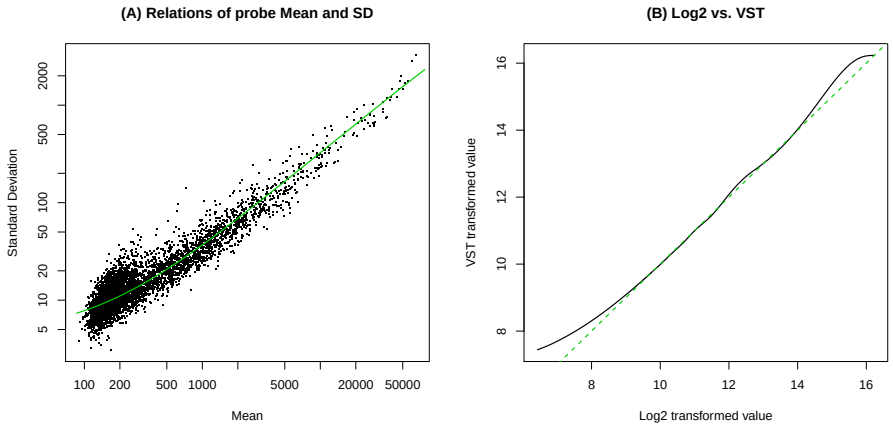

Figure 2: (A) The relations between probe standard deviation and mean by linear fitting. (B) Log2 vs. VST transformed values. The green line in figure A is the fitted curve; the green dotted line in figure B represents Log2 = VST.

Preprocess:

```
> ## VST transform and Quantile normalization
> x.lumi.vst <- lumiT(x.lumi)
> x.lumi.vst.quantile <- lumiN(x.lumi.vst, method='quantile')
> ## log2 transform and Quantile normalization
> x.lumi.log <- lumiT(x.lumi, method='log2')
> x.lumi.log.quantile <- lumiN(x.lumi.log, method='quantile')
> ## VSN normalization: use lts.quantile=0.5 since in the blood/placenta
> ##  comparison more genes are differentially expressed than what is
> ##   expected by the default of 0.9.
> x.lumi.vsn <- lumiN(x.lumi, method='vsn', lts.quantile=0.5)
> ## Add the vsn based on technical replicates
> vsn.pair <- exprs(x.lumi)
> cor.i <- NULL
> for(i in 1:length(repl1)) {
+         vsn.pair[, c(i, i+length(repl1))] <- exprs(vsn2(vsn.pair[, c(repl1[i], repl2[i])
+ }
> # vsn.quantile <- normalize.quantiles(vsn.pair)
> # rownames(vsn.quantile) <- rownames(vsn.pair)
> # colnames(vsn.quantile) <- colnames(vsn.pair)
>
>
> normDataList <- list('VST-Quantile'=exprs(x.lumi.vst.quantile),
+                      'Log2-Quantile'=exprs(x.lumi.log.quantile),
+                      'VSN'=exprs(x.lumi.vsn))                # , 'VSN-Quantile'=vsn.quant
>
> ## scatter plots:
> ## pairs(exprs(x.lumi.vsn), panel=function(...){par(new=TRUE);smoothScatter(..., nrpoint
```

3 Evaluation of the VST algorithm

3.1 Correlation between the technical replicate microarrays

A good preprocessing method will improve the correlation between the technical replicate microarrays. Here will calculate the correlation between six pairs of technical replicate chips and plot them as the box plot, as shown in Figure 3. We can see VST improves the consistency between replicates.

```
> ## Check the correlation between technique replicates
> tempDataList <- c(normDataList, list(vsn.pair))
> names(tempDataList) <- c(names(normDataList), 'VSN-techReplicate')
> chipCorList <- matrix(as.numeric(NA), nrow=length(repl1), ncol=length(tempDataList))
> colnames(chipCorList) <- names(tempDataList)
> for (i in seq(along= tempDataList))
+   for (j in seq(along=repl1))
+     chipCorList[j,i] = cor(tempDataList[[i]][, c(repl1[j], repl2[j])])[1,2]
```

