University of Louisville
ThinkIR: The University of Louisville's Institutional Repository

Electronic Theses and Dissertations

8-2011

Consistency of differentially expressed gene rankings based on subsets of microarray data.

Dake Yang
University of Louisville

Follow this and additional works at: https://ir.library.louisville.edu/etd

Recommended Citation

Yang, Dake, "Consistency of differentially expressed gene rankings based on subsets of microarray data." (2011). *Electronic Theses and Dissertations.* Paper 1615.
https://doi.org/10.18297/etd/1615

This Master's Thesis is brought to you for free and open access by ThinkIR: The University of Louisville's Institutional Repository. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of ThinkIR: The University of Louisville's Institutional Repository. This title appears here courtesy of the author, who has retained all other copyrights. For more information, please contact thinkir@louisville.edu.

CONSISTENCY OF DIFFERENTIALLY EXPRESSED GENE RANKINGS BASED ON SUBSETS OF MICROARRAY DATA

By

Dake Yang

A Thesis

Submitted to the Department of Bioinformatics and Biostatistics University of Louisville

In Fulfillment of the Requirements

for the Degree of

Master of Science

Department of Bioinformatics and Biostatistics

University of Louisville

Louisville, KY

August 2011

Copyright 2011 by Dake Yang

All rights reserved

Consistency of Differentially Expressed Gene Rankings Based on Subsets of Microarray Data

By

Dake Yang

A Thesis Approved on

by the following Thesis Committee:

_____

Thesis Director (Guy Brock)

_____

Rudolph S. Parrish

_____

_____

_____

ABSTRACT

CONSISTENCY OF DIFFERENTIALLY EXPRESSED GENE RANKINGS BASED ON SUBSETS OF MICROARRAY DATA

Dake Yang

July 21, 2011

Data derived from gene expression microarrays are frequently used to identify candidate genes which can characterize and distinguish between two biological phenotypes. A key step in this process is the selection of an appropriate test statistic to identify which genes are differentially expressed between the two tissues. Although many methods have been explicitly developed for this purpose, the traditional *t*-test still remains a popular choice. In this study, we evaluate the empirical impact of choice of test-statistic on the resulting list of differentially expressed genes, in particular when the available sample size is small.

We evaluated several different methods for detecting differentially expressed genes (*t*-test, empirical Bayes, and SAM) using ten different publicly available data sets. First, we obtained gene lists based on the full data using the different methods. Then, we selected subsamples from the full data, and obtained gene lists based on these subsamples. The consistency was quantified using several scores. Factors evaluated in the empirical study included the size of the subset and the length of the differentially expressed gene list.

We found that when the sample size of the subset is small, the resulting gene list based on the t-test has a very low consistency, while empirical Bayes and SAM have much higher consistencies. This result is particularly evident when considering only the top ranked genes. When sample sizes are larger, all three methods have the

same performance. We recommend that investigators use these moderated versions in lieu of the t-test when the sample size is small.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

DNA microarray is a tool in biotechnology for comparing the expression of genes on a genomic scale across two or more samples of messenger RNA (mRNA). The microarray consists of thousands of features, which are small DNA oligonucleotides. Each feature contains a particular gene sequence, called a probe, which allows the microarray to simultaneously monitor the relative expression of mRNAs in two or more tissue samples for thousands of genes within the cell. In microarray experiments, one common but important question is how to identify those genes whose expression level changes across different experimental samples or different types of tissue samples (Parrish 2009). Thus, the goal of differential gene expression analysis is to determine the genes whose expression level changes significantly according to the experimental condition.

The microarray is a novel tool for comparing the expression of thousands of genes, but it also brings to light new statistical problems because of the high dimensionality of the data, with a relatively small number of replicates. There are two main issues with microarray data (Xu 2009). First, microarray gene expression data are commonly considered to be highly noisy, because there are many flaws inherent in the current technology. Noise comes from two primary sources in microarray experiments, biological noise and technical noise (Tu 2002). Biological noise consists

of the inherent variation from patient to patient, the variation in the tissues sampled, and the variation in the cellular composition of the tissue that is subsequently hybridized to the array. Technical noise comes from the differences in sample processing and experimental variables. Secondly, microarray data always contain a much larger number of genes (usually several thousand or more features) relative to the number of samples (usually less than one hundred). With the large number of hypothesis tests being conducted and a relatively small sample size, many spurious results can appear and many biologically meaningful genes can fail to reach statistical significance. In gene expression analysis, a gene is considered differentially expressed when its expression level changes systematically between two treatment conditions (McCarthy and Smyth 2009). The biological significance is typically measured by the relative expression level between the two conditions (the fold-change), while the statistical significance is measured by comparison of a test-statistic to a reference distribution (the p-value). Ranking of significant genes on the basis of the two methods does not always agree, leading investigators to often make comprises between the two approaches.

Due to the huge data volume and inherent variation in microarrays, several statistical methods have been proposed to address these problems (Baldi 2001; Efron 2009). In our paper, we discuss three commonly used methods for identifying which genes are differentially expressed between two different experimental conditions or tissue types. They are the *t*-test, the significance analysis of microarrays (SAM) (Tusher 2001), and the empirical Bayesian (Smyth 2005) methods. In each case, the hypotheses been tested for each gene are the null hypothesis H_0: the gene is not differentially expressed between two different experimental conditions or tissue types, and the alternative hypothesis H_A: the gene is differentially expressed between two

different experimental conditions or tissue types. The *t*-test is a traditional method for detecting differentially expressed genes in microarray analysis. The *t* statistic can be used to determine which genes are significantly differentially expressed by comparing the test statistic to a t-distribution with *n*-2 degrees of freedom, where *n* is the total number of microarrays. But in the *t*-test method, a gene with a low expression level but small variance can result in a large absolute *t* statistic even for small mean differences between the two conditions. So this gene can be declared to be differentially expressed, even if the difference in expression is not biologically meaningful.

To solve this problem, alternative methods have been proposed. To solve the small variance problem of the *t*-test, SAM modifies the standard *t*- statistic by adding a small 'fudge factor' to the variance in the denominator. The SAM method compares this modified test statistic to an expected value under the null hypothesis, which is determined by permutations at the gene expression measurements. Differences between observed and expected values which are above a threshold are considered statistically significant, where the threshold is determined by the desired false discovery rate. Smyth et al. 2004 proposed a similarly derived empirical Bayes approach, which shrinks the estimated sample variances towards a pooled estimate. Both SAM and empirical Bayes estimators are designed to have robust performance, particularly for experiments with small numbers of arrays.

Although these statistical methods have been proposed as improved methods to determine differential gene expression, not too many studies (Haslett 2002; Jeffery 2006; Hong and Breitling 2008) have empirically compared the performances of these different statistical methods. Thus, it is important to evaluate the success of these methods through empirical comparisons. The traditional *t*-test still remains a popular

choice among biologists, despite the reputed disadvantages in this method when the available sample size is small. So, the goal of this thesis research is to compare and evaluate how well the three methods (the *t*-test, SAM, and empirical Bayes) perform with real microarray data. To accomplish this, we evaluate the consistency in determining differentially expressed genes for each method, based on numerical measures of the degree of overlap between two gene lists (Zhang 2009). Further, we evaluate the consistency for each method, when comparing differentially expressed gene lists obtained from subsets of two different datasets of the same disease. We consider that the methods are performing well in detecting the differentially expressed gene lists when the scores are high. We define that one method is better than the other, when the scores calculated by one method are higher than other scores from the other method under the same situation.

The rest of this thesis is organized as follows. In Chapter 2, section 2.1 we describe the three DEGs (differentially expressed gene) detecting methods (the *t*-test, SAM, and empirical Bayes). In section 2.2, we describe the scores which are used to measure the consistency between two gene lists (POG, POGR, nPOG and nPOGR). We describe the two groups of data sets for evaluating the consistency of DEdetection methods in sections 2.3.1 and 2.3.2. Chapter 3 gives the results from our study, and Chapter 4 is the discussion

CHAPTER II

METHODS

2.1 Methods for detecting the differentially expressed genes

We selected three popular methods (*t*-test, empirical Bayes and significance analysis of microarrays (SAM)) for determining differentially expressed genes (DEGs) in microarray data. The standard notations for the three methods we defined as follows. Suppose the goal is to detect differentially expressed genes between two classes of samples, normal and diseased. Suppose group one (normal) contains n samples and the group two (diseased) contains m samples. Suppose there are gene expression measurements for all the G genes. For the any given ith gene (i=1,2,...G), let $\overline{X}_{1i}$ be the mean of the ith gene expression measurements amongst the samples from group one, with corresponding standard deviation $S_{X_{1i}}$. Similarly, define $\overline{X}_{2i}$ and $S_{X_{2i}}$ to be the mean and standard deviation of the ith gene expression measurements from the samples of group 2. The hypothesis being tested for each of genes is H_0: $\mu_{1i} = \mu_{2i}$ vs H_A: $\mu_{1i} \neq \mu_{2i}$, where μ_{1i} and μ_{2i} are the population mean expression measurements for the gene in the two groups.

2.1.1 T-test

The two-sample *t*-test on each gene can be used to detect which genes are differentially expressed between two classes of samples. The *t*-statistic for the *i*th gene assuming using a pooled estimator for the standard deviation is defined as follows:

$$t_i = \frac{(\overline{X}_{1i} - \overline{X}_{2i})}{\sqrt{\frac{nS^2_{X_{1i}} + mS^2_{X_{2i}}}{n+m-2}}\sqrt{\frac{1}{n}+\frac{1}{m}}}.$$

The gene-specific *t*-test uses only the gene expression values specific to each gene to calculate differential expression. Under the null hypothesis, the *t*-statistic follows a *t*-distribution with approximately $n + m$ -2 degrees of freedom. The *P*-value for the two-sided hypothesis test is calculated as $2P(T_{n+m-2} \geq t_i)$, where T_{n+m-2} denotes a random variable with a t distribution with $n+m-2$ degrees of freedom. The null hypothesis is rejected if the *P*-value is less than the α level of the test.

2.1.2 SAM (Significance Analysis of Microarrays)

With the high levels of noise in microarray data and the small sample sizes, the adequacy of the *t*-statistic is questionable. Since there are a large number of genes inmicroarray datasets, there will always be some genes which have a low standard deviation by chance. Therefore, these genes will have a large t-statistic and will be falsely predicted to be differentially expression genes. Several authors have proposed improvements on the *t* test for genome-wide expression measurements.

To improve upon the deficiencies of the *t*-test mentioned above, Tusher (Tusher 2001) proposed the significance analysis of microarrays (SAM) method to

select differentially expressed genes by adding a small, strictly positive constant S_0, (the so called *fudge factor*) to the denominator of the usual *t*-statistic. Define gene specific "relative differences" $d_i (i = 1, 2, ...G)$ and gene specific standard deviations $S_i (i = 1, 2, ...G)$.

$$d_i = \frac{\overline{X}_{1i} - \overline{X}_{2i}}{S_i + S_0},$$

$$S_i = \sqrt{\frac{1/n + 1/m}{n+m-2} \{ \sum_{j=1}^{n} (X_{1ij} - \overline{X}_{1i})^2 + \sum_{j=1}^{m} (X_{2ij} - \overline{X}_{2i})^2 \}}.$$

The *fudge factor* S_0 is calculated by minimizing the coefficient of variation of d_i as a function of S_i (Tusher 2001). A permutation procedure is used to calculate the *P*-value for each gene.

DE genes are determined by comparing the observed relative differences d_i with the expected relative differences d_i^E, where d_i^E are calculated by averaging the relative differences from permutations of the data. Genes with differences between observed and expected values greater than a threshold Δ are declared statistically significant. The threshold Δ is a user controlled parameter, and SAM provides an estimate of the FDR (false discovery rate) for each Δ value.

2.1.3 Empirical Bayes

A hierarchical model (Smyth 2004; Smyth 2005) is used to model distribution of the variances and differences in mean expression between the two tissue types for each gene. Based on the hierarchical model for the variances, for each gene, a posterior estimate of the population variance σ_i^2 for each gene, given the sample

variance S_i^2, is obtained. This expectation is termed $\tilde{S}_i^2$, and is calculated as a weighted average of the prior variance S_0^2 and the sample variance S_i^2,

$$\tilde{S}_i^2 = \frac{d_0 S_0^2 + d_i S_i^2}{d_0 + d_i},$$

Where d_0 is the degrees of freedom for the prior distribution of S_i^2 and d_i is the degrees of freedom for the ordinary *t*-statistic. The moderated *t*-statistic is then defined as

$$\tilde{t}_i = \frac{\overline{X}_{1i} - \overline{X}_{2i}}{\tilde{S}_i \sqrt{\frac{1}{n+m}}}.$$

The moderated *t*-statistic follows a *t*-distribution with $d_i + d_0$ degrees of freedom, under the null hypothesis.

2.2 POG and POGR scores

In this section, we define the metrics used to measure the consistency between two gene lists. For example, two separate lists of differentially expressed genes produced by independent studies of the same disease. A relatively straightforward metric is the percentage of overlapping genes (POG) metric, which has been used by numerous previous studies (Shi 2006; Chen 2007). The POG measures the consistency of two separate gene lists by determining the percentage of overlapped genes between the two lists. Specifically, assume that *n* genes are shared between list 1 with length l_1, and list 2 with length l_2. The calculation of the POG score depends on which list is determined to be the 'reference' list, i.e. the POG between list 1 and list 2 is $POG_{12} = \frac{n}{l_1}$ and the POG between list 2 and list 1 is $POG_{21} = \frac{n}{l_2}$. The POG is

a useful metric to evaluate the reproducibility of gene lists from independent studies, and has been previously used to measure the consistency of gene lists obtained from separate studies of the same disease (Zhang 2009). We can also consider the POG score as equivalent to a popular index of diagnostic accuracy, the sensitivity.

To see this, assume that the reference list (say, list 1) contains the 'true' differentially expressed genes, equivalent to the individuals which truly have the disease of interest in the evaluate of a diagnostics test. Then, the length of list 1 (l_1) is the same as the denominator of the sensitivity, and the length of the overlap of the two lists (n) is the same as the numerator of the sensitivity.

An extension to the POG score, called the percentage of overlapping genes-related or POGR score, considers not only those genes which are shared between the two lists, but also those genes which are highly correlated with each other. The motivation here is to consider a measure of the reproducibility of gene lists, which accounts for the coordinated molecular changes which frequently occur in biological systems. Similar to the POG score, the POGR (percentage of overlapping genes-related) metric is used to evaluate the consistency between two gene lists. Let n_{r12} (or n_{r21}) represent the number of genes between two lists which are not shared but are significantly correlated. Here, two genes are defined as a correlated pair if and only if they are significantly correlated in both datasets. Then, the formula for the POGR scores are,

$$POGR_{12} = \left(n + n_{r12}\right) / l_1 \text{ or } POGR_{21} = \left(n + n_{r21}\right) / l_2 .$$

Both the POG and POGR are in the range of 0 to 1. Generally, a high POG or

POGR score is wanted between two lists. A problem with both, however, is that they are positively correlated with the length of the lists. Further, when the expressed genes are highly correlated with each other in a biological system, the POGR score between two randomly selected gene lists can be high, since n_{r12} (or n_{r21}) may be high. This is because that, when the genes are highly correlated, the n_{r12} maybe high. To solve this problem, Zhang et al, (2009) introduced normalized versions of the POG and POGR scores (nPOG, nPOGR), where described below.

Define $E(n)$ as the expected number of the shared genes between two randomly generated lists with lengths l_1 and l_2 lists. Then, the expected POG is

$$E(POG_{12}) = E(n)/l_1 \quad \text{or} \quad E(POG_{21}) = E(n)/l_2 .$$

The normalized POG is similar in calculation to the kappa coefficient, which represents a chance-corrected index of agreement between categorical variables. The E(POG_{12}) [or E(POG_{21})] can be estimated by simulation, by averaging the POG scores from10,000 pairs of randomly generated lists (with lengths l_1 and l_2). Randomly selected lists of length l_1 (or l_2) can be generated by drawing l_1 (or l_2) genes without replacement from the complete list of genes. The POG as the ratio of the observed scores to the maximum potential scores as follows:

$$nPOG_{12} = \frac{POG_{12} - E(POG_{12})}{1 - E(POG_{12})} = \frac{n - E(n)}{l_1 - E(n)}$$

$$nPOG_{21} = \frac{POG_{21} - E(POG_{21})}{1 - E(POG_{21})} = \frac{n - E(\mathrm{n})}{l_2 - E(n)}$$

We can normalize the POGR score in a similar way. The $E(POGR_{12})$ or

$[E(POGR_{21})]$ can be calculated by adding the expected number of correlated genes between the two lists, $E(n_{r12})$ [or $E(n_{r21})$], to the expected number of overlapped genes, $E(n)$. So, the normalized POGR score is

$$nPOG_{12} = \frac{POGR_{12} - E(POGR_{12})}{1 - E(POGR_{12})} = \frac{n + n_{r12} - E(n) - E(n_{r12})}{l_1 - E(n) - E(n_{r12})}$$

or

$$nPOG_{21} = \frac{POGR_{21} - E(POGR_{21})}{1 - E(POGR_{21})} = \frac{n + n_{r21} - E(n) - E(n_{r21})}{l_2 - E(n) - E(n_{r21})}$$

The nPOGR score corrects for chance agreement, and removes the effects of the length of the gene lists and data correlations. Hence, the score may be more appropriate for comparing the consistency levels between differentially expressed gene lists with different lengths and from different studies.

2.3 Study Design

2.3.1 Consistency of DE (differentially expressed) detection methods within a single dataset.

Four datasets of different diseases were used to evaluate the consistency of methods to detect differential expression, based on subsets of the same data (Table 1). Before all the procedures we filtered the raw data to remove invariant transcripts , using the 'nsFilter' command in the 'genefilter' R package.

Transcripts were filtered using the 'var.cutoff' with the value set to 0.5, so that all genes with a variance below 0.5 were removed. This is motivated by the observation that in many tissues only 40% of the genes are expressed. We additionally set the

Data set	Reference	Raw Size Samples × Genes	Used Size Samples × Genes	Analysis(number)	Platform
ALL	Sabina et al., 2004	128×12625	79×4399	BCR/ABL(37), NEG(42)	Affymetrix
ColonCA	Alon et al., 1999	62×2000	62×2000	Tumor(40), Normal(22)	Affymetrix
lungExpression	Beer et al., 2002	86×3171	86×3171	death(24), live(62)	Affymetrix
golubEsets	Golub et al., 1999	72×7129	72×2698	ALL(47), AML(25)	Affymetrix

Table 1. Details of four data sets used to study consistency of DE detection methods within a single data set.

argument 'remove.dupEntrez=TRUE', so that for transcripts mapping to the same Entrez Gene ID, the transcript with the largest variance will be retained and the others removed.

The ALL (Acute Lymphoblastic Leukemia) data from the Ritz Laboratory(Sabina 2004), consists of 128 samples. We selected 79 samples representing two molecular biology subtypes, BCR/ABL (an oncogene fusion protein consisting of BCR and ABL genes) (37 samples) and NEG (42 samples). After screening out invariant expression profiles, the final number of probes in the data set was 4,399. The ColonCA data contains 2,000 expression measurements from 40 colon cancer tumor samples and 22 normal samples (Alon 1999). The lungExpression data (Beer 2002) consists of 3,171 expression measurements from 86 lung cancer patients, of whom 62 were alive at the end of the study and 24 had died. The golubEsets data (Golub 1999)are the combined training samples and test samples. There are 47 patients with acute lymphoblastic leukemia (ALL) and 25 patients with acute myeloid leukemia (AML). The samples were assayed using Affymetrix Hgu6800 chips and data on the expression of 7129 genes (Affymetrix probes) are available. After screening out invariant probes, a total of 2,698 probes remained. All data sets were stored as normalized gene expression format (class exprSet) in

Bioconductor.

To evaluate the consistency of the three DE detection methods, we randomly selected a varying number of subsets without replacement from the full data (Four different numbers of subsets were used for each data set). Here we ignored the possible dependency between samples from the same group (diseased / normal) in the design, implicitly assuming that the samples from within a single group were independent. Next, we used three methods to detect the differentially expressed genes lists based on both the subsets and the full data. The genes were then ranked on the basis of the absolute values of their test statistics. Next, we used different lengths of lists to calculate the scores we defined in section 2.2 (POG, nPOG, POGR and nPOGR). We repeated the three steps above 100 times, and calculate the mean of each of scores, for each combination of method, dataset, subset length, and list length.

2.3.2 Consistency of DE detection methods between data sets

Six datasets for three diseases were analyzed to evaluate the consistency of subsets from different data sets that were collected for the same disease. The prostate cancer cDNA microarray data consists of 62 tumors and 41 normal prostate samples measured for 46,205 clones (Lapointe 2004). After screening out invariant expression profiles, the final number of probes in the data set was 20,540. While the oligo microarray (Affymetrix U95Av2) data contain 52 tumor and 50 non-tumor samples(Singh 2002). After screening out invariant probes, a total of 4,399 probes remained. For lung cancer, the cDNA microarray data consist of 13 squamous cell lung cancer and five normal lung specimens(Garber 2001), and 13,822 probes remained after screening. While the data by Affymetrix human U95A oligonucleotide

Data set	Reference	Raw Size	Used Size	Analysis(number)	Platform

		Samples × Genes	Samples × Genes		
prostate cancer	Lapointe et al., 2004	103×46,205	103×20540	Tumor(62), Normal(41)	cDNA
prostate cancer	Singh et al., 2002	102×12600	102×4399	Tumor(52), Normal(50)	Affymetrix
lung cancer	Garber et al., 2001	123×24000	123×13822	Tumor(67), Normal(56)	cDNA
lung cancer	Bhattacharjee et al., 2001	203×12600	203×4399	Tumor(186), Normal(17)	Affymetrix
Duchenne muscular dystrophy	Haslett et al., 2002	23×12625	23×4399	DMD(12), Normal(11)	Affymetrix
Duchenne muscular dystrophy	Pescatori et al., 2007	37×22283	37×6352	DMD(23), Control(14)	Affymetrix

Table 2. Details of six data sets used to study consistency of DE detectionmethods between different datasets for one disease.

arrays consist of 21 squamous cell lung carcinomas and 17 normal lung specimens(Bhattacharjee 2001). After screening, the final number of probes in the data was 4,399. The two datasets for Duchenne muscular dystrophy (DMD) are based on Affymetrix (Santa Clara, CA) HG-U95Av2 and HG-U133A GeneChips, respectively. One dataset contains 24 samples from 12 DMD patients and 12 unaffected controls(Haslett 2002), and the other consists of 36 samples from 22 DMD patients and 14 controls (Pescatori 2007). After screening, there were 4,399 and 6,352 probes remained for the two data sets. For each disease, we only analyzed the genes that were present on both platforms. The cDNA data were log base 2 transformed and then normalized as median 0 and standard deviation 1 per array. The Affymetrix GeneChip data were normalized by RMA (Robust Multi-array Analysis).

We again used three methods (*t*-test, empirical Bayes and SAM) to detect the the differentially expressed gene lists from each dataset. To begin, we measured overlap genes from the two data sets of one disease. And the overlap genes are the interesting genes which we detected differentially expressed genes from. Second, we randomly selected a varying number of subsets without replacement from the two data sets for one disease (Four different numbers of subsets were used to each data set). We assumed that the samples in the same group were independent. Next, we used

three methods (*t*-test, empirical Bayes and SAM) to detect the differentially expressed gene lists from the two subsets of the two datasets. Genes were subsequently ranked on the basis of the absolute values of their test statistics. And then, we used different lengths of lists to calculate the scores we defined in section2.2 (POG, nPOG, POGR and nPOGR). We repeated the three steps above 100 times, and calculate the mean of each of scores, for each combination of method, dataset, subset length, and list length. We experimented with different cutoffs to determine whether a gene pair was significantly correlated.

CHAPTER III

RESULT

3.1 Consistency of DE detection methods using subsets from the same data set

For the four datasets from the Biocuductor website, we compared the differentially expressed gene lists determined using subsets from the same dataset to test the consistency of the three methods. For the ALL dataset, we obtained various lengths of differentially expressed gene lists using the three methods (*t*-test, empirical Bayes and SAM) for different sample sizes. We separately chose subsets of size 3, 6, 10 and 25 randomly and without replacement each from 37 BCR/ABL and 42 NEG samples (total sample size of 6, 12, 20 and 50 samples, respectively). Test statistics for differential gene expression between the two tissue types for each gene were calculated using each of the three methods. Then, different lengths of gene lists, 50, 100, 500 and 1000, were used to test the overlap of genes with the gene lists taken from the full data.

POG score for ALL					
Method	**length/subset**	**50**	**100**	**500**	**1000**
T-test	**6**	0.070	0.088	0.195	0.299
	10	0.098	0.138	0.260	0.347
	20	0.266	0.327	0.421	0.475
	50	0.578	0.625	0.677	0.696
Empirical Bayes	**6**	0.132	0.168	0.240	0.331
	10	0.152	0.192	0.289	0.365
	20	0.306	0.368	0.439	0.485
	50	0.582	0.642	0.682	0.700
SAM	**6**	0.166	0.187	0.256	0.343
	10	0.167	0.205	0.298	0.373
	20	0.353	0.397	0.455	0.492
	50	0.619	0.695	0.695	0.711

Table 3 POG score for ALL data set

The POG scores for the ALL data are in Table 3. From Table 3 we found that the POG scores were low when the length of the lists and the number of the subsets were small. The POG scores increased with larger subsets or longer length of the gene lists. Then, we discovered from the data that, the values under the *t*-test were generally smaller than the values from other two methods. It was more distinct when the length of the lists and the number of the subsets were small. And the values converged to each other when the length of the lists and the number of the subsets went up.

POGR score for ALL					
Method	**length/subset**	**50**	**100**	**500**	**1000**
T-test	**6**	0.463	0.570	0.821	0.908
	10	0.542	0.642	0.848	0.923
	20	0.826	0.852	0.905	0.946
	50	0.930	0.937	0.956	0.970
Empirical Bayes	**6**	0.665	0.725	0.845	0.919
	10	0.681	0.740	0.864	0.926
	20	0.878	0.874	0.909	0.948
	50	0.946	0.946	0.954	0.969
SAM	**6**	0.690	0.740	0.859	0.928
	10	0.699	0.747	0.869	0.931
	20	0.895	0.882	0.917	0.954
	50	0.963	0.947	0.957	0.975

Table 4. POGR scores for ALL data

The POGR scores for the ALL data are showed in Table 4. The result of POGR score was quite similar to the POG score. The POGR scores are uniformly higher than the POG scores, reflecting the strong correlation between the genes in the data set. However, the effect of subset size, length of list, and choice of DE detection method were all similar to the POG score results.

nPOG score for ALL					
Method	**length/subset**	**50**	**100**	**500**	**1000**
T-test	**6**	0.057	0.066	0.094	0.093
	10	0.086	0.117	0.164	0.154
	20	0.258	0.313	0.347	0.319
	50	0.573	0.616	0.636	0.607
Empirical Bayes	**6**	0.120	0.148	0.144	0.135
	10	0.141	0.172	0.197	0.178
	20	0.298	0.355	0.368	0.332
	50	0.576	0.633	0.642	0.613
SAM	**6**	0.155	0.166	0.162	0.151
	10	0.156	0.185	0.207	0.189
	20	0.345	0.385	0.386	0.342
	50	0.614	0.687	0.656	0.626

Table 5. nPOG scores for ALL data

The nPOG scores for the ALL data are showed in Table 5. From Table 5, we found that the nPOG scores were low when the number of the subsets was small. And the nPOG scores went up when the subsets became larger. But, the nPOG score was not affected by the length of the gene list. Also, we found that, the values under the *t*-test were smaller than the values from the other two methods. It was more distinct when the number of the subsets was small. And the values approached each other when the subsets size increased. So, in general the empirical Bayes and SAM methods performed better than *t*-test.

nPOGR score for ALL					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.047	0.045	0.040	0.025
	10	0.078	0.100	0.113	0.091
	20	0.253	0.300	0.309	0.270
	50	0.567	0.609	0.617	0.584
Empirical Bayes	**6**	0.112	0.129	0.098	0.078
	10	0.131	0.155	0.150	0.121
	20	0.292	0.342	0.331	0.285
	50	0.571	0.626	0.622	0.593
SAM	**6**	0.147	0.151	0.113	0.099
	10	0.147	0.172	0.158	0.137
	20	0.339	0.374	0.349	0.301
	50	0.609	0.683	0.637	0.610

Table 6. nPOGR scores for ALL data.

The nPOGR scores for the ALL data are showed in Table 6. The result for nPOGR score was similar to the nPOG score. The nPOGR scores were all similar in magnitude to the nPOG scores, reflecting the fact that normalization accounts for the strong gene-gene correlations present in this data.

Then, we studied the consistency of relatively small-scaled experiments for colon cancer with 62 samples (40 tumor samples, 22 normal samples). We detected differentially expressed genes by different sample size and length of gene lists using the three DE detection methods. We chose subsets of total size 6, 12, 27 and 51 randomly and without replacement from the 40 tumor samples and 22 normal samples. The subsets were chosen in a 2:1 ratio. Then, we used different length of gene lists, 50, 100, 500 and 1000 to test the overlap between the gene lists taken from the full data and the gene lists based on the subset. The results for the POG, POGR, nPOG and nPOGR scores are given in 'Appendix A', Table 1- 4. Overall, the results are similar

to those from the ALL data, expect that the difference between the three DE detection methods is smaller.

Third, for the lung expression data, it contains 86 samples of whom 62 were alive at the end of the study and 24 had died during the study. We detected differentially expressed genes by different sample sizes and length of gene lists using the three DE detection methods. We chose total subset size of 8, 12, 28 and 48 samples randomly and without replacement each from the complete dataset. The subsets were chosen in a 3:1 ratio from amongst those who lived versus those who died during the follow-up period. Then, we used different length of gene lists, 50, 100, 500 and 1000 to test the overlap genes with the gene lists taken from the full data. The results for the POG, POGR, nPOG and nPOGR scores of lung expression data are given in 'Appendix A', Tables 5- 8 and are similar to the ALL and colon cancer data.

Finally, for the Leukemia data, 47 patients with acute lymphoblastic leukemia (ALL) and 25 patients with acute myeloid leukemia (AML) were monitored in the experiment. We chose total subset size of 6, 12, 27 and 51 samples randomly and without replacement from the complete. The subsets were chosen in a 2:1 ratio of ALL to AML patients. Then, we used different lengths of gene lists, 50, 100, 500 and 1000 to test the overlap between gene lists based on the full data and gene lists based on the subsets. The results for the POG, POGR, nPOG and nPOGR scores of Leukemia data are given in 'Appendix A', Table 9-12. The results were similar to the three datasets above.

Table 7 summarizes the results from the four datasets, and shows the average nPOG scores over the four lengths of the gene lists from the four datasets. Figure 1 plots to the nPOG scores listed in table 7.

nPOG

Methods/Subsets	ALL			
	6	**10**	**20**	**50**
T-test	0.077	0.130	0.309	0.608
eBayes	0.137	0.172	0.338	0.616
SAM	0.159	0.184	0.364	0.646
Methods/Subsets	**Colon Cancer**			
	6	**10**	**20**	**50**
T-test	0.133	0.271	0.499	0.759
eBayes	0.175	0.294	0.510	0.764
SAM	0.182	0.301	0.528	0.775
Methods/Subsets	**Lung Expression**			
	6	**10**	**20**	**50**
T-test	0.051	0.098	0.168	0.332
eBayes	0.069	0.117	0.179	0.337
SAM	0.088	0.137	0.205	0.352
Methods/Subsets	**Leukemia**			
	6	**10**	**20**	**50**
T-test	0.176	0.297	0.572	0.762
eBayes	0.228	0.326	0.583	0.767
SAM	0.253	0.349	0.626	0.803

Table 7. The summarize of the nPOG scores for four data sets used to study consistency of DE detection methods within a single data set.

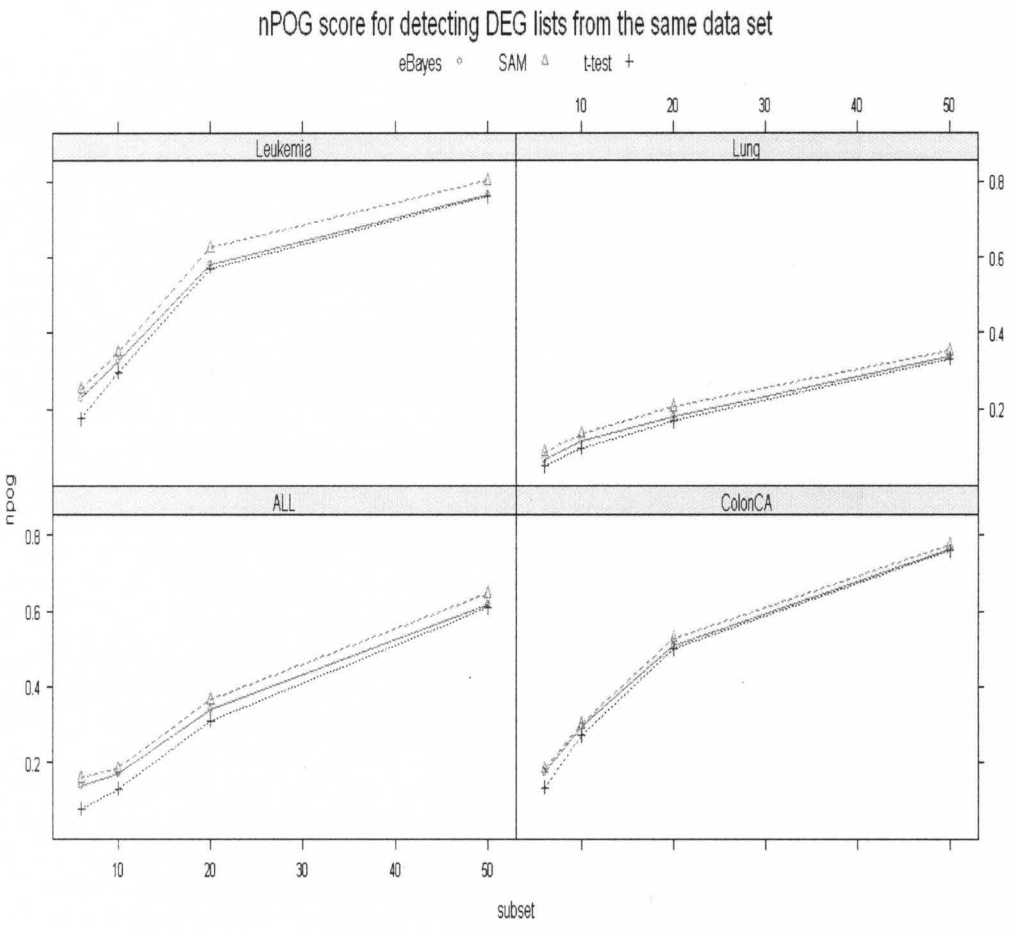

Figure 1. The figure of the nPOG scores for four data sets used to study consistency of DE detection methods within a single data set.

Table 8 summarizes the results from the four datasets, and shows the average nPOGR scores over the four lengths of the gene lists from the four datasets. Figure 2 plots to the nPOGR scores listed in table 8 and the cutoff = 0.8.