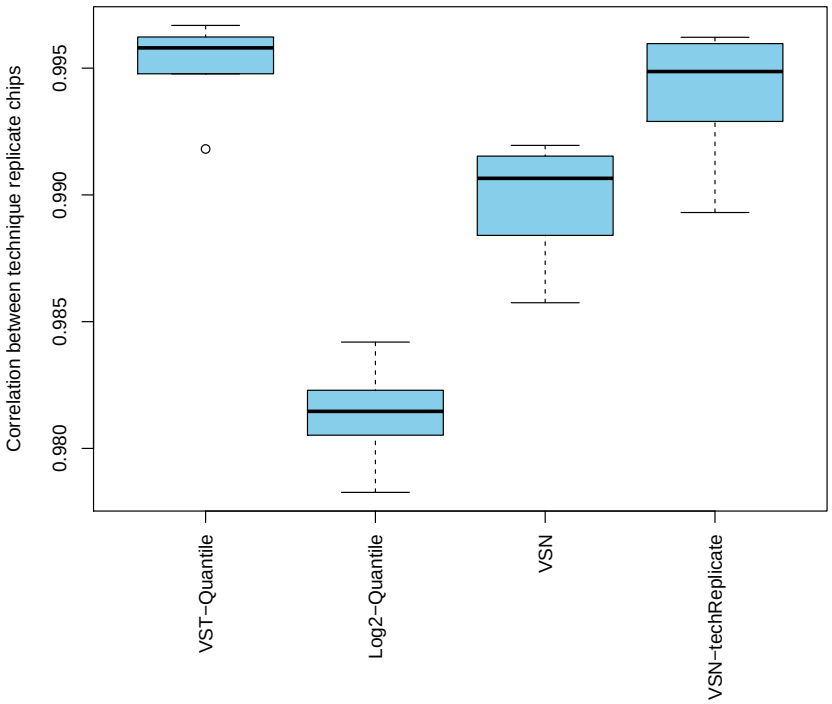

Figure 3: Comparison of the correlation between technical replicate chips after preprocessing. The VSN-techReplicate method performed the VSN within each pair of technical replicate samples and then calculated their correlations.

3.2 Variance stabilizing between the technique replicate microarrays

A good variance stabilizing method should stabilize the variance between the technique replicates. Here we plot the mean and standard deviation relations between a pair of technique replicates, as shown in Figure 4. Users can select other pairs of replicates and plot the pictures.

3.3 Variation within replicates vs. variation between conditions

To assess the signal to noise ratio, we assess

$$\frac{\sigma^2_{\text{between groups}}}{\sigma^2_{\text{within groups}}}.$$

For n groups, by its generalisation, the F-statistic.

```
> fac <- factor(paste(x.lumi$pctBlood, x.lumi$pctPlacenta, sep=":"))
> rf <- lapply(normDataList, function(x) {
+ filtered.x = x[presentCount > 0,]
+ ftest.x = rowFtests(filtered.x, fac=fac)
+ ftest.x$IDs <- rownames(filtered.x)
+ return(ftest.x)
+ })
> ef <- sapply( rf, function(x) ecdf(x$p.value))
```

The result is shown in Figure 5. We can see the difference among these methods are not big, however, the VST is consistently better than the log2 and VSN methods.