The nPOGR score (Table 8)

nPOGR(cutoff=0.8)

Methods/Subsets	**ALL**			
	6	**10**	**20**	**50**
T-test	0.039	0.096	0.283	0.594
eBayes	0.104	0.139	0.313	0.603
SAM	0.128	0.153	0.341	0.635
Methods/Subsets	**Colon Cancer**			
	6	**10**	**20**	**50**
T-test	0	0	0.431	0.773
eBayes	0	0.030	0.449	0.783
SAM	0	0.048	0.473	0.784
Methods/Subsets	**Lung Expression**			
	6	**10**	**20**	**50**
T-test	0	0	0.061	0.249
eBayes	0	0.014	0.077	0.257
SAM	0	0.048	0.122	0.286
Methods/Subsets	**Leukemia**			
	6	**10**	**20**	**50**
T-test	0.392	0.515	0.708	0.851
eBayes	0.455	0.543	0.718	0.854
SAM	0.487	0.575	0.754	0.878

Table 8. The average of the nPOGR scores for four data sets used to study consistency of DE detection methods within a single data set.

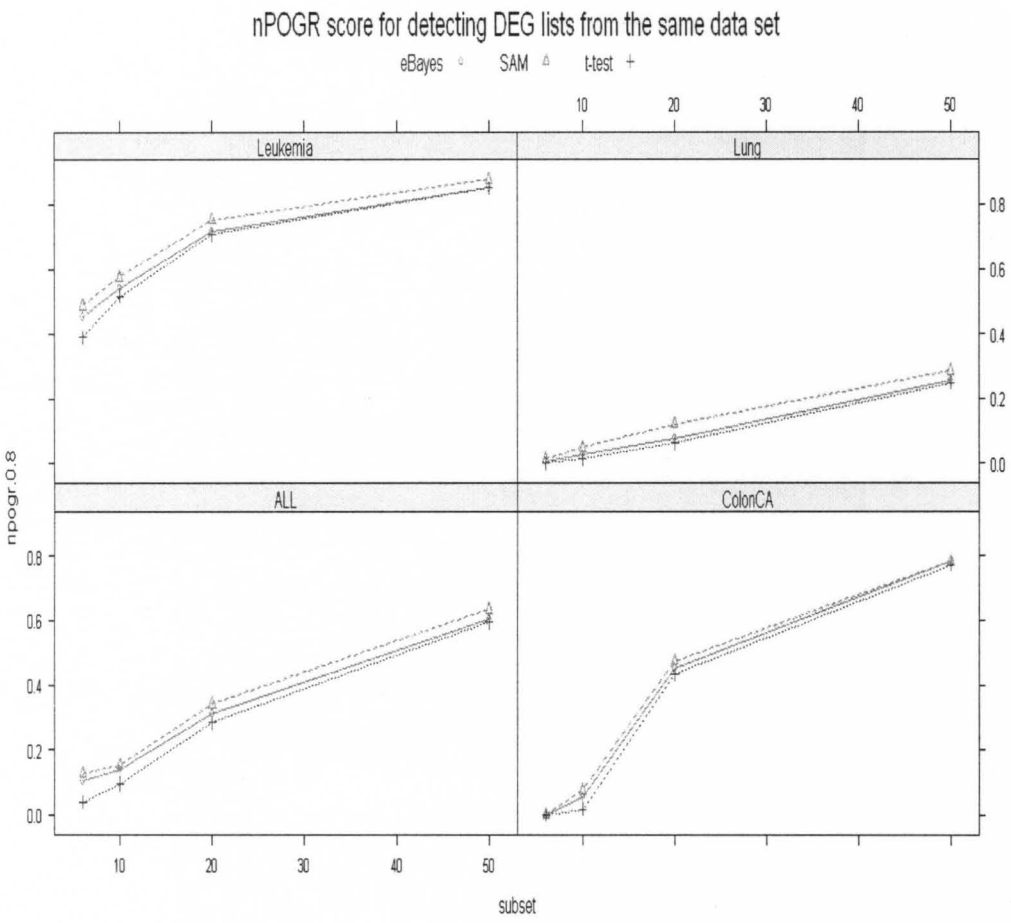

Figure 2. The plot to the nPOGR scores for four data sets in Table 8.

3.2 Consistency of DE detection methods from different data sets of the same disease.

For the six datasets of three diseases (prostate cancer, lung cancer, and DMD), we compared the differentially expressed gene lists from the two datasets for the same disease to test the consistency of the three DE detection methods.

First, for the prostate cancer data, here are 103 samples in the cDNA microarray data, which are divided into two parts, 62 tumor samples and 41 normal prostate samples. For the Affymetrix microarray data, there are 52 tumor and 50 non-tumor samples. A total of 2262 genes were found to be in common between both platforms. We considered these 2262 genes to be the complete data in both to cases, as the whole gene lists of interest. Then, we randomly picked subset of 3, 5, 10 and 25 samples from the tumor and normal tissue samples for each of the two data sets. Then, we separately used three methods to detect the differentially expressed gene lists from two subsets of the full data. Third, we use different lengths of lists 50, 100, 500, 1000 to calculate the POG, POGR, nPOG and nPOGR scores. We repeated the same procedure for 100 times, and calculated the mean of the 100 scores.

nPOG score for Prostate Cancer					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.014	0.019	0.033	0.023
	10	0.036	0.036	0.031	0.023
	20	0.076	0.090	0.083	0.040
	50	0.085	0.088	0.085	0.057
Empirical Bayes	**6**	0.039	0.049	0.056	0.027
	10	0.049	0.049	0.041	0.034
	20	0.086	0.103	0.090	0.042
	50	0.084	0.093	0.101	0.066
SAM	**6**	0.031	0.041	0.054	0.028
	10	0.043	0.046	0.040	0.039
	20	0.083	0.101	0.090	0.043
	50	0.082	0.083	0.089	0.070

Table 9. nPOG score for Prostate Cancer

nPOGR 0.8 score for Prostate Cancer									
		From cDNA to Affymetrix				**From Affymetrix to cDNA**			
Method	**subset/length**	**50**	**100**	**500**	**1000**	**50**	**100**	**500**	**1000**
T-test	**6**	0.087	0.071	0.033	0.005	0.082	0.062	0.023	0.001
	10	0.136	0.104	0.037	0.007	0.116	0.092	0.041	0.006
	20	0.15	0.114	0.059	0.019	0.122	0.105	0.049	0.011
	50	0.351	0.217	0.099	0.039	0.333	0.204	0.089	0.026
Empirical	**6**	0.194	0.188	0.081	0.009	0.168	0.188	0.186	0.128
Bayes	**10**	0.234	0.212	0.121	0.034	0.177	0.19	0.179	0.137
	20	0.394	0.405	0.223	0.087	0.188	0.201	0.193	0.142
	50	0.531	0.511	0.272	0.085	0.341	0.301	0.27	0.171
SAM	**6**	0.199	0.183	0.083	0.014	0.145	0.165	0.175	0.122
	10	0.225	0.199	0.113	0.029	0.224	0.228	0.183	0.108
	20	0.366	0.385	0.211	0.073	0.243	0.272	0.242	0.155
	50	0.474	0.492	0.246	0.061	0.262	0.301	0.364	0.254

Table 10. nPOGR score for Prostate Cancer with the cutoff= 0.8.

The nPOG and nPOGR scores for the prostate cancer data are showed in Table 9 and 10 respectively. From Table 9, the nPOG scores were very low for all subset sizes, indicating low consistency between the two data sets, irrespective of the DE detective

method. From Table 10, the nPOGR scores were generally higher than the nPOG scores, although there appears to be an anomaly in that the nPOGR scores decrease with the length of the list, contrary to what was expected. In both cases, the scores increased when the subsets became larger. Also, we found that, the values under the *t*-test were smaller than the values from the other two methods. It was more distinct when the number of the subsets was small. The values converged to each other when the number of the subsets increased.

For the lung cancer data, there are 18 samples in the cDNA microarray data, which are divided into two parts, 13 tumor samples and 5 normal prostate samples. For the Affymetrix microarray data, there are 21 tumor and 17 non-tumor samples. A total of 1893 genes were found to be in common between the both platforms. We considered these 1893 genes to be the complete data in both to cases. Then, we randomly picked subsets of 4, 8, 12 and 16 samples from the tumor and normal tissue samples for each of the two data sets. For cDNA data, the subsets were chosen in a 3:1 ratio of tumor to normal samples. For Affymetrix data the subsets were chosen in a 1:1 ratio of tumor to normal samples. Then, we separately used three methods to detect the differentially expressed gene lists from two subsets of the full data. Third, we used different lengths of lists 50, 100, 500, 1000 to calculate the scores. At last, we repeated the same procedure 100 times, and calculated the mean of the 100 scores. The results for the nPOG and nPOGR scores of lung cancer data are given in 'Appendix A', Table 13-14.The scores are generally higher compared to the prostate cancer data, and the differences between the *t*-test and the other two methods are more pronounced, for the nPOG score. The rest results are similar to the prostate cancer data sets.

For the duchenne muscular dystrophy (DMD) data, there are 24 samples in the first Affymetrix data, which are divided into 12 DMD patient samples and 12 control samples (Haslett *et al.*, 2002). For the other Affymetrix there are 22 DMD patient samples and 14 control samples (Pescatori *et al.*, 2007). A total of 3158 genes were founded to be common between both data sets. We considered these 3158 genes to be the complete data in both data sets. Then, we randomly picked subsets of 6, 8, 12 and 24 samples for each of the data sets without replacement. For Haslett's data, the subsets were chosen in a ratio of 1:1 between patients and normal samples, and the subsets were selected in a ratio of 2:1 between patients and controls in Pescatori's data. Then, we separately used three methods to detect the differentially expressed gene lists from two subsets of the full data. Third, we used different lengths of lists 50, 100, 500, 1000 to calculate the POG, POGR, nPOG and nPOGR scores. We repeated the procedure 100 times, and calculated the mean of the 100 scores. The results for the nPOG and nPOGR scores of lung cancer data are given in 'Appendix A', Table 15-16.The result was almost same to the lung cancer data sets.

Table 11 and 12 give the average nPOG and nPOGR scores for each of the three data sets, covered over the four subsets sizes. Figure 3 plots the nPOG scores from Table 11, while Figures 5 and 6 plot the nPOGR scores from Table 12.

Methods/Subsets	Prostate Cancer			
	6	10	20	50
T-test	0.022	0.032	0.072	0.079
eBayes	0.043	0.043	0.080	0.086
SAM	0.039	0.042	0.079	0.081
Methods/Subsets	Lung Cancer			
	4	8	12	16
T-test	0.144	0.152	0.195	0.224
eBayes	0.205	0.208	0.240	0.260
SAM	0.210	0.214	0.248	0.273
Methods/Subsets	DMD			
	6	12	18	24
T-test	0.119	0.265	0.342	0.400
eBayes	0.251	0.335	0.389	0.447
SAM	0.240	0.335	0.395	0.449

Table 11. The summarize of the nPOG scores for six data sets used to study consistency of DE detection methods between different data sets of the same disease.

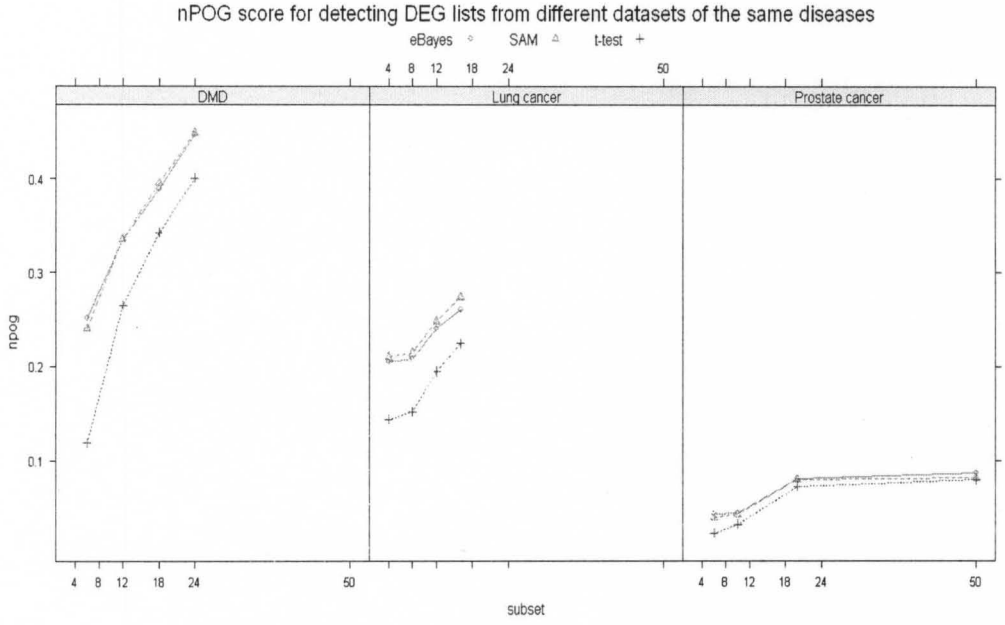

Figure 3. The plot of the nPOG scores for six data sets in Table 11.

nPOGR 0.75

Methods/Subsets	From Data A to Data B				From Data B to Data A			
	Prostate Cancer				Prostate Cancer			
	6	10	20	50	6	10	20	50
T-test	0.049	0.071	0.086	0.177	0.042	0.064	0.072	0.163
eBayes	0.118	0.15	0.277	0.35	0.168	0.171	0.181	0.271
SAM	0.12	0.142	0.259	0.318	0.152	0.186	0.228	0.295
Methods/Subsets	Lung Cancer				Lung Cancer			
	4	8	12	16	4	8	12	16
T-test	0.064	0.081	0.139	0.196	0.081	0.099	0.148	0.196
eBayes	0.332	0.339	0.378	0.406	0.286	0.294	0.34	0.358
SAM	0.339	0.345	0.387	0.409	0.292	0.302	0.34	0.359
Methods/Subsets	DMD				DMD			
	6	12	18	24	6	12	18	24
T-test	0.158	0.185	0.213	0.263	0.16	0.191	0.254	0.285
eBayes	0.472	0.545	0.585	0.61	0.485	0.565	0.613	0.656
SAM	0.461	0.543	0.585	0.603	0.466	0.558	0.612	0.655

Table 12. The summarize of the nPOGR scores for six data sets used to study consistency of DE detection methods between different data sets of the same disease, with the cutoff= 0.75.

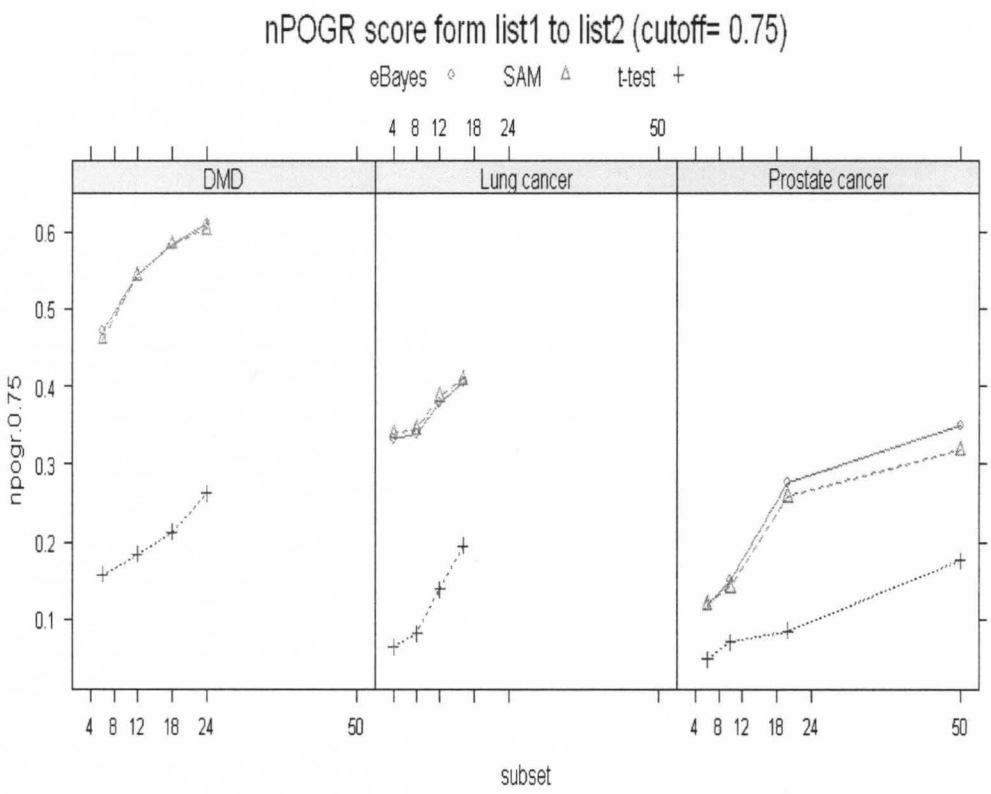

Figure 4. The plot of the nPOGR scores getting from list 1 to list 2 for six data sets in Table 12

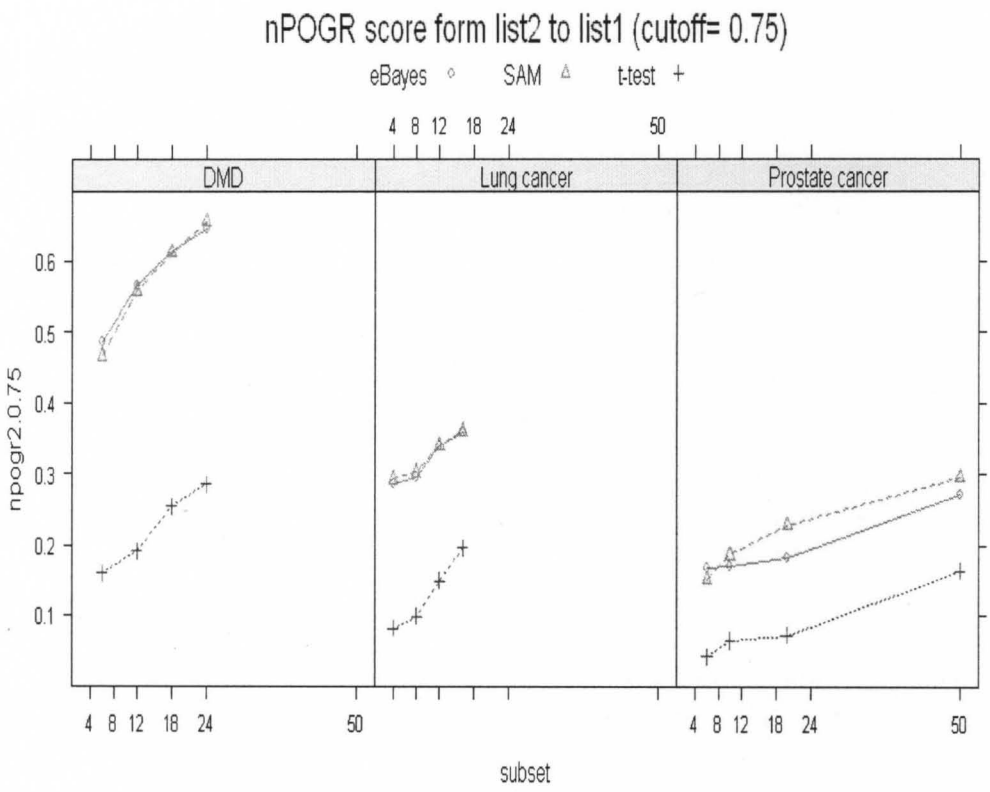

Figure 5. The plot of the nPOGR scores from list 2 to list 1 for six data sets in Table 12.

CHAPTER IV

DISCUSSION

In our article, three distinct methods were used for selecting differentially expressed genes, the *t*-test, empirical Bayes and SAM. The calculation of the test statistic for each method captures different aspects of the changes in expression measurements for each gene, resulting in potentially different lists of DE genes. The traditional *t*-test still remains a popular choice, and we compared the *t*-test with the empirical Bayes and SAM methods to evaluate the empirical impact of choice of test-statistic on the consistency of the resulting list of differentially expressed genes. Of particular interest was the consistency of each method when the available sample size was small.

Based on our results, the traditional *t*-statistic had the poorest performance relative to SAM and the empirical Bayesian methods. This was particularly evident in the studies involving two data sets for the same disease, although less evident for the studies involving subsets of a single data set. The *t*-statistic can be unstable in small data sets, because the variance estimate can be quite skewed by outlying expression values. Further, due to the large numbers of genes in microarray datasets, there will always be some genes with low standard deviation by chance. In this case,

these genes will have large t-statistics and would be considered to be differentially expressed falsely. In this case, the empirical Bayes and SAM statistics, which can be considered as modified *t*-statistic methods, gave better results.

When the length of the gene lists increased, the POG and POGR scores generally increased. But for the nPOG and nPOGR scores, the values of the scores did not change uniformly with the length of the lists. and the degree of change was smaller than for the POG and POGR scores. So, the POG and POGR scores are more strongly affected by the length of the gene list, and the nPOG and nPOGR scores are more appropriate to evaluate the consistency of DE detection methods with different lengths of lists. Next, we found that all the scores increased when the size of the subsets increased. We can conclude that the sample size is important when evaluating the consistency of the DE detection methods. In the first part of the experiment, we compared the differentially expressed gene lists from different subsamples with the full data in the same study to avoid any platform and site differences. According to our result, the differentially expressed gene lists can still be very inconsistent, especially when the subsets were small.

The DE gene lists from small-scaled microarray studies may only contain a small portion of the total number of differentially expressed genes in a disease, so the POG scores will generally be low. However, each differentially expressed gene list may include mostly true genes. Usually, the global expression changes of genes in a disease may introduce great uncertainty to the findings at the individual gene level (Jeffery 2006). In this case, even if there are many 'true' differentially expressed genes in a disease, we may be only discovering the most significant ones. Thus, the POGR score is helpful for measuring how well different methods for detecting DE genes retain the biologically important genes, particularly when considering small

subsets of the samples. As the result, when the sample size of the microarray study is small, the nPOGR score may give the best result.

According to the result we got, we used the nPOG and nPOGR scores to evaluate the efficacy of the methods. These three methods were then applied to ten real microarray data sets. Our results indicated that SAM was the most consistent method in the analysis of all the data sets, although the empirical Bayes method performed similarly to SAM. When the sample size of the subset was small (3 arrays per group), the resulting gene list based on all three mehods had a low consistency with the gene list based on the full data, although the empirical Bayes and SAM methods had higher consistencies relative to the *t*-test.

This disparity between the methods is more apparent when considering the studies using two different data sets for the same disease. In this case, both the SAM and empirical Bayes methods had considerably higher consistencies relative to the t-test. In light of all the evidence, we recommend that investigators use these moderated versions in lieu of the *t*-test when the sample size is small (less than 5 per group).

There are several limitations with our study that are planned as areas of future research. First, our findings suggest that the moderated versions of the *t*-test are more consistent than the traditional *t*-test for microarray studies. One aspect which we did not explicitly evaluate is how robust each of the methods are to departures from normality. Both SAM and the empirical Bayes *t*-test are moderated by adding a strictly positive constant to the denominator of the ordinary *t*-statistic. The addition of this constant reduces the chance of 'detecting' genes (declaring them to be statistically significant) which have a low standard deviation by chance. Since common departures from normality (skewness and/or outliers) would inflate the sample variance, this suggests that the moderated versions of the *t*-test would be more robust in this case.

However, an explicit study involving non-normal real and/or simulated data would be needed to test this hypothesis.

In our study design we ignored design factors which could cause dependencies between samples from the same phenotypic group (diseased / control). This may have an impact on the results, and in particular a sampling design which accounts for the design factors in the study may be more appropriate. Additionally, using a more complicated model for the gene expression values, which includes the original design variables in the model, would be worthwhile to investigate. In this case, a comparison between the standard linear model and the hierarchical empirical Bayesian model (Smyth 2004) would be warranted. In subsequent research, we will check the experimental design for each data set, to determine whether this should be accounted for when taking subsamples from the data.

In our paper, we randomly chose subsets from the full data set and evaluated consistency between the DE detection methods using fixed list lengths. The smallest subset size we used was 6 (3 in each group), and sample sizes this small will have correspondingly low power to detect DE genes. Since we used fixed list lengths when evaluating the consistency for each method, our scores will only reflect the consistency based on the *test statistic* itself, and ignores the ability of each method to retain power when the sample size is small. Subsequent studies which compare lists based on a significance threshold would be needed to test whether any differences between the methods existed on this account.

Lastly, in our research we evaluated the consistency of the DE detection methods using only real data. To supplement these experiments, it would be interesting to use simulation methods (Parrish 2009) to create simulated data which reflect the real microarray data but with a larger number of samples. This would

enable us to better determine a 'true' list of DE genes, and to better map the rate of convergence between the methods as the sample size increases.

REFERENCES

Alon, U., N, Baska., D, A, Notterman., K,Gish., S,Ybarra., D,Mack., And A,J, Levine (1999). "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays." Proc. Natl. Acad. Sci. USA **96**: 6745–6750.

Baldi, P., Anthony, D. Long (2001). "A Bayesian framework for the analysis of microarray expression data: regularized t -test and statistical inferences of gene changes." Bioinformatics **17**(6): 509–519

Beer, D. G., Kardia, Sharon L. R., Huang, Chiang-Ching., Giordano, Thomas J., Levin, Albert M., Misek, David E., Lin, Lin., Chen, Guoan., Gharib, Tarek G., Thomas, Dafydd G., Lizyness, Michelle L., Kuick, Rork., Hayasaka, Satoru., Taylor, Jeremy M. G., Iannettoni, Mark D., Orringer, Mark B., Hanash, Samir (2002). "Gene-expression profiles predict survival of patients with lung adenocarcinoma." Nature Medicine **8**(8): 816-824.

Bhattacharjee, A. (2001). "Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses." Proceedings of the National Academy of Sciences **98**(24): 13790-13795.

Chen, J. J., Hsueh, Huey-Miin., Delongchamp, Robert R., Lin, Chien-Ju., Tsai, Chen-An (2007). "Reproducibility of microarray data: a further analysis of microarray quality control (MAQC) data." BMC Bioinformatics **8**(1): 412.

Efron, B. (2009). "Empirical Bayes Estimates for Large-Scale Prediction Problems." Journal of the American Statistical Association **104**(487): 1015-1028.

Garber, M. E. (2001). "Diversity of gene expression in adenocarcinoma of the lung." Proceedings of the National Academy of Sciences **98**(24): 13784-13789.

Golub, T., R., D, K, Slonim., P, Tamayo., C, Huard., M, Gaasenbeek., J, P, Mesirov., H, Coller., M,L, Loh., J, R, Downing., M, A,Caligiuri., C, D, Bloomfield., E, S,Lander. (1999). "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring." Science **286**(5439): 531-537.

Haslett, J. N. (2002). "Gene expression comparison of biopsies from Duchenne muscular dystrophy (DMD) and normal skeletal muscle." Proceedings of the National Academy of Sciences **99**(23): 15000-15005.

Hong, F. and R. Breitling (2008). "A comparison of meta-analysis methods for detecting differentially expressed genes in microarray experiments." Bioinformatics **24**(3): 374-382.

Jeffery, I. B., Higgins, Desmond G., Culhane, Aedín C. (2006). "Comparison and evaluation of methods for generating differentially expressed gene lists from microarray data." BMC Bioinformatics **7**(1): 359.

Lapointe, J. (2004). "Gene expression profiling identifies clinically relevant subtypes of prostate cancer." Proceedings of the National Academy of Sciences **101**(3): 811-816.

McCarthy, D. J. and G. K. Smyth (2009). "Testing significance relative to a fold-change threshold is a TREAT." Bioinformatics **25**(6): 765-771.

Parrish, R. S., Spencer Iii, Horace J., Xu, Ping (2009). "Distribution modeling and simulation of gene expression data." Computational Statistics & Data Analysis **53**(5): 1650-1660.

Pescatori, M., Broccolini, A., Minetti, C., Bertini, E., Bruno, C., D'Amico, A., Bernardini, C., Mirabella, M., Silvestri, G., Giglio, V., Modoni, A., Pedemonte, M., Tasca, G., Galluzzi, G., Mercuri, E., Tonali, P. A., Ricci, E. (2007). "Gene expression profiling in the early phases of DMD: a constant molecular signature characterizes DMD muscle from early postnatal life throughout disease progression." The FASEB Journal **21**(4): 1210-1226.

Sabina, C., Xiaochun, Li., Robert, Gentleman., Antonella, Vitale., Marco, Vignetti., Franco, Mandelli.,Jerome, Ritz., and Robin, Foa. (2004). "Gene expression profile of adult T-cell acute lymphocytic leukemia identifies distinct subsets of patients with different response to therapy and survival." Blood **103**(7): 2771-2778.

Shi, L., Reid, Laura H.. Jones, Wendell D., Shippy, Richard., Warrington, Janet A., Baker, Shawn C., Collins, Patrick J., de Longueville, Francoise. (2006). "The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements." Nature Biotechnology **24**(9): 1151-1161.

Singh, D., Phillip, G, Febbo., Kenneth, Ross., Donald, G, Jackson., Judith, Manola., Christine, Ladd., Pablo, Tamayo., Andrew ,A, Renshaw., Anthony, V, D'Amico., Jerome, P, Richie., Eric, S, Lander., Massimo, Loda., Philip, W, Kantoff., Todd, R, Golub., William, R, Sellers. (2002). "Gene expression correlates of clinical prostate cancer behavior." Cancer Cell. 2002 Mar;1(2):203-9.

Smyth, G. (2004). "Linear models and empirical Bayes methods for assessing differential expression in Microarray Experiments." Stat Appl Genet Mol Biol **3**(1): Article 3.

Smyth, G. (2005). "Limma: Linear Models for Microarray Data." Bioinformatics and Computational Biology Solutions using R and Bioconductor. Edited by Gentleman R, Carey V, Dudoit S, Irizarry R, Huber W, New York: Springer: 397-420.

Smyth, G. K., Michaud, J., Scott, H. S. (2005). "Use of within-array replicate spots for assessing differential expression in microarray experiments." Bioinformatics **21**(9): 2067-2075.

Tu, Y. (2002). "Quantitative noise analysis for gene expression microarray experiments." Proceedings of the National Academy of Sciences **99**(22): 14031-14036.

Tusher, V., Goss., Robert, Tibshirani., Gilbert Chu (2001). "Significance analysis of microarrays applied to the ionizing radiation response." Proceedings of the National Academy of Sciences **98**(9): 5116-5121.

Xu, P., Brock, Guy N., Parrish, Rudolph S. (2009). "Modified linear discriminant analysis approaches for classification of high-dimensional microarray data." Computational Statistics & Data Analysis **53**(5): 1674-1687.

Zhang, M., Zhang, L., Zou, J., Yao, C., Xiao, H., Liu, Q., Wang, J., Wang, D., Wang, C., Guo, Z. (2009). "Evaluating reproducibility of differential expression discoveries in microarray studies by considering correlated molecular changes." Bioinformatics **25**(13): 1662-1668.

APPENDIX A

Tables

POG score for Colon Cancer					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.140	0.170	0.380	0.560
	10	0.270	0.298	0.489	0.627
	20	0.502	0.524	0.663	0.729
	50	0.761	0.784	0.841	0.860
Empirical Bayes	**6**	0.175	0.223	0.416	0.574
	10	0.299	0.318	0.507	0.635
	20	0.518	0.540	0.668	0.731
	50	0.770	0.790	0.844	0.861
SAM	**6**	0.179	0.234	0.420	0.577
	10	0.301	0.332	0.515	0.637
	20	0.538	0.568	0.677	0.736
	50	0.793	0.797	0.847	0.865

Table 1

POGR score for Colon Cancer					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.410	0.512	0.807	0.907
	10	0.518	0.633	0.882	0.935
	20	0.796	0.829	0.930	0.956
	50	0.934	0.944	0.973	0.978
Empirical Bayes	**6**	0.482	0.599	0.832	0.916
	10	0.578	0.659	0.890	0.936
	20	0.810	0.841	0.931	0.955
	50	0.939	0.952	0.973	0.978
SAM	**6**	0.495	0.602	0.833	0.914
	10	0.597	0.670	0.892	0.935
	20	0.832	0.858	0.933	0.954
	50	0.958	0.949	0.973	0.977

Table 2

nPOG score for Colon Cancer					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.118	0.124	0.170	0.121
	10	0.249	0.261	0.319	0.255
	20	0.489	0.499	0.551	0.457
	50	0.755	0.772	0.787	0.720
Empirical Bayes	**6**	0.154	0.181	0.218	0.149
	10	0.280	0.282	0.343	0.270
	20	0.505	0.516	0.558	0.462
	50	0.765	0.778	0.790	0.723
SAM	**6**	0.158	0.192	0.224	0.154
	10	0.282	0.296	0.354	0.274
	20	0.525	0.546	0.569	0.473
	50	0.788	0.786	0.795	0.729

Table 3

nPOGR score for Colon Cancer					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0	0	0	0
	10	0	0	0.074	0
	20	0.560	0.496	0.437	0.232
	50	0.859	0.836	0.782	0.614
Empirical Bayes	**6**	0	0	0	0
	10	0.103	0	0.130	0
	20	0.589	0.531	0.446	0.228
	50	0.871	0.859	0.787	0.612
SAM	**6**	0	0	0	0
	10	0.144	0.021	0.146	0
	20	0.638	0.583	0.462	0.211
	50	0.910	0.851	0.781	0.596

Table 4

POG score for Lung Expression					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.064	0.079	0.209	0.346
	10	0.099	0.127	0.253	0.378
	20	0.182	0.204	0.311	0.416
	50	0.341	0.364	0.455	0.519
Empirical Bayes	**6**	0.088	0.105	0.220	0.352
	10	0.130	0.156	0.261	0.382
	20	0.202	0.218	0.315	0.417
	50	0.351	0.374	0.456	0.518
SAM	**6**	0.102	0.123	0.237	0.367
	10	0.151	0.180	0.273	0.393
	20	0.232	0.254	0.336	0.427
	50	0.377	0.388	0.462	0.528

Table 5

POGR score for Lung Expression					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.065	0.080	0.258	0.443
	10	0.100	0.128	0.302	0.475
	20	0.183	0.206	0.346	0.499
	50	0.343	0.368	0.487	0.593
Empirical Bayes	**6**	0.090	0.108	0.272	0.450
	10	0.132	0.158	0.311	0.481
	20	0.204	0.221	0.353	0.505
	50	0.353	0.381	0.489	0.594
SAM	**6**	0.104	0.132	0.304	0.474
	10	0.154	0.188	0.337	0.502
	20	0.234	0.260	0.391	0.530
	50	0.379	0.398	0.510	0.614

Table 6

nPOG score for Lung Expression					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.050	0.050	0.059	0.043
	10	0.086	0.099	0.115	0.093
	20	0.166	0.178	0.182	0.147
	50	0.330	0.343	0.354	0.299
Empirical Bayes	**6**	0.074	0.077	0.073	0.052
	10	0.118	0.129	0.125	0.098
	20	0.187	0.192	0.187	0.149
	50	0.341	0.353	0.355	0.297
SAM	**6**	0.089	0.095	0.093	0.074
	10	0.139	0.154	0.139	0.115
	20	0.218	0.229	0.211	0.164
	50	0.367	0.367	0.361	0.311

Table 7

nPOGR score for Lung Expression					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0	0	0	0
	10	0.036	0.016	0	0
	20	0.122	0.097	0.020	0.006
	50	0.294	0.284	0.228	0.190
Empirical Bayes	**6**	0.023	0	0	0
	10	0.070	0.050	0	0
	20	0.144	0.114	0.031	0.017
	50	0.305	0.299	0.231	0.192
SAM	**6**	0.038	0.017	0	0
	10	0.093	0.084	0.004	0.011
	20	0.176	0.159	0.088	0.066
	50	0.333	0.318	0.263	0.230