3.4 Correlation between the expression profiles and dilution profile

Here we want to compare the correlation between the expression profiles and dilution profile. Because these concordant genes are more likely to be related with the dilution process, a good transformation should improve or at least not worsen the correlation of the expression profiles and dilution profile. Figure 6 shows, VST transformed data improve this correlation because there are more probes with high correlation (the absolute values of correlation coefficient close to 1).

```
> modelProfile1 <- c(100, 95, 75, 50, 25, 0, 100, 95, 75, 50, 25, 0)
> corrList <- lapply(normDataList, function(x) {
+         x <- x[presentCount > 0, ]
+         corr1 <- apply(x, 1, cor, y=modelProfile1)
+         return(corr1)
+         } )
```

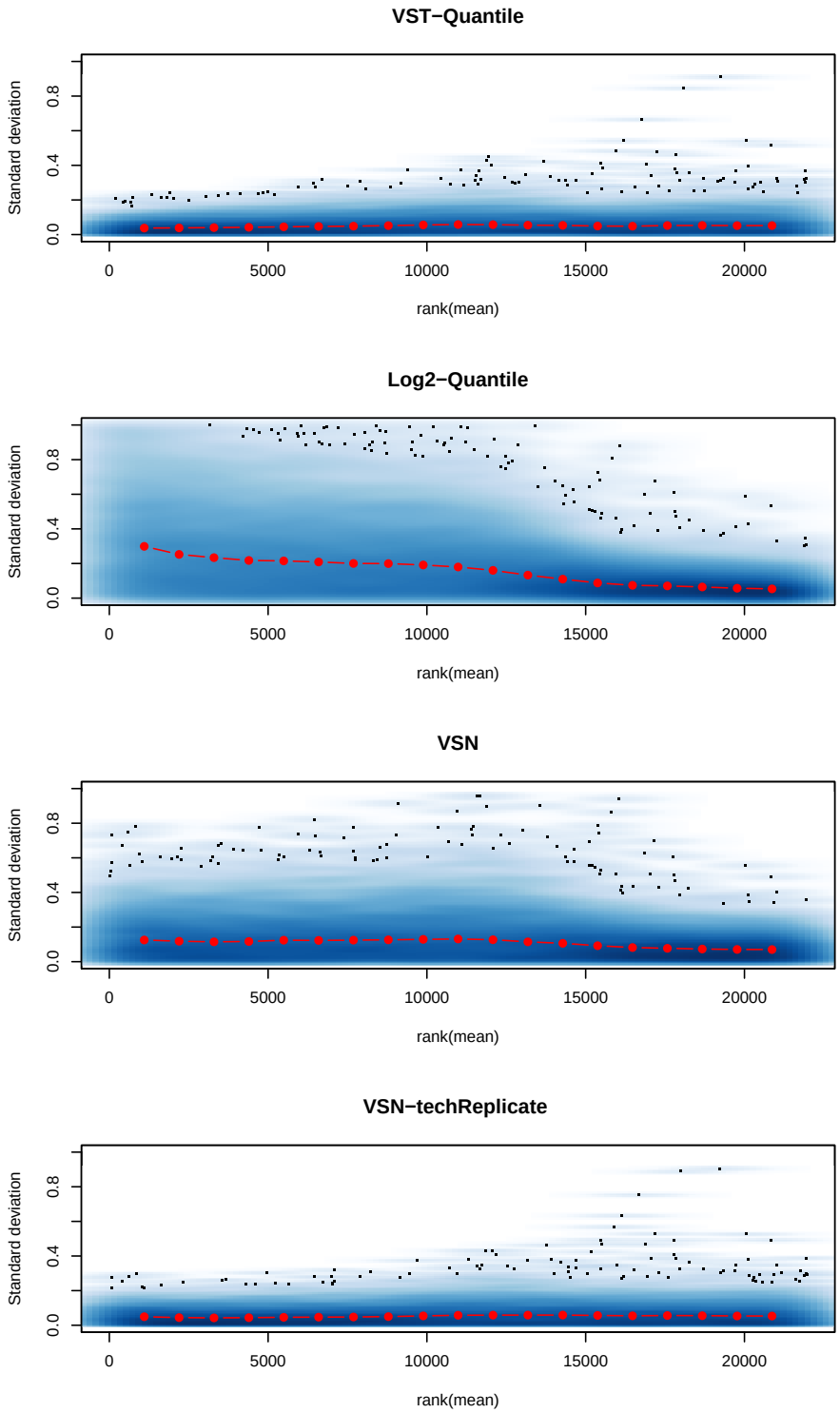

Figure 4: Mean and standard deviation relations of the technical replicate microarrays A01 and B01. The VSN-techReplicate method performed the VSN only within the pair of technical replicate samples.