Table 8

POG score for Leukimia					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.177	0.204	0.340	0.485
	10	0.280	0.340	0.456	0.547
	20	0.582	0.591	0.662	0.718
	50	0.750	0.784	0.817	0.844
Empirical Bayes	**6**	0.246	0.272	0.378	0.497
	10	0.328	0.378	0.470	0.552
	20	0.602	0.608	0.667	0.721
	50	0.759	0.791	0.818	0.845
SAM	**6**	0.266	0.294	0.408	0.511
	10	0.366	0.404	0.487	0.556
	20	0.657	0.672	0.698	0.729
	50	0.823	0.833	0.839	0.850

Table 9

POGR score for Leukimia					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.549	0.552	0.650	0.737
	10	0.657	0.684	0.712	0.768
	20	0.834	0.816	0.821	0.845
	50	0.917	0.909	0.906	0.921
Empirical Bayes	**6**	0.632	0.616	0.677	0.746
	10	0.706	0.711	0.720	0.771
	20	0.843	0.836	0.824	0.846
	50	0.921	0.917	0.906	0.921
SAM	**6**	0.623	0.652	0.706	0.761
	10	0.708	0.749	0.746	0.783
	20	0.831	0.896	0.853	0.856
	50	0.914	0.954	0.927	0.927

Table 10

nPOG score for Leukimia					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.160	0.172	0.190	0.182
	10	0.265	0.312	0.330	0.281
	20	0.574	0.576	0.585	0.552
	50	0.745	0.776	0.775	0.753
Empirical Bayes	**6**	0.231	0.242	0.237	0.202
	10	0.314	0.352	0.348	0.289
	20	0.593	0.593	0.592	0.556
	50	0.755	0.783	0.777	0.755
SAM	**6**	0.250	0.266	0.273	0.223
	10	0.353	0.380	0.369	0.295
	20	0.650	0.659	0.629	0.568
	50	0.820	0.826	0.803	0.762

Table 11

nPOGR score for Leukimia					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.486	0.441	0.352	0.287
	10	0.612	0.610	0.466	0.372
	20	0.812	0.773	0.667	0.579
	50	0.906	0.886	0.825	0.786
Empirical Bayes	**6**	0.581	0.522	0.403	0.313
	10	0.668	0.644	0.481	0.380
	20	0.823	0.797	0.672	0.581
	50	0.910	0.896	0.825	0.785
SAM	**6**	0.571	0.567	0.455	0.354
	10	0.670	0.690	0.530	0.412
	20	0.809	0.871	0.727	0.608
	50	0.902	0.943	0.863	0.803

Table 12

nPOG score for lung Cancer					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**4**	0.083	0.110	0.204	0.179
	8	0.094	0.122	0.201	0.190
	12	0.112	0.159	0.266	0.242
	16	0.130	0.193	0.303	0.271
Empirical Bayes	**4**	0.166	0.209	0.255	0.191
	8	0.167	0.209	0.256	0.201
	12	0.177	0.231	0.299	0.251
	16	0.188	0.248	0.321	0.284
SAM	**4**	0.166	0.218	0.261	0.194
	8	0.166	0.221	0.263	0.204
	12	0.183	0.247	0.308	0.254
	16	0.202	0.270	0.330	0.290

Table 13

nPOGR 0.75 score for Lung Cancer									
		From cDNA to Affymetrix				**From Affymetrix to cDNA**			
Method	**subset/length**	**50**	**100**	**500**	**1000**	**50**	**100**	**500**	**1000**
T-test	**4**	0.172	0.041	0.033	0.009	0.191	0.081	0.042	0.010
	8	0.212	0.054	0.047	0.012	0.232	0.094	0.050	0.019
	12	0.311	0.153	0.058	0.034	0.321	0.173	0.058	0.041
	16	0.377	0.214	0.107	0.086	0.365	0.218	0.107	0.092
Empirical	**4**	0.465	0.416	0.267	0.178	0.342	0.340	0.272	0.190
Bayes	**8**	0.472	0.429	0.265	0.188	0.354	0.353	0.272	0.198
	12	0.512	0.457	0.304	0.237	0.406	0.386	0.318	0.248
	16	0.550	0.477	0.325	0.271	0.401	0.409	0.340	0.283
SAM	**4**	0.468	0.436	0.272	0.181	0.345	0.353	0.278	0.191
	8	0.466	0.435	0.271	0.190	0.361	0.367	0.278	0.200
	12	0.519	0.475	0.312	0.240	0.396	0.388	0.325	0.250
	16	0.538	0.487	0.332	0.277	0.389	0.412	0.348	0.288

Table 14

nPOG score for DMD					
Method	**subset/length**	**50**	**100**	**500**	**1000**
T-test	**6**	0.061	0.085	0.163	0.165
	12	0.190	0.246	0.327	0.296
	18	0.256	0.333	0.404	0.376
	24	0.296	0.429	0.455	0.422
Empirical Bayes	**6**	0.242	0.261	0.278	0.221
	12	0.297	0.348	0.375	0.319
	18	0.319	0.410	0.432	0.394
	24	0.377	0.496	0.485	0.431
SAM	**6**	0.233	0.247	0.265	0.214
	12	0.306	0.346	0.372	0.318
	18	0.336	0.418	0.432	0.395
	24	0.383	0.492	0.485	0.436

Table 15

nPOGR 0.75 score for DMD Cancer									
		From Haslett to Pescatori				**From Pescatori to Haslett**			
Method	**subset/length**	**50**	**100**	**500**	**1000**	**50**	**100**	**500**	**1000**
T-test	**6**	0.178	0.169	0.147	0.138	0.185	0.174	0.141	0.138
	12	0.247	0.232	0.143	0.117	0.263	0.222	0.163	0.117
	18	0.383	0.188	0 181	0.099	0.397	0.268	0.221	0.129
	24	0.377	0.299	0.216	0.159	0.407	0.312	0.245	0.174
Empirical	**6**	0.601	0.598	0.422	0.267	0.636	0.604	0.423	0.276
Bayes	**12**	0.688	0.678	0.469	0.344	0.736	0.691	0.487	0.344
	18	0.714	0.711	0.504	0.409	0.783	0.740	0.531	0.399
	24	0.725	0.749	0.530	0.437	0.803	0.775	0.570	0.435
SAM	**6**	0.591	0.586	0.406	0.261	0.614	0.582	0.405	0.262
	12	0.687	0.674	0.466	0.345	0.730	0.681	0.479	0.340
	18	0.717	0.707	0.506	0.409	0.774	0.743	0.530	0.402
	24	0.697	0.747	0.529	0.440	0.812	0.793	0.571	0.442

Table 16

APPENDIX B

R code for 3.1

ALL data

```
## Load packages
library(limma)
library(ALL)
library(genefilter)
library(multtest)
library(samr)

## 1. Get ALL data, compare BRC/ABL with NEG for mol.biol
data(ALL)
subset <- intersect(which(ALL$mol.biol%in%c("BCR/ABL","NEG")),
                    grep("^B", as.character(ALL$BT)))
## selects 79 samples
all2 <- ALL[ ,subset]   #Note: ALL data is log2 transformed
## table(pData(all2)$mol.biol)
## ALL1/AF4   BCR/ABL E2A/PBX1         NEG     NUP-98   p15/p16
##         0        37         0        42         0         0
filt.all2 <- nsFilter(all2, require.entrez=FALSE,
          require.GOBP=FALSE, remove.dupEntrez=TRUE,
          feature.exclude="^AFFX", var.cutoff=0.5)
filt.all2 <- filt.all2$eset
dim(filt.all2)
all2<-filt.all2
##Features   Samples
##     4399        79

all2<-filt.all2
pData(all2)$mol.biol <- factor(pData(all2)$mol.biol)

## 2. Calculate and store p-values for FULL data

## 2A. eBayes using limma
## Design drops the column vector of ones
design <- model.matrix( ~mol.biol - 1, data = all2)
colnames(design) <- c("BCR_ABL", "NEG")
contr <- makeContrasts(BCR_ABL - NEG, levels=design)

## now fit eBayes model using lmFit
fit1 <- lmFit(all2, design)
fitc <- contrasts.fit(fit1, contr)
fitc <- eBayes(fitc)
## get top 1000 genes based on p-values
res.ebayes.all <- topTable(fitc, adjust.method="BH", number=10000
```

```
## 2B. T-tests
tt <- rowttests(all2, "mol.biol")

ttest.all <- tt[order(tt$p.value),]

## 2C. SAM

data <- list(x = exprs(all2), y = pData(all2)$mol.biol,
             geneid = featureNames(all2), genenames=featureNames(all2),
             logged2=TRUE)
res.sam <- samr(data = data, resp.type="Two class unpaired",nperms=10 )


## 3. Take top 1000 genes from each method
## 3a.eBayes
res.ebayes.all <- topTable(fitc, adjust.method="BH", number=1000)
head(res.ebayes.all)


## 3b. T-tests
ttest.all<-head(tt[order(tt$p.value),],n=1000)

## 3c. SAM
sam.top1000 <- sort(abs(res.sam$tt), decreasing=TRUE)[1:1000]

#############################################################################
##                        Calculate the expectation value of ALL data.
#############################################################################
## Run at least 100
nsim <- 100
n.subsample <- c(6, 10, 20, 50)
k.size <- c(50, 100, 500, 1000)

pog.E   <- array(NA, dim=c(4, 4, nsim),
             dimnames = list(n.subsample, k.size, 1:nsim))

pogr.E <- array(NA, dim=c(4, 4, nsim),
             dimnames = list(n.subsample, k.size, 1:nsim))

pogr.E.0.8 <- array(NA, dim=c(4, 4, nsim),
             dimnames = list(n.subsample, k.size, 1:nsim))

source("pogr.R")
for (i in 1:4) {    ## size of subset
  for (j in 1:nsim) {

  set.seed((i*100+j))

subsetBCRnABL <- intersect(which(all2$mol.biol%in%c("BCR/ABL")),
                    grep("^B", as.character(ALL$BT)))

subsetNEG <- intersect(which(all2$mol.biol%in%c("NEG")),
                    grep("^B", as.character(ALL$BT)))
```

```
## For subsets 1
sub.idxBCRnABL1 <- sample(subsetBCRnABL, n.subsample[i]/2, replace=FALSE)
sub.idxNEG1 <- sample(subsetNEG, n.subsample[i]/2, replace=FALSE)
sub.idx1<- union(sub.idxBCRnABL1,sub.idxNEG1 )


sub.mol.biol1 <- pData(all2)$mol.biol[sub.idx1]
allsub1 <- all2[,sub.idx1]

## 1. Randomly pick two lists (size 1000)
     ## head(exprs(allsub1))
     idx1 <- sample(1:nrow(allsub1), 1000, replace=FALSE)
     idx2 <- sample(1:nrow(all2), 1000, replace=FALSE)
     list1 <- rownames(exprs(allsub1))[idx1]
     list2 <- rownames(exprs(all2))[idx2]

## Check overlap with original gene list in terms of ranking for top 1000
## First, simple E[POG] score

pog.E[i,4, j] <- length(intersect(list1[1:1000], list2[1:1000]))/1000
pog.E[i,3, j] <- length(intersect(list1[1:500], list2[1:500]))/500
pog.E[i,2, j] <- length(intersect(list1[1:100], list2[1:100]))/100
pog.E[i,1, j] <- length(intersect(list1[1:50], list2[1:50]))/50

all.genes <- union(list1, list2)
cor.all.genes <- cor(t(exprs(all2)[all.genes, ]))

diag(cor.all.genes) <- 0


pogr.E[i, 4, j] <- pogr(cor.all.genes, list1[1:1000], list2[1:1000], 0.5)
pogr.E[i, 3, j] <- pogr(cor.all.genes, list1[1:500], list2[1:500], 0.5)
pogr.E[i, 2, j] <- pogr(cor.all.genes, list1[1:100], list2[1:100], 0.5)
pogr.E[i, 1, j] <- pogr(cor.all.genes, list1[1:50], list2[1:50], 0.5)

pogr.E.0.8[i, 4, j] <- pogr(cor.all.genes, list1[1:1000], list2[1:1000], 0.8)
pogr.E.0.8[i, 3, j] <- pogr(cor.all.genes, list1[1:500], list2[1:500], 0.8)
pogr.E.0.8[i, 2, j] <- pogr(cor.all.genes, list1[1:100], list2[1:100], 0.8)
pogr.E.0.8[i, 1, j] <- pogr(cor.all.genes, list1[1:50], list2[1:50], 0.8)
}}

## cutoff=0.5
exp.POG   <- ftable(apply(pog.E, c(1,2), mean))   ## saved as a matrix
exp.POGR <- ftable(apply(pogr.E, c(1,2), mean))

## cutoff=0.8
exp.POGR.0.8 <- ftable(apply(pogr.E.0.8, c(1,2), mean))

##write.csv( )
write.csv(exp.POG,file="E(pog).csv",row.names =FALSE)
write.csv(exp.POGR,file="E(pogr).csv",row.names =FALSE)

## cutoff=0.8
write.csv(exp.POGR.0.8,file="E(pogr)08.csv",row.names =FALSE)
```

```
##read.csv
EPOG <- read.csv("E(pog).csv")
EPOGR <- read.csv("E(pogr).csv")

## cutoff=0.8
EPOGR.0.8 <- read.csv("E(pogr)08.csv")

#############################################################################
##                          Calculate the scores of DEGs
#############################################################################

nsim <- 25
n.subsample <- c(6, 10, 20, 50)
k.size <- c(50, 100, 500, 1000)

## Create array to store results (what are they?)
pog.array <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ebayes", "ttest", "SAM"), n.subsample, k.size, 1:nsim))
pogr.array <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ebayes", "ttest", "SAM"), n.subsample, k.size, 1:nsim))
npog.array <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ebayes", "ttest", "SAM"), n.subsample, k.size, 1:nsim))
npogr.array <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ebayes", "ttest", "SAM"), n.subsample, k.size, 1:nsim))
pogr.array.0.8 <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ebayes", "ttest", "SAM"), n.subsample, k.size, 1:nsim))
npogr.array.0.8 <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ebayes", "ttest", "SAM"), n.subsample, k.size, 1:nsim))

## To calculate normalized scores
EPOG <- read.csv("E(pog).csv")
EPOGR <- read.csv("E(pogr).csv")
## cutoff=0.8
EPOGR.0.8 <- read.csv("E(pogr)08.csv")

## do the same for POGR array and any other results

## need loops over size of subsample and # simulations
## other factors will be 'inside' of looops

## source code for POGR score
source("pogr.R")
for (i in 1:4) {
  for (j in 1:nsim) {

  ## set seed for each simulation so results are exactly reproducible
  set.seed((i*100+j))

  ## 1. size of subsample
subset <- intersect(which(ALL$mol.biol%in%c("BCR/ABL","NEG")),
                    grep("^B", as.character(ALL$BT)))
## selects 79 samples
subsetBCRnABL <- intersect(which(all2$mol.biol%in%c("BCR/ABL")),
                    grep("^B", as.character(ALL$BT)))
```

```
subsetNEG <- intersect(which(all2$mol.biol%in%c("NEG")),
                       grep("^B", as.character(ALL$BT)))
sub.idxBCRnABL <- sample(subsetBCRnABL, n.subsample[i]/2, replace=FALSE)
sub.idxNEG <- sample(subsetNEG, n.subsample[i]/2, replace=FALSE)
sub.idx<- union(sub.idxBCRnABL,sub.idxNEG )

sub.mol.biol <- pData(all2)$mol.biol[sub.idx]
allsub <- all2[,sub.idx]

###############################################################################
##                          2A. E-BAYES (LIMMA) METHOD
###############################################################################
lnames(design) <- c("BCR_ABL", "NEG")
contr <- makeContrasts(BCR_ABL - NEG, levels=design)
fit1sub <- lmFit(all2[,sub.idx], design)
fitcsub <- contrasts.fit(fit1sub, contr)
fitcsub <- eBayes(fitcsub)
## Full gene list
res.ebayes.all <- topTable(fitc, adjust.method="BH", number=1000)
ressub <- topTable(fitcsub, adjust.method="BH", number=1000)

pog.array["ebayes",i, 4, j] <- length(intersect(ressub$ID[1:1000],
res.ebayes.all$ID[1:1000]))/1000
pog.array["ebayes",i, 3, j] <- length(intersect(ressub$ID[1:500],
res.ebayes.all$ID[1:500]))/500
pog.array["ebayes",i, 2, j] <- length(intersect(ressub$ID[1:100],
res.ebayes.all$ID[1:100]))/100
pog.array["ebayes",i, 1, j] <- length(intersect(ressub$ID[1:50], res.ebayes.all$ID[1:50]))/50

## Normalized POG
npog.array["ebayes",i, 4, j] <- (pog.array["ebayes",i, 4, j] - EPOG[i,4])/(1-EPOG[i,4])
npog.array["ebayes",i, 3, j] <- (pog.array["ebayes",i, 3, j] - EPOG[i,3])/(1-EPOG[i,3])
npog.array["ebayes",i, 2, j] <- (pog.array["ebayes",i, 2, j] - EPOG[i,2])/(1-EPOG[i,2])
npog.array["ebayes",i, 1, j] <- (pog.array["ebayes",i, 1, j] - EPOG[i,1])/(1-EPOG[i,1])

## Now, look at POGR score
all.genes <- union(ressub$ID, res.ebayes.all$ID)
cor.all.genes <- cor(t(exprs(all2)[all.genes, ]))

diag(cor.all.genes) <- 0

## cutoff=0.5
pogr.array["ebayes",i, 4, j] <- pogr(cor.all.genes, res.ebayes.all$ID[1:1000],
ressub$ID[1:1000], 0.5)
pogr.array["ebayes",i, 3, j] <- pogr(cor.all.genes, res.ebayes.all$ID[1:500], ressub$ID[1:500],
0.5)
pogr.array["ebayes",i, 2, j] <- pogr(cor.all.genes, res.ebayes.all$ID[1:100], ressub$ID[1:100],
0.5)
pogr.array["ebayes",i, 1, j] <- pogr(cor.all.genes, res.ebayes.all$ID[1:50], ressub$ID[1:50],
0.5)

npogr.array["ebayes",i, 4, j] <- (pogr.array["ebayes",i, 4, j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array["ebayes",i, 3, j] <- (pogr.array["ebayes",i, 3, j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array["ebayes",i, 2, j] <- (pogr.array["ebayes",i, 2, j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array["ebayes",i, 1, j] <- (pogr.array["ebayes",i, 1, j] - EPOGR[i,1])/(1-EPOGR[i,1])
```

```
## cutoff=0.8
pogr.array.0.8["ebayes",i, 4, j] <- pogr(cor.all.genes, res.ebayes.all$ID[1:1000],
ressub$ID[1:1000], 0.8)
pogr.array.0.8["ebayes",i, 3, j] <- pogr(cor.all.genes, res.ebayes.all$ID[1:500],
ressub$ID[1:500], 0.8)
pogr.array.0.8["ebayes",i, 2, j] <- pogr(cor.all.genes, res.ebayes.all$ID[1:100],
ressub$ID[1:100], 0.8)
pogr.array.0.8["ebayes",i, 1, j] <- pogr(cor.all.genes, res.ebayes.all$ID[1:50], ressub$ID[1:50],
0.8)

npogr.array.0.8["ebayes",i, 4, j] <- (pogr.array.0.8["ebayes",i, 4, j] -
EPOGR.0.8[i,4])/(1-EPOGR.0.8[i,4])
npogr.array.0.8["ebayes",i, 3, j] <- (pogr.array.0.8["ebayes",i, 3, j] -
EPOGR.0.8[i,3])/(1-EPOGR.0.8[i,3])
npogr.array.0.8["ebayes",i, 2, j] <- (pogr.array.0.8["ebayes",i, 2, j] -
EPOGR.0.8[i,2])/(1-EPOGR.0.8[i,2])
npogr.array.0.8["ebayes",i, 1, j] <- (pogr.array.0.8["ebayes",i, 1, j] -
EPOGR.0.8[i,1])/(1-EPOGR.0.8[i,1])

##############################################################################
##                                    2B.T-TEST
##############################################################################

ttsub <- rowttests(all2[,sub.idx], all2[,sub.idx]$mol.biol)

ttest.sub<-head(ttsub[order(ttsub$p.value),],n=1000)

ttest.all<-head(tt[order(tt$p.value),],n=1000)

## First, simple POG score
rownames(ttest.all)
rownames(ttest.sub)
pog.array["ttest",i, 4, j] <- length(intersect(rownames(ttest.all)[1:1000],
rownames(ttest.sub)[1:1000]))/1000
pog.array["ttest",i, 3, j] <- length(intersect(rownames(ttest.all)[1:500],
rownames(ttest.sub)[1:500]))/500
pog.array["ttest",i, 2, j] <- length(intersect(rownames(ttest.all)[1:100],
rownames(ttest.sub)[1:100]))/100
pog.array["ttest",i, 1, j] <- length(intersect(rownames(ttest.all)[1:50],
rownames(ttest.sub)[1:50]))/50

## Normalized POG
npog.array["ttest",i, 4, j] <- (pog.array["ttest",i, 4, j] - EPOG[i,4])/(1-EPOG[i,4])
npog.array["ttest",i, 3, j] <- (pog.array["ttest",i, 3, j] - EPOG[i,3])/(1-EPOG[i,3])
npog.array["ttest",i, 2, j] <- (pog.array["ttest",i, 2, j] - EPOG[i,2])/(1-EPOG[i,2])
npog.array["ttest",i, 1, j] <- (pog.array["ttest",i, 1, j] - EPOG[i,1])/(1-EPOG[i,1])

## Now, look at POGR score
all.genes <- union(rownames(ttest.all), rownames(ttest.sub))
cor.all.genes <- cor(t(exprs(all2)[all.genes, ]))
## need to find which are significant (just pick a cut-off (say 0.5) and use that)
## need to look at all OFF DIAGONAL values of cor.all.genes (set diagonal values to zero)
diag(cor.all.genes) <- 0
```

```
## maybe have another dimension for correlation cutoff?? (0.5, 0.8)

## cutoff=0.5
pogr.array["ttest",i, 4, j] <- pogr(cor.all.genes, rownames(ttest.all)[1:1000],
rownames(ttest.sub)[1:1000], 0.5)
pogr.array["ttest",i, 3, j] <- pogr(cor.all.genes, rownames(ttest.all)[1:500],
rownames(ttest.sub)[1:500], 0.5)
pogr.array["ttest",i, 2, j] <- pogr(cor.all.genes, rownames(ttest.all)[1:100],
rownames(ttest.sub)[1:100], 0.5)
pogr.array["ttest",i, 1, j] <- pogr(cor.all.genes, rownames(ttest.all)[1:50],
rownames(ttest.sub)[1:50], 0.5)

npogr.array["ttest",i, 4, j] <- (pogr.array["ttest",i, 4, j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array["ttest",i, 3, j] <- (pogr.array["ttest",i, 3, j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array["ttest",i, 2, j] <- (pogr.array["ttest",i, 2, j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array["ttest",i, 1, j] <- (pogr.array["ttest",i, 1, j] - EPOGR[i,1])/(1-EPOGR[i,1])

## cutoff=0.8
pogr.array.0.8["ttest",i, 4, j] <- pogr(cor.all.genes, rownames(ttest.all)[1:1000],
rownames(ttest.sub)[1:1000], 0.8)
pogr.array.0.8["ttest",i, 3, j] <- pogr(cor.all.genes, rownames(ttest.all)[1:500],
rownames(ttest.sub)[1:500], 0.8)
pogr.array.0.8["ttest",i, 2, j] <- pogr(cor.all.genes, rownames(ttest.all)[1:100],
rownames(ttest.sub)[1:100], 0.8)
pogr.array.0.8["ttest",i, 1, j] <- pogr(cor.all.genes, rownames(ttest.all)[1:50],
rownames(ttest.sub)[1:50], 0.8)

npogr.array.0.8["ttest",i, 4, j] <- (pogr.array.0.8["ttest",i, 4, j] -
EPOGR.0.8[i,4])/(1-EPOGR.0.8[i,4])
npogr.array.0.8["ttest",i, 3, j] <- (pogr.array.0.8["ttest",i, 3, j] -
EPOGR.0.8[i,3])/(1-EPOGR.0.8[i,3])
npogr.array.0.8["ttest",i, 2, j] <- (pogr.array.0.8["ttest",i, 2, j] -
EPOGR.0.8[i,2])/(1-EPOGR.0.8[i,2])
npogr.array.0.8["ttest",i, 1, j] <- (pogr.array.0.8["ttest",i, 1, j] -
EPOGR.0.8[i,1])/(1-EPOGR.0.8[i,1])


##############################################################################
##                                    2C. SAM
##############################################################################
data <- list(x = exprs(all2[,sub.idx]), y = pData(all2[,sub.idx])$mol.biol,
             geneid = featureNames(all2[,sub.idx]),
genenames=featureNames(all2[,sub.idx]),
             logged2=TRUE)

samsub <- samr(data = data, resp.type="Two class unpaired", nperms=10)
samsub.top1000 <- sort(abs(samsub$tt), decreasing=TRUE)[1:1000]

## First, simple POG score
pog.array["SAM",i, 4, j] <- length(intersect(names(sam.top1000)[1:1000],
names(samsub.top1000)[1:1000]))/1000
pog.array["SAM",i, 3, j] <- length(intersect(names(sam.top1000)[1:500],
names(samsub.top1000)[1:500]))/500
pog.array["SAM",i, 2, j] <- length(intersect(names(sam.top1000)[1:100],
names(samsub.top1000)[1:100]))/100
```

```
pog.array["SAM",i, 1, j] <- length(intersect(names(sam.top1000)[1:50],
names(samsub.top1000)[1:50]))/50

## Normalized POG
npog.array["SAM",i, 4, j] <- (pog.array["SAM",i, 4, j] - EPOG[i,4])/(1-EPOG[i,4])
npog.array["SAM",i, 3, j] <- (pog.array["SAM",i, 3, j] - EPOG[i,3])/(1-EPOG[i,3])
npog.array["SAM",i, 2, j] <- (pog.array["SAM",i, 2, j] - EPOG[i,2])/(1-EPOG[i,2])
npog.array["SAM",i, 1, j] <- (pog.array["SAM",i, 1, j] - EPOG[i,1])/(1-EPOG[i,1])


## Now, look at POGR score
all.genes <- union(names(samsub.top1000), names(sam.top1000))
cor.all.genes <- cor(t(exprs(all2)[all.genes, ]))

diag(cor.all.genes) <- 0

## cut point=0.5
pogr.array["SAM",i, 4, j] <- pogr(cor.all.genes, names(sam.top1000)[1:1000],
names(samsub.top1000)[1:1000], 0.5)
pogr.array["SAM",i, 3, j] <- pogr(cor.all.genes, names(sam.top1000)[1:500],
names(samsub.top1000)[1:500], 0.5)
pogr.array["SAM",i, 2, j] <- pogr(cor.all.genes, names(sam.top1000)[1:100],
names(samsub.top1000)[1:100], 0.5)
pogr.array["SAM",i, 1, j] <- pogr(cor.all.genes, names(sam.top1000)[1:50],
names(samsub.top1000)[1:50], 0.5)

npogr.array["SAM",i, 4, j] <- (pogr.array["SAM",i, 4, j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array["SAM",i, 3, j] <- (pogr.array["SAM",i, 3, j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array["SAM",i, 2, j] <- (pogr.array["SAM",i, 2, j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array["SAM",i, 1, j] <- (pogr.array["SAM",i, 1, j] - EPOGR[i,1])/(1-EPOGR[i,1])

## cutoff=0.8
pogr.array.0.8["SAM",i, 4, j] <- pogr(cor.all.genes, names(sam.top1000)[1:1000],
names(samsub.top1000)[1:1000], 0.8)
pogr.array.0.8["SAM",i, 3, j] <- pogr(cor.all.genes, names(sam.top1000)[1:500],
names(samsub.top1000)[1:500], 0.8)
pogr.array.0.8["SAM",i, 2, j] <- pogr(cor.all.genes, names(sam.top1000)[1:100],
names(samsub.top1000)[1:100], 0.8)
pogr.array.0.8["SAM",i, 1, j] <- pogr(cor.all.genes, names(sam.top1000)[1:50],
names(samsub.top1000)[1:50], 0.8)

npogr.array.0.8["SAM",i, 4, j] <- (pogr.array.0.8["SAM",i, 4, j] -
EPOGR.0.8[i,4])/(1-EPOGR.0.8[i,4])
npogr.array.0.8["SAM",i, 3, j] <- (pogr.array.0.8["SAM",i, 3, j] -
EPOGR.0.8[i,3])/(1-EPOGR.0.8[i,3])
npogr.array.0.8["SAM",i, 2, j] <- (pogr.array.0.8["SAM",i, 2, j] -
EPOGR.0.8[i,2])/(1-EPOGR.0.8[i,2])
npogr.array.0.8["SAM",i, 1, j] <- (pogr.array.0.8["SAM",i, 1, j] -
EPOGR.0.8[i,1])/(1-EPOGR.0.8[i,1])
        }}

#########################################################################
##                        END OF SIMULATION LOOPS
#########################################################################
```

Codes for Conlon Cancer, Lung Expression and Leukimia are similar to the ALL data.

```
#########################################################################
##                    Plot nPOG and nPOGR scores for four data sets
#########################################################################

library(lattice)
nPOG.ALL <- read.csv("npog ALL.csv")
nPOG.colonCA <- read.csv("npog colonCA.csv")
nPOG.LungExpression <- read.csv("npog michigan.csv")
nPOG.Leukemia <- read.csv("npog Golub_Merge.csv")


## Plot all using lattice - xyplot
npog.all.mean <- rowMeans(nPOG.ALL)
npog.colon.mean <- rowMeans(nPOG.colonCA)
npog.lung.mean <- rowMeans(nPOG.LungExpression)
npog.luek.mean <- rowMeans(nPOG.Leukemia)

plot.df <- data.frame(npog = c(npog.all.mean, npog.colon.mean, npog.lung.mean,
                               npog.luek.mean),
                      data = rep(c("ALL", "ColonCA", "Lung", "Leukemia"), each=12),
                      subset = rep(c(6,10,20,50), 12),
                      method = rep(rep(c("eBayes", "t-test", "SAM"), each=4), 4))
## plot.df

xyplot(npog~subset | data, groups = method, type="l", data=plot.df,
      auto.key=list(columns=3,title="nPOG score for detecting DEG lists from the same data
set"))

#########################################################
## nPOGR
nPOGR.ALL <- read.csv("npogr.csv")
nPOGR.colonCA <- read.csv("npogr colonCA.csv")
nPOGR.LungExpression <- read.csv("npogr michigan.csv")
nPOGR.Leukemia <- read.csv("npogr Golub_Merge.csv")

npogr.all.mean <- rowMeans(nPOGR.ALL)
npogr.colon.mean <- rowMeans(nPOGR.colonCA)
npogr.lung.mean <- rowMeans(nPOGR.LungExpression)
npogr.luek.mean <- rowMeans(nPOGR.Leukemia)

plot.df <- data.frame(npogr = c(npogr.all.mean, npogr.colon.mean, npogr.lung.mean,
                                npogr.luek.mean),
                      data = rep(c("ALL", "ColonCA", "Lung", "Leukemia"), each=12),
                      subset = rep(c(6,10,20,50), 12),
                      method = rep(rep(c("eBayes", "t-test", "SAM"), each=4), 4))
##plot.df

xyplot(npogr~subset | data, groups = method, type="l", data=plot.df,
      auto.key=list(columns=3,title="nPOGR score for detecting DEG lists from the same
data set"))
```