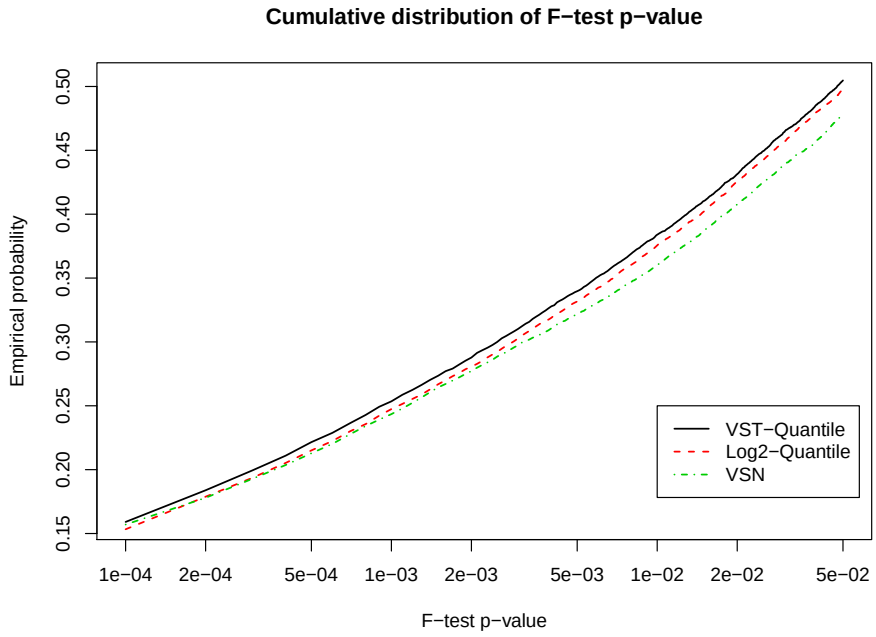

Figure 5: Cumulative distribution functions of p-values obtained from a) reporter-wise F-tests (by factor `fac`). These are monotonous measures of the ratio between variation within replicates and variation between conditions, or in other words, the signal-to-noise ratio.