R code for section 3.2

Prostate Cancer

```
## load packages
library(Biobase)
library(GEOquery)
library(limma)
library(genefilter)
library(multtest)
library(samr)
library(affy)
library(impute)
library(hgu95av2.db)
library(hgu133a.db)
library(marray)


################################################################################
##                        Prostate Cancer cDNA data from Lapointe
################################################################################
setwd("/mnt/fs2/home/d0yang03/prostatecDNA")

files <- dir(pattern="*\\.xls$")

## read.SMD(files)
prostate.cDNA <- read.maimages(files, source="smd")


prostate.eset <- normalizeWithinArrays(prostate.cDNA, method = "median")

## dim(prostate.eset)
## 43008       86

geneNames <- prostate.eset$genes[,3]
rownames(prostate.eset$M)<-geneNames
prostate.eset2<-prostate.eset$M[unique(rownames(prostate.eset$M)),]


################################################################################
##                          prostate Cancer Affy data from Lapointe
################################################################################


setwd("/mnt/fs2/home/d0yang03/prostateAffy")
prostate.Affy <- ReadAffy()
head(pData(prostate.Affy))
dim(exprs(prostate.Affy))
## 409600       102
prostate.Affy.rma <- rma(prostate.Affy)

filt.prostate.Affy.rma   <- nsFilter(prostate.Affy.rma , require.entrez=TRUE,
                    require.GOBP=FALSE, remove.dupEntrez=TRUE,
```

```
                    feature.exclude="Affy", var.cutoff=0.5)
dim(filt.prostate.Affy.rma$eset)
## Features   Samples
##      4399         102

prostate.Affy.rma2 <- filt.prostate.Affy.rma$eset
prostate.Affy.ID <-
mget(featureNames(prostate.Affy.rma2),hgu95av2SYMBOL,ifnotfound=NA)

featureNames(prostate.Affy.rma2) <- prostate.Affy.ID

prostate.cDNA2 <- prostate.eset$M[!apply(prostate.eset$M,1,function(y) any(is.na(y))),]
## overlap genes
overlap <- intersect(prostate.Affy.ID,rownames(prostate.cDNA2))

prostate.cDNA3 <- prostate.cDNA2[which(rownames(prostate.cDNA2)%in%overlap),]
prostate.cDNA4<-prostate.cDNA3[unique(rownames(prostate.cDNA3)),]

prostate.Affy.rma3 <-
prostate.Affy.rma2[which(featureNames(prostate.Affy.rma2)%in%overlap)]


## 2B. eBayes using limma

pd1 <- data.frame(grp=c(rep(1,86)))
pd1$grp[c(2,4,8,13,17,19,22,24,26,29,31,32,34,37,38,
          43,44,47,50,53,54,57,59,63,65,68,74,77,78,80,83,84)]<-2
pd1$grp <- factor(pd1$grp, labels=c("Tumor","Normal"))
rownames(pd1) <- colnames(prostate.cDNA3)
design1 <- model.matrix( ~grp - 1, data = pd1)
colnames(design1) <- c("Tumor", "Normal")
contr1 <- makeContrasts(Tumor-Normal, levels=design1)


## prostate.Affy
pd2 <- data.frame(grp=c(rep(1,50),rep(2,52)))
pd2$grp <- factor(pd2$grp, labels=c("Normal","Tumor"))
rownames(pd2) <- rownames(pData(prostate.Affy.rma3))
pData(prostate.Affy.rma3) <- pd2
phenoData(prostate.Affy.rma3)
design2 <- model.matrix( ~grp - 1, data = prostate.Affy.rma3)
colnames(design2) <- c("Tumor", "Normal")
contr2 <- makeContrasts(Tumor-Normal, levels=design2)


## prostate.cDNA
fit1 <- lmFit(prostate.cDNA4, design1)
fitc1 <- contrasts.fit(fit1, contr1)
fitc1 <- eBayes(fitc1)
res.prostate.cDNA <- topTable(fitc1, adjust.method="BH",number=1000)

## prostate.Affy
fit2 <- lmFit(prostate.Affy.rma3, design2)
fitc2 <- contrasts.fit(fit2, contr2)
fitc2 <- eBayes(fitc2)
```

```
res.prostate.Affy <- topTable(fitc2, adjust.method="BH",number=1000)



## 2A. T-tests

tt1 <- rowttests(prostate.cDNA4, fac=pd1$grp)
## res.ttest <- rowttests(prostate.eset$M,fac=pd1$grp)

ttest.prostate.cDNA <- tt1[order(tt1$p.value),]

## prostate.Affy
tt2 <- rowttests(prostate.Affy.rma3, "grp")
ttest.prostate.Affy <- tt2[order(tt2$p.value),]



## 2C. SAM
## prostate.cDNA
data1 <- list(x = prostate.cDNA4, y = pd1$grp,
                geneid = rownames(prostate.cDNA4), rownames(prostate.cDNA4),
                logged2=TRUE)
res.sam1 <- samr(data = data1, resp.type="Two class unpaired",nperms=50 )

## prostate.Affy
data2 <- list(x = exprs(prostate.Affy.rma3), y = pData(prostate.Affy.rma3)$grp,
                geneid = featureNames(prostate.Affy.rma3),
genenames=featureNames(prostate.Affy.rma3),
                logged2=TRUE)
res.sam2 <- samr(data = data2, resp.type="Two class unpaired",nperms=50 )


##############################################################################
##                  Maybe just take top 1000 genes to comprise gene list
##############################################################################
## 3. Take top 1000 genes from each method

## 3a. T-tests
ttest.prostate.cDNA<-head(tt1[order(tt1$p.value),],n=1422)

## 729
ttest.prostate.Affy<-head(tt2[order(tt2$p.value),],n=729)

length(intersect(rownames(ttest.prostate.cDNA), rownames(ttest.prostate.Affy)))
## 551

## corrletion
##
cor(t(exprs(DMD.haslett.rma)[rownames(ttest.DMD.haslett),])[1,],t(exprs(DMD.pescatori.rm
a)[rownames(ttest.DMD.pescatori),]))

## 3b.eBayes

ebayes.prostate.cDNA <- topTable(fitc1, adjust.method="BH", number=632)
```

```
ebayes.prostate.Affy <- topTable(fitc2, adjust.method="BH", number=2000)

length(intersect(ebayes.prostate.cDNA$ID,ebayes.prostate.Affy$ID))

.
## 3c. SAM
sam.prostate.cDNA <- sort(abs(res.sam1$tt), decreasing=TRUE)[1:1031]

sam.prostate.Affy <- sort(abs(res.sam2$tt), decreasing=TRUE)[1:1181]


##############################################################################
##                          Calculate the expectation value
##############################################################################
nsim <- 100
k.size <- c(50, 100, 500, 1000)
n.subsample <- c(6,10,20,50)

pog.E <- array(NA, dim=c(4, 4, nsim),
               dimnames = list(n.subsample, k.size, 1:nsim))

pogr.E <- array(NA, dim=c(4, 4, nsim),
               dimnames = list(n.subsample, k.size, 1:nsim))

pogr.E.0.7 <- array(NA, dim=c(4, 4, nsim),
               dimnames = list(n.subsample, k.size, 1:nsim))

source("POGR2.R")
cor.prostate.cDNA <- cor(t(prostate.cDNA4))
cor.prostate.Affy <- cor(t(exprs(prostate.Affy.rma3)))

source("POGR2.R")
for (i in 1:4) {
  for (j in 1:nsim) {
set.seed((i*100+j))
subset.prostate.cDNA.Nor <- which(pd1$grp%in%c("Normal"))
subset.prostate.cDNA.Tur <- which(pd1$grp%in%c("Tumor"))

sub.idx.cDNA.Nor <- sample(subset.prostate.cDNA.Nor, n.subsample[i]/4, replace=FALSE)
sub.idx.cDNA.Tur <- sample(subset.prostate.cDNA.Tur, n.subsample[i]*3/4,
replace=FALSE)
sub.idx1<- union(sub.idx.cDNA.Nor, sub.idx.cDNA.Tur)

sub.prostate.cDNA <- pd1$grp[sub.idx1]
prostate.cDNA.sub <- prostate.cDNA4[,sub.idx1]

## prostate.Affy subsets
subset.prostate.Affy.Nor <- which(pd2$grp%in%c("Normal"))
subset.prostate.Affy.Tur <- which(pd2$grp%in%c("Tumor"))

sub.idx.prostate.Affy.Nor <- sample(subset.prostate.Affy.Nor, n.subsample[i]/2,
replace=FALSE)
sub.idx.prostate.Affy.Tur <- sample(subset.prostate.Affy.Tur, n.subsample[i]/2,
replace=FALSE)
sub.idx2<- union(sub.idx.prostate.Affy.Nor, sub.idx.prostate.Affy.Tur)
```

```
sub.prostate.Affy <- pData(prostate.Affy.rma3)[sub.idx2,]
prostate.Affy.sub <- prostate.Affy.rma3[,sub.idx2]


## 1. Randomly pick two lists (size 1000)
    ## head(exprs(allsub1))
    idx1 <- sample(1:nrow(prostate.cDNA.sub), 1000, replace=FALSE)
    idx2 <- sample(1:nrow(prostate.Affy.sub ), 1000, replace=FALSE)
    list1 <- rownames(prostate.cDNA.sub)[idx1]
    list2 <- rownames(exprs(prostate.Affy.sub))[idx2]

## Check overlap with original gene list in terms of ranking for top 1000
## First, simple E[POG] score

pog.E[i, 4, j] <- length(intersect(list1[1:1000], list2[1:1000]))/1000
pog.E[i, 3, j] <- length(intersect(list1[1:500], list2[1:500]))/500
pog.E[i, 2, j] <- length(intersect(list1[1:100], list2[1:100]))/100
pog.E[i, 1, j] <- length(intersect(list1[1:50], list2[1:50]))/50

## Now, look at POGR.E score
## Need to calculate all pairwise correlations between these two sets of genes

pogr.E[i, 4, j] <- pogr(cor.prostate.cDNA ,cor.prostate.Affy, list1[1:1000], list2[1:1000], 0.5)
pogr.E[i, 3, j] <- pogr(cor.prostate.cDNA ,cor.prostate.Affy, list1[1:500], list2[1:500], 0.5)
pogr.E[i, 2, j] <- pogr(cor.prostate.cDNA ,cor.prostate.Affy, list1[1:100], list2[1:100], 0.5)
pogr.E[i, 1, j] <- pogr(cor.prostate.cDNA ,cor.prostate.Affy, list1[1:50], list2[1:50], 0.5)


pogr.E.0.7[i, 4, j] <- pogr(cor.prostate.cDNA ,cor.prostate.Affy, list1[1:1000], list2[1:1000],
0.7)
pogr.E.0.7[i, 3, j] <- pogr(cor.prostate.cDNA ,cor.prostate.Affy, list1[1:500], list2[1:500],
0.7)
pogr.E.0.7[i, 2, j] <- pogr(cor.prostate.cDNA ,cor.prostate.Affy, list1[1:100], list2[1:100],
0.65)
pogr.E.0.7[i, 1, j] <- pogr(cor.prostate.cDNA ,cor.prostate.Affy, list1[1:50], list2[1:50], 0.7)

}}

##############################################################################
##                                  Calculate the scores
##############################################################################

nsim <- 50
k.size <- c(50, 100, 500, 1000)
n.subsample <- c(6,10,20,50)

## Create array to store results (what are they?)
pog.array <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))
pogr.array <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))
pogr.array2 <- array(NA, dim=c(3, 4, 4, nsim),
                   dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))
npog.array <- array(NA, dim=c(3, 4, 4, nsim),
```

```
                dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))
npogr.array <- array(NA, dim=c(3,4, 4, nsim),
                dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))
npogr.array2 <- array(NA, dim=c(3,4, 4, nsim),
                dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))

pogr.array.0.7 <- array(NA, dim=c(3, 4, 4, nsim),
                dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))
npogr.array.0.7 <- array(NA, dim=c(3, 4, 4, nsim),
                dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))
pogr.array2.0.7 <- array(NA, dim=c(3, 4, 4, nsim),
                dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))
npogr.array2.0.7 <- array(NA, dim=c(3, 4, 4, nsim),
                dimnames = list(c("ttest","ebayes", "SAM"),n.subsample, k.size, 1:nsim))


## To calculate normalized scores
EPOG <- read.csv("E(pog)prostate.sub.csv")
EPOGR <- read.csv("E(pogr)prostate.sub.csv")
## cutoff=0.7
EPOGR.0.7 <- read.csv("E(pogr)prostate.sub.07.csv")

## cor matrix for two datasets
cor.prostate.cDNA <- cor(t(prostate.cDNA4))
cor.prostate.Affy <- cor(t(exprs(prostate.Affy.rma3)))

source("POGR2.R")

for (i in 1:4) {
    for (j in 1:nsim) {

set.seed((i*100+j))
subset.prostate.cDNA.Nor <- which(pd1$grp%in%c("Normal"))
subset.prostate.cDNA.Tur <- which(pd1$grp%in%c("Tumor"))

sub.idx.cDNA.Nor <- sample(subset.prostate.cDNA.Nor, n.subsample[i]/2, replace=FALSE)
sub.idx.cDNA.Tur <- sample(subset.prostate.cDNA.Tur, n.subsample[i]/2, replace=FALSE)
sub.idx1<- union(sub.idx.cDNA.Nor, sub.idx.cDNA.Tur)

sub.prostate.cDNA <- pd1$grp[sub.idx1]
prostate.cDNA.sub <- prostate.cDNA4[,sub.idx1]

## prostate.Affy subsets
subset.prostate.Affy.Nor <- which(pd2$grp%in%c("Normal"))
subset.prostate.Affy.Tur <- which(pd2$grp%in%c("Tumor"))

sub.idx.prostate.Affy.Nor <- sample(subset.prostate.Affy.Nor, n.subsample[i]/2,
replace=FALSE)
sub.idx.prostate.Affy.Tur <- sample(subset.prostate.Affy.Tur, n.subsample[i]/2,
replace=FALSE)
sub.idx2<- union(sub.idx.prostate.Affy.Nor, sub.idx.prostate.Affy.Tur)

sub.prostate.Affy <- pData(prostate.Affy.rma3)[sub.idx2,]
prostate.Affy.sub <- prostate.Affy.rma3[,sub.idx2]
```

```
##############################################################################
##                                    2A.T-TEST
##############################################################################

## prostate.cDNA
design1 <- model.matrix( ~sub.prostate.cDNA - 1)
colnames(design1) <- c("Tumor", "Normal")
contr1 <- makeContrasts(Tumor-Normal, levels=design1)

## prostate.Affy
design2 <- model.matrix( ~sub.prostate.Affy - 1)
colnames(design2) <- c("Tumor", "Normal")
contr1 <- makeContrasts(Tumor-Normal, levels=design2)

## prostate.cDNAlett
tt1 <- rowttests(prostate.cDNA.sub, sub.prostate.cDNA)
ttest.prostate.cDNA <- tt1[order(tt1$p.value),]

## prostate.Affy
tt2 <- rowttests(prostate.Affy.sub, sub.prostate.Affy)
ttest.prostate.Affy <- tt2[order(tt2$p.value),]



## First, simple POG score
pog.array["ttest",i,4,j] <- length(intersect(rownames(ttest.prostate.cDNA)[1:1000],
                            rownames(ttest.prostate.Affy)[1:1000]))/1000
pog.array["ttest",i,3,j] <- length(intersect(rownames(ttest.prostate.cDNA)[1:500],
                            rownames(ttest.prostate.Affy)[1:500]))/500
pog.array["ttest",i,2,j] <- length(intersect(rownames(ttest.prostate.cDNA)[1:100],
                            rownames(ttest.prostate.Affy)[1:100]))/100
pog.array["ttest",i,1,j] <- length(intersect(rownames(ttest.prostate.cDNA)[1:50],
                            rownames(ttest.prostate.Affy)[1:50]))/50


## Normalized POG
npog.array["ttest",i,4,j] <- (pog.array["ttest",i,4,j] - EPOG[i,4])/(1-EPOG[i,4])
npog.array["ttest",i,3,j] <- (pog.array["ttest",i,3,j] - EPOG[i,3])/(1-EPOG[i,3])
npog.array["ttest",i,2,j] <- (pog.array["ttest",i,2,j] - EPOG[i,2])/(1-EPOG[i,2])
npog.array["ttest",i,1,j] <- (pog.array["ttest",i,1,j] - EPOG[i,1])/(1-EPOG[i,1])


## Now, look at POGR score
## cutoff=0.5
pogr.array["ttest",i,4,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                        rownames(ttest.prostate.cDNA)[1:1000],
rownames(ttest.prostate.Affy)[1:1000], 0.5)
pogr.array["ttest",i,3,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                        rownames(ttest.prostate.cDNA)[1:500],
rownames(ttest.prostate.Affy)[1:500], 0.5)
pogr.array["ttest",i,2,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                        rownames(ttest.prostate.cDNA)[1:100],
rownames(ttest.prostate.Affy)[1:100], 0.5)
pogr.array["ttest",i,1,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                        rownames(ttest.prostate.cDNA)[1:50],
```

```
rownames(ttest.prostate.Affy)[1:50], 0.5)


## Need to also go other direction
pogr.array2["ttest",i,4,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                        rownames(ttest.prostate.Affy)[1:1000],
rownames(ttest.prostate.cDNA)[1:1000], 0.5)
pogr.array2["ttest",i,3,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                        rownames(ttest.prostate.Affy)[1:500],
rownames(ttest.prostate.cDNA)[1:500], 0.5)
pogr.array2["ttest",i,2,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                        rownames(ttest.prostate.Affy)[1:100],
rownames(ttest.prostate.cDNA)[1:100], 0.5)
pogr.array2["ttest",i,1,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                        rownames(ttest.prostate.Affy)[1:50],
rownames(ttest.prostate.cDNA)[1:50], 0.5)



npogr.array["ttest",i,4,j] <- (pogr.array["ttest",i,4,j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array["ttest",i,3,j] <- (pogr.array["ttest",i,3,j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array["ttest",i,2,j] <- (pogr.array["ttest",i,2,j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array["ttest",i,1,j] <- (pogr.array["ttest",i,1,j] - EPOGR[i,1])/(1-EPOGR[i,1])


npogr.array2["ttest",i,4,j] <- (pogr.array2["ttest",i,4,j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array2["ttest",i,3,j] <- (pogr.array2["ttest",i,3,j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array2["ttest",i,2,j] <- (pogr.array2["ttest",i,2,j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array2["ttest",i,1,j] <- (pogr.array2["ttest",i,1,j] - EPOGR[i,1])/(1-EPOGR[i,1])

## cutoff=0.7
pogr.array.0.7["ttest",i,4,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                        rownames(ttest.prostate.cDNA)[1:1000],
rownames(ttest.prostate.Affy)[1:1000], 0.65)
pogr.array.0.7["ttest",i,3,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                        rownames(ttest.prostate.cDNA)[1:500],
rownames(ttest.prostate.Affy)[1:500], 0.65)
pogr.array.0.7["ttest",i,2,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                        rownames(ttest.prostate.cDNA)[1:100],
rownames(ttest.prostate.Affy)[1:100], 0.65)
pogr.array.0.7["ttest",i,1,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                        rownames(ttest.prostate.cDNA)[1:50],
rownames(ttest.prostate.Affy)[1:50], 0.65)



pogr.array2.0.7["ttest",i,4,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                        rownames(ttest.prostate.Affy)[1:1000],
rownames(ttest.prostate.cDNA)[1:1000], 0.65)