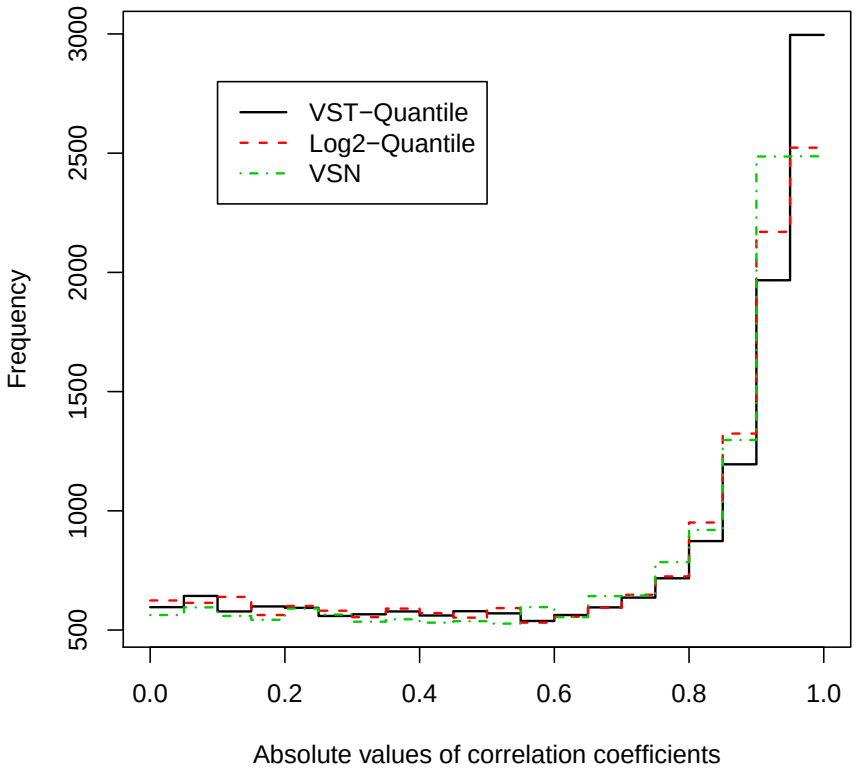

Figure 6: Compare the histogram of the correlation between the expression profiles and dilution profile

3.5 Evaluation based on the identification of differentially expressed genes

For better evaluation, we want to evaluate the VST algorithm based on the detection of differentially expressed genes. First, we want to see the percentage of concordant probes (a probe with a correlation coefficient larger than 0.8 between the normalized intensity profile and the real dilution profile (six dilution ratios with two replicates at each dilution)) among the most significant probes (ranking based on F-test p-values). The result is shown in Figure 7. We can see the VST processed data has obviously higher percentage of concordant probes than the log2 and VSN methods.

```
> topNumList <- seq(50, 3000, by=100)
> corTh <- 0.8
> highCorrNumMatrix <- NULL
> for (i in 1:length(rf)) {
+         probeList <- rf[[i]]$IDs
+         ordProbe.i <- probeList[order(abs(rf[[i]]$p.value), decreasing=FALSE)]
+         corr1 <- corrList[[i]]
+         matchNum.j <- NULL
+         for (topNum.j in topNumList) {
+                 topProbe.j <- ordProbe.i[1:topNum.j]
+                 matchNum.j <- c(matchNum.j, length(which(abs(corr1[topProbe.j]) > corTh)
+         }
+         highCorrNumMatrix <- cbind(highCorrNumMatrix, matchNum.j)
+ }
> rownames(highCorrNumMatrix) <- topNumList
> colnames(highCorrNumMatrix) <- names(rf)
```

The result is shown in Figure 7. We can see the difference among these methods are not big, however, the VST is consistently better than the log2 and VSN methods.

Next, we selected the differentially expressed genes by comparing two conditions. The p-values will be estimated by the Bioconductor *limma* package. To better evaluate the overall performance, we first ranked the probes with their p-values from low to high, then calculate the percentage of concordant probes among different number of most significant probes, as shown in Figure 8. The result indicates that VST-quantile outperforms Log2.Quantile in terms of the concordance evaluation.

Identify the differentially expressed genes by using limma package:

```
> ## Select the comparing chip index
> sampleInfo <- pData(phenoData(x.lumi))
> sampleType <- paste(sampleInfo[,'pctBlood'], sampleInfo[,'pctPlacenta'], sep=':')
> sampleType <- paste('c', sampleType, sep='')
> ## Comparing index
> ## used in the paper (the most challenging comparison):
> compareInd <- c(repl1[1:2], repl2[1:2])
> compareType <- sampleType[compareInd]
> fitList.limma <- NULL
> for (i in 1:length(normDataList)) {
```

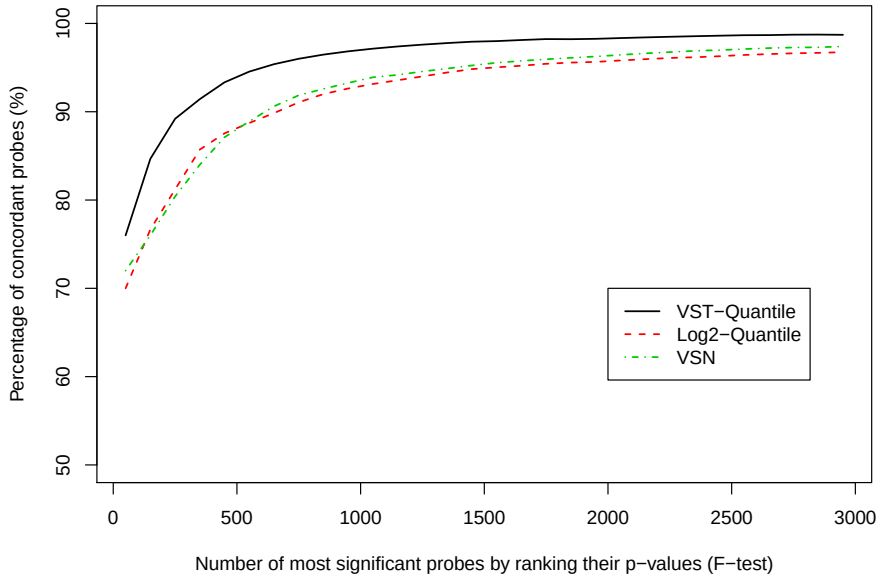

Figure 7: Cumulative distribution functions of p-values obtained from a) reporter-wise F-tests (by factor `fac`). These are monotonous measures of the ratio between variation within replicates and variation between conditions, or in other words, the signal-to-noise ratio.

```
+         selDataMatrix <- normDataList[[i]]
+         selDataMatrix <- selDataMatrix[presentCount > 0, ]
+         selProbe <- rownames(selDataMatrix)
+         compareMatrix <- selDataMatrix[, compareInd]
+
+         design <- model.matrix(~ 0 + as.factor(compareType))
+         colnames(design) <- c('A', 'B')
+         fit1 <- lmFit(compareMatrix, design)
+         contMatrix <- makeContrasts('A-B'=A - B, levels=design)
+         fit2 <- contrasts.fit(fit1, contMatrix)
+         fit <- eBayes(fit2)
+         fitList.limma <- c(fitList.limma, list(fit))
+ }
> names(fitList.limma) <- names(normDataList)
```

Estimate the number of concordance probes (a probe with a correlation coefficient larger than 0.8 between the normalized intensity profile and the real dilution profile (six dilution ratios with two replicates at each dilution)) among the top differentially expressed genes (ranked based on p-values estimated by *limma*).:

```
> ## Check the correlation of the top differentiated probes based on the limma results
> ## rank the probes based on the p-values of limma result
> fitList <- fitList.limma
> topNumList <- c(30, seq(35, 1000, by=30))
> corTh <- 0.8
> highCorrNumMatrix <- NULL
> for (i in 1:length(fitList)) {
+         probeList <- rownames(fitList[[i]]$p.value)
+         ordProbe.i <- probeList[order(abs(fitList[[i]]$p.value[,1]), decreasing=FALSE)]
+         profileMatrix <- normDataList[[i]][ordProbe.i, ]
+
+         modelProfile1 <- c(100, 95, 75, 50, 25, 0, 100, 95, 75, 50, 25, 0)
+         corr1 <- apply(profileMatrix, 1, cor, y=modelProfile1)
+         names(corr1) <- ordProbe.i
+         matchNum.j <- NULL
+         for (topNum.j in topNumList) {
+                 topProbe.j <- ordProbe.i[1:topNum.j]
+                 matchNum.j <- c(matchNum.j, length(which(abs(corr1[topProbe.j]) > corTh)
+         }
+         highCorrNumMatrix <- cbind(highCorrNumMatrix, matchNum.j)
+ }
> rownames(highCorrNumMatrix) <- topNumList
> colnames(highCorrNumMatrix) <- names(fitList)
```

4 Conclusion

The users can select different samples for the comparison and change the cut-off thresholds in the evaluation. The results should be similar, i.e., the VST

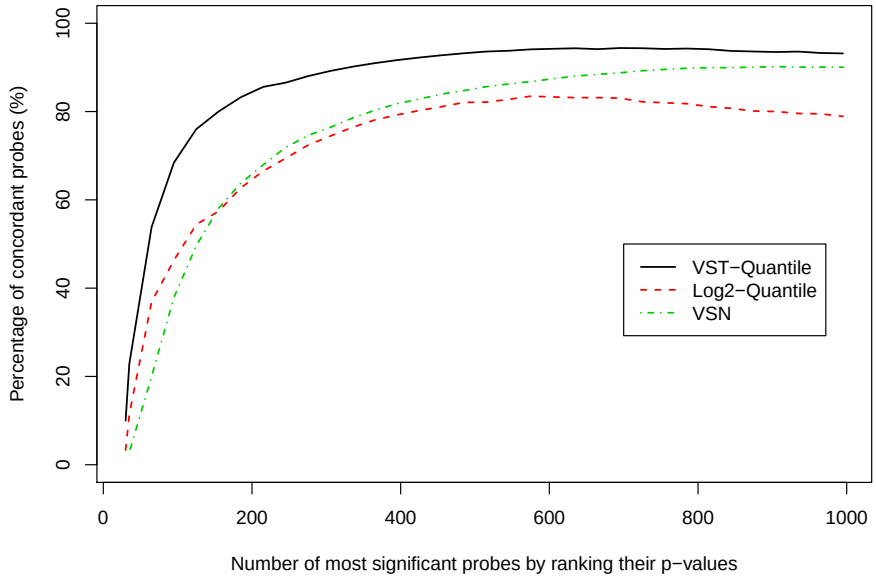

Figure 8: The concordance between the expression and dilution profiles of the selected differentially expressed genes

algorithm is better than the log2 transformation and VSN for this evaluation data set because it utilizes the mean and standard deviation information at the bead-level.

5 Session Info

```
> toLatex(sessionInfo())
```

- R version 2.15.0 RC (2012-03-22 r58802), `i386-apple-darwin9.8.0`
- Locale: `C`
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: AnnotationDbi 1.18.0, Biobase 2.16.0, BiocGenerics 0.2.0, DBI 0.2-5, RColorBrewer 1.0-5, RSQLite 0.11.1, annotate 1.34.0, genefilter 1.38.0, ggplot2 0.9.0, limma 3.12.0, lumi 2.8.0, lumiBarnes 1.3.7, lumiHumanAll.db 1.16.0, lumiHumanIDMapping 1.10.0, methylumi 2.2.0, nleqslv 1.9.3, org.Hs.eg.db 2.7.1, reshape2 1.2.1, scales 0.2.0, vsn 3.24.0
- Loaded via a namespace (and not attached): BSgenome 1.24.0, BiocInstaller 1.4.0, Biostrings 2.24.0, DNAcopy 1.30.0, GenomicRanges 1.8.0, IRanges 1.14.0, KernSmooth 2.23-7, MASS 7.3-17, Matrix 1.0-6, RCurl 1.91-1, Rsamtools 1.8.0, XML 3.9-4, affy 1.34.0,

affyio 1.24.0, bigmemory 4.2.11, bitops 1.0-4.1, colorspace 1.1-1, dichromat 1.2-4, digest 0.5.2, genoset 1.6.0, grid 2.15.0, hdrcde 2.16, lattice 0.20-6, memoise 0.1, mgcv 1.7-13, munsell 0.3, nlme 3.1-103, plyr 1.7.1, preprocessCore 1.18.0, proto 0.3-9.2, rtracklayer 1.16.0, splines 2.15.0, stats4 2.15.0, stringr 0.6, survival 2.36-12, tools 2.15.0, xtable 1.7-0, zlibbioc 1.2.0

6 Reference

1. Lin, S.M., Du, P., Kibbe, W.A., "Model-based Variance-stabilizing Transformation for Illumina Mi-croarray Data", under review

2. Barnes, M., Freudenberg, J., Thompson, S., Aronow, B. and Pav-lidis, P. (2005) "Experimental comparison and cross-validation of the Affymetrix and Illumina gene expression analysis platforms", Nucleic Acids Res, 33, 5914-5923.