pogr.array2.0.7["ttest",i,3,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                        rownames(ttest.prostate.Affy)[1:500],
rownames(ttest.prostate.cDNA)[1:500], 0.65)
pogr.array2.0.7["ttest",i,2,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                        rownames(ttest.prostate.Affy)[1:100],
rownames(ttest.prostate.cDNA)[1:100], 0.65)
```

```
pogr.array2.0.7["ttest",i,1,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    rownames(ttest.prostate.Affy)[1:50],
rownames(ttest.prostate.cDNA)[1:50], 0.65)

npogr.array.0.7["ttest",i,4,j] <- (pogr.array.0.7["ttest",i,4,j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array.0.7["ttest",i,3,j] <- (pogr.array.0.7["ttest",i,3,j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array.0.7["ttest",i,2,j] <- (pogr.array.0.7["ttest",i,2,j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array.0.7["ttest",i,1,j] <- (pogr.array.0.7["ttest",i,1,j] - EPOGR[i,1])/(1-EPOGR[i,1])

npogr.array2.0.7["ttest",i,4,j] <- (pogr.array2.0.7["ttest",i,4,j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array2.0.7["ttest",i,3,j] <- (pogr.array2.0.7["ttest",i,3,j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array2.0.7["ttest",i,2,j] <- (pogr.array2.0.7["ttest",i,2,j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array2.0.7["ttest",i,1,j] <- (pogr.array2.0.7["ttest",i,1,j] - EPOGR[i,1])/(1-EPOGR[i,1]

#############################################################################
##                          2B. E-BAYES (LIMMA) METHOD
#############################################################################

## prostate.cDNA
fit1 <- lmFit(prostate.cDNA.sub, design1)
fitc1 <- contrasts.fit(fit1, contr1)
fitc1 <- eBayes(fitc1)
ebayes.prostate.cDNA <- topTable(fitc1, adjust.method="BH",number=1000)

## prostate.Affy
fit2 <- lmFit(prostate.Affy.sub, design2)
fitc2 <- contrasts.fit(fit2, contr2)
fitc2 <- eBayes(fitc2)
ebayes.prostate.Affy <- topTable(fitc2, adjust.method="BH",number=1000)

## First, simple POG score
pog.array["ebayes",i,4,j] <- length(intersect(ebayes.prostate.cDNA$ID[1:1000],
                    ebayes.prostate.Affy$ID[1:1000]))/1000
pog.array["ebayes",i,3,j] <- length(intersect(ebayes.prostate.cDNA$ID[1:500],
                    ebayes.prostate.Affy$ID[1:500]))/500
pog.array["ebayes",i,2,j] <- length(intersect(ebayes.prostate.cDNA$ID[1:100],
                    ebayes.prostate.Affy$ID[1:100]))/100
pog.array["ebayes",i,1,j] <- length(intersect(ebayes.prostate.cDNA$ID[1:50],
                    ebayes.prostate.Affy$ID[1:50]))/50

## Normalized POG
npog.array["ebayes",i,4,j] <- (pog.array["ebayes",i,4,j] - EPOG[i,4])/(1-EPOG[i,4])
npog.array["ebayes",i,3,j] <- (pog.array["ebayes",i,3,j] - EPOG[i,3])/(1-EPOG[i,3])
npog.array["ebayes",i,2,j] <- (pog.array["ebayes",i,2,j] - EPOG[i,2])/(1-EPOG[i,2])
npog.array["ebayes",i,1,j] <- (pog.array["ebayes",i,1,j] - EPOG[i,1])/(1-EPOG[i,1])

## Now, look at POGR score

## cutoff=0.5
pogr.array["ebayes",i,4,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    ebayes.prostate.cDNA$ID[1:1000], ebayes.prostate.Affy$ID[1:1000],
0.5)
pogr.array["ebayes",i,3,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    ebayes.prostate.cDNA$ID[1:500], ebayes.prostate.Affy$ID[1:500], 0.5)
pogr.array["ebayes",i,2,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
```

```
                    ebayes.prostate.cDNA$ID[1:100], ebayes.prostate.Affy$ID[1:100], 0.5)
pogr.array["ebayes",i,1,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    ebayes.prostate.cDNA$ID[1:50], ebayes.prostate.Affy$ID[1:50], 0.5)

pogr.array2["ebayes",i,4,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    ebayes.prostate.Affy$ID[1:1000], ebayes.prostate.cDNA$ID[1:1000],
0.5)
pogr.array2["ebayes",i,3,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    ebayes.prostate.Affy$ID[1:500], ebayes.prostate.cDNA$ID[1:500],   0.5)
pogr.array2["ebayes",i,2,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    ebayes.prostate.Affy$ID[1:100], ebayes.prostate.cDNA$ID[1:100],   0.5)
pogr.array2["ebayes",i,1,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    ebayes.prostate.Affy$ID[1:50], ebayes.prostate.cDNA$ID[1:50],   0.5)

npogr.array["ebayes",i,4,j] <- (pogr.array["ebayes",i,4,j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array["ebayes",i,3,j] <- (pogr.array["ebayes",i,3,j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array["ebayes",i,2,j] <- (pogr.array["ebayes",i,2,j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array["ebayes",i,1,j] <- (pogr.array["ebayes",i,1,j] - EPOGR[i,1])/(1-EPOGR[i,1])

npogr.array2["ebayes",i,4,j] <- (pogr.array2["ebayes",i,4,j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array2["ebayes",i,3,j] <- (pogr.array2["ebayes",i,3,j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array2["ebayes",i,2,j] <- (pogr.array2["ebayes",i,2,j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array2["ebayes",i,1,j] <- (pogr.array2["ebayes",i,1,j] - EPOGR[i,1])/(1-EPOGR[i,1])


## cutoff=0.7
pogr.array.0.7["ebayes",i,4,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    ebayes.prostate.cDNA$ID[1:1000], ebayes.prostate.Affy$ID[1:1000],
0.65)
pogr.array.0.7["ebayes",i,3,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    ebayes.prostate.cDNA$ID[1:500], ebayes.prostate.Affy$ID[1:500], 0.65)
pogr.array.0.7["ebayes",i,2,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    ebayes.prostate.cDNA$ID[1:100], ebayes.prostate.Affy$ID[1:100], 0.65)
pogr.array.0.7["ebayes",i,1,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    ebayes.prostate.cDNA$ID[1:50], ebayes.prostate.Affy$ID[1:50], 0.65)

pogr.array2.0.7["ebayes",i,4,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    ebayes.prostate.Affy$ID[1:1000], ebayes.prostate.cDNA$ID[1:1000],
0.65)
pogr.array2.0.7["ebayes",i,3,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    ebayes.prostate.Affy$ID[1:500], ebayes.prostate.cDNA$ID[1:500],
0.65)
pogr.array2.0.7["ebayes",i,2,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    ebayes.prostate.Affy$ID[1:100], ebayes.prostate.cDNA$ID[1:100],
0.65)
pogr.array2.0.7["ebayes",i,1,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                    ebayes.prostate.Affy$ID[1:50], ebayes.prostate.cDNA$ID[1:50],   0.65)

npogr.array.0.7["ebayes",i,4,j] <- (pogr.array.0.7["ebayes",i,4,j] -
EPOGR.0.7[i,4])/(1-EPOGR.0.7[i,4])
npogr.array.0.7["ebayes",i,3,j] <- (pogr.array.0.7["ebayes",i,3,j] -
EPOGR.0.7[i,3])/(1-EPOGR.0.7[i,3])
npogr.array.0.7["ebayes",i,2,j] <- (pogr.array.0.7["ebayes",i,2,j] -
EPOGR.0.7[i,2])/(1-EPOGR.0.7[i,2])
npogr.array.0.7["ebayes",i,1,j] <- (pogr.array.0.7["ebayes",i,1,j] -
```

```
EPOGR.0.7[i,1])/(1-EPOGR.0.7[i,1])

npogr.array2.0.7["ebayes",i,4,j] <- (pogr.array2.0.7["ebayes",i,4,j] -
EPOGR.0.7[i,4])/(1-EPOGR.0.7[i,4])
npogr.array2.0.7["ebayes",i,3,j] <- (pogr.array2.0.7["ebayes",i,3,j] -
EPOGR.0.7[i,3])/(1-EPOGR.0.7[i,3])
npogr.array2.0.7["ebayes",i,2,j] <- (pogr.array2.0.7["ebayes",i,2,j] -
EPOGR.0.7[i,2])/(1-EPOGR.0.7[i,2])
npogr.array2.0.7["ebayes",i,1,j] <- (pogr.array2.0.7["ebayes",i,1,j] -
EPOGR.0.7[i,1])/(1-EPOGR.0.7[i,1])


##############################################################################
##                                    2C. SAM
##############################################################################
## prostate.cDNA
data1 <- list(x = prostate.cDNA.sub, y = sub.prostate.cDNA,
              geneid = rownames(prostate.cDNA.sub),
genenames=rownames(prostate.cDNA.sub),
              logged2=TRUE)
res.sam1 <- samr(data = data1, resp.type="Two class unpaired",nperms=50 )
sam.prostate.cDNA    <- sort(abs(res.sam1$tt), decreasing=TRUE)[1:1000]

## prostate.Affy
data2 <- list(x = exprs(prostate.Affy.sub), y = sub.prostate.Affy,
              geneid = featureNames(prostate.Affy.rma3),
genenames=featureNames(prostate.Affy.rma3),
              logged2=TRUE)
res.sam2 <- samr(data = data2, resp.type="Two class unpaired",nperms=50 )
sam.prostate.Affy <- sort(abs(res.sam2$tt), decreasing=TRUE)[1:1000]

## First, simple POG score
pog.array["SAM",i,4,j] <- length(intersect(names(sam.prostate.cDNA)[1:1000],
                          names(sam.prostate.Affy)[1:1000]))/1000
pog.array["SAM",i,3,j] <- length(intersect(names(sam.prostate.cDNA)[1:500],
                          names(sam.prostate.Affy)[1:500]))/500
pog.array["SAM",i,2,j] <- length(intersect(names(sam.prostate.cDNA)[1:100],
                          names(sam.prostate.Affy)[1:100]))/100
pog.array["SAM",i,1,j] <- length(intersect(names(sam.prostate.cDNA)[1:50],
                          names(sam.prostate.Affy)[1:50]))/50

## Normalized POG
npog.array["SAM",i,4,j] <- (pog.array["SAM",i,4,j] - EPOG[i,4])/(1-EPOG[i,4])
npog.array["SAM",i,3,j] <- (pog.array["SAM",i,3,j] - EPOG[i,3])/(1-EPOG[i,3])
npog.array["SAM",i,2,j] <- (pog.array["SAM",i,2,j] - EPOG[i,2])/(1-EPOG[i,2])
npog.array["SAM",i,1,j] <- (pog.array["SAM",i,1,j] - EPOG[i,1])/(1-EPOG[i,1])

## Now, look at POGR score

## cutoff=0.5
pogr.array["SAM",i,4,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    names(sam.prostate.cDNA)[1:1000], names(sam.prostate.Affy)[1:1000],
0.5)
pogr.array["SAM",i,3,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    names(sam.prostate.cDNA)[1:500], names(sam.prostate.Affy)[1:500],
```

```
0.5)
pogr.array["SAM",i,2,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    names(sam.prostate.cDNA)[1:100], names(sam.prostate.Affy)[1:100],
0.5)
pogr.array["SAM",i,1,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    names(sam.prostate.cDNA)[1:50], names(sam.prostate.Affy)[1:50], 0.5)

pogr.array2["SAM",i,4,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                     names(sam.prostate.Affy)[1:1000],names(sam.prostate.cDNA)[1:1000],
0.5)
pogr.array2["SAM",i,3,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                     names(sam.prostate.Affy)[1:500],names(sam.prostate.cDNA)[1:500],
0.5)
pogr.array2["SAM",i,2,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                     names(sam.prostate.Affy)[1:100],names(sam.prostate.cDNA)[1:100],
0.5)
pogr.array2["SAM",i,1,j] <- pogr(cor.prostate.Affy,cor.prostate.cDNA,
                     names(sam.prostate.Affy)[1:50],names(sam.prostate.cDNA)[1:50], 0.5)

npogr.array["SAM",i,4,j] <- (pogr.array["SAM",i,4,j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array["SAM",i,3,j] <- (pogr.array["SAM",i,3,j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array["SAM",i,2,j] <- (pogr.array["SAM",i,2,j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array["SAM",i,1,j] <- (pogr.array["SAM",i,1,j] - EPOGR[i,1])/(1-EPOGR[i,1])

npogr.array2["SAM",i,4,j] <- (pogr.array2["SAM",i,4,j] - EPOGR[i,4])/(1-EPOGR[i,4])
npogr.array2["SAM",i,3,j] <- (pogr.array2["SAM",i,3,j] - EPOGR[i,3])/(1-EPOGR[i,3])
npogr.array2["SAM",i,2,j] <- (pogr.array2["SAM",i,2,j] - EPOGR[i,2])/(1-EPOGR[i,2])
npogr.array2["SAM",i,1,j] <- (pogr.array2["SAM",i,1,j] - EPOGR[i,1])/(1-EPOGR[i,1])

## cutoff=0.7
pogr.array.0.7["SAM",i,4,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    names(sam.prostate.cDNA)[1:1000], names(sam.prostate.Affy)[1:1000],
0.65)
pogr.array.0.7["SAM",i,3,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    names(sam.prostate.cDNA)[1:500], names(sam.prostate.Affy)[1:500],
0.65)
pogr.array.0.7["SAM",i,2,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    names(sam.prostate.cDNA)[1:100], names(sam.prostate.Affy)[1:100],
0.65)
pogr.array.0.7["SAM",i,1,j] <- pogr(cor.prostate.cDNA,cor.prostate.Affy,
                    names(sam.prostate.cDNA)[1:50], names(sam.prostate.Affy)[1:50], 0.65)

pogr.array2.0.7["SAM",i,4,j] <-pogr(cor.prostate.Affy,cor.prostate.cDNA,
                     names(sam.prostate.Affy)[1:1000],names(sam.prostate.cDNA)[1:1000],
0.65)
pogr.array2.0.7["SAM",i,3,j] <-pogr(cor.prostate.Affy,cor.prostate.cDNA,
                     names(sam.prostate.Affy)[1:500],names(sam.prostate.cDNA)[1:500],
0.65)
pogr.array2.0.7["SAM",i,2,j] <-pogr(cor.prostate.Affy,cor.prostate.cDNA,
                     names(sam.prostate.Affy)[1:100],names(sam.prostate.cDNA)[1:100],
0.65)
pogr.array2.0.7["SAM",i,1,j] <-pogr(cor.prostate.Affy,cor.prostate.cDNA,
                     names(sam.prostate.Affy)[1:50],names(sam.prostate.cDNA)[1:50],
0.65)
```

```
npogr.array.0.7["SAM",i,4,j] <- (pogr.array.0.7["SAM",i,4,j] -
EPOGR.0.7[i,4])/(1-EPOGR.0.7[i,4])
npogr.array.0.7["SAM",i,3,j] <- (pogr.array.0.7["SAM",i,3,j] -
EPOGR.0.7[i,3])/(1-EPOGR.0.7[i,3])
npogr.array.0.7["SAM",i,2,j] <- (pogr.array.0.7["SAM",i,2,j] -
EPOGR.0.7[i,2])/(1-EPOGR.0.7[i,2])
npogr.array.0.7["SAM",i,1,j] <- (pogr.array.0.7["SAM",i,1,j] -
EPOGR.0.7[i,1])/(1-EPOGR.0.7[i,1])

npogr.array2.0.7["SAM",i,4,j] <- (pogr.array2.0.7["SAM",i,4,j] -
EPOGR.0.7[i,4])/(1-EPOGR.0.7[i,4])
npogr.array2.0.7["SAM",i,3,j] <- (pogr.array2.0.7["SAM",i,3,j] -
EPOGR.0.7[i,3])/(1-EPOGR.0.7[i,3])
npogr.array2.0.7["SAM",i,2,j] <- (pogr.array2.0.7["SAM",i,2,j] -
EPOGR.0.7[i,2])/(1-EPOGR.0.7[i,2])
npogr.array2.0.7["SAM",i,1,j] <- (pogr.array2.0.7["SAM",i,1,j] -
EPOGR.0.7[i,1])/(1-EPOGR.0.7[i,1])
}}


#########################################################################
##                              Plot the nPOG and nPOGR
#########################################################################

## nPOG

library(lattice)

nPOG.DMD <- read.csv("npog DMD.sub.csv")

nPOG.Lung <- read.csv("npog lung.sub.csv")

nPOG.Prostate <- read.csv("npog prostate.sub.csv")


## Plot all using lattice - xyplot

npog.DMD.mean <- rowMeans(nPOG.DMD)

npog.Lung.mean <- rowMeans(nPOG.Lung)

npog.Prostate.mean <- rowMeans(nPOG.Prostate)


plot.df <- data.frame(npog = c(npog.DMD.mean, npog.Lung.mean, npog.Prostate.mean),

                      data = rep(c("DMD", "Lung cancer", "Prostate cancer"), each=12),

                        subset = c(rep(c(6, 12, 18, 24),3),rep(c(4, 8, 12, 16),3),
rep(c(6,10,20,50),3)),

                      method = rep(rep(c("t-test","eBayes","SAM"), each=4), 3))
```

```
xyplot(npog~subset | data, groups = method, type="l", data=plot.df,
      auto.key=list(columns=3,title="nPOG score for detecting DEG lists from different
datasets of the same diseases" ),layout=c(3,1),
      scales=list(x=list(at=c(4,8,12,18,24,50) )) )


## nPOGR
nPOGR.DMD <- read.csv("npogr DMD.sub.csv")
nPOGR.Lung <- read.csv("npogr lung.sub.csv")
nPOGR.Prostate <- read.csv("npogr prostate.sub.csv")


## Plot all using lattice - xyplot
npogr.DMD.mean <- rowMeans(nPOGR.DMD)
npogr.Lung.mean <- rowMeans(nPOGR.Lung)
npogr.Prostate.mean <- rowMeans(nPOGR.Prostate)


plot.df <- data.frame(npogr = c(npogr.DMD.mean, npogr.Lung.mean, npogr.Prostate.mean),
                      data = rep(c("DMD", "Lung cancer", "Prostate cancer"), each=12),
                       subset = c(rep(c(6, 12, 18, 24),3),rep(c(4, 8, 12, 16),3),
rep(c(6,10,20,50),3)),
                      method = rep(rep(c("t-test", "eBayes",    "SAM"), each=4), 3))



xyplot(npogr~subset | data, groups = method, type="l", data=plot.df,
      auto.key=list(columns=3),layout=c(3,1),
      scales=list(x=list(at=c(4,8,12,18,24,50) )) )


## nPOGR2
nPOGR2.DMD <- read.csv("npogr2 DMD.sub.csv")
```

```
nPOGR2.Lung <- read.csv("npogr2 lung.sub.csv")

nPOGR2.Prostate <- read.csv("npogr2 prostate.sub.csv")


## Plot all using lattice - xyplot

npogr2.DMD.mean <- rowMeans(nPOGR2.DMD)

npogr2.Lung.mean <- rowMeans(nPOGR2.Lung)

npogr2.Prostate.mean <- rowMeans(nPOGR2.Prostate)


plot.df <- data.frame(npogr2 = c(npogr2.DMD.mean, npogr2.Lung.mean,
npogr2.Prostate.mean),

                      data = rep(c("DMD", "Lung cancer", "Prostate cancer"), each=12),

                      subset = c(rep(c(6, 12, 18, 24),3),rep(c(4, 8, 12, 16),3),
rep(c(6,10,20,50),3)),

                      method = rep(rep(c("t-test", "eBayes",    "SAM"), each=4), 3))


xyplot(npogr2~subset | data, groups = method, type="l", data=plot.df,

      auto.key=list(columns=3),layout=c(3,1),

      scales=list(x=list(at=c(4,8,12,18,24,50) )) )



## nPOGR 0.75

nPOGR.DMD.0.75 <- read.csv("npogr 0.75 DMD.sub.csv")

nPOGR.Lung.0.75 <- read.csv("npogr 0.75 lung.sub.csv")

nPOGR.Prostate.0.75 <- read.csv("npogr 0.7 prostate.sub.csv")


## Plot all using lattice - xyplot

npogr.DMD.0.75.mean <- rowMeans(nPOGR.DMD.0.75)

npogr.Lung.0.75.mean <- rowMeans(nPOGR.Lung.0.75)

npogr.Prostate.0.75.mean <- rowMeans(nPOGR.Prostate.0.75)
```

```
plot.df <- data.frame(npogr.0.75 = c(npogr.DMD.0.75.mean, npogr.Lung.0.75.mean,
npogr.Prostate.0.75.mean),

                    data = rep(c("DMD", "Lung cancer", "Prostate cancer"), each=12),

                      subset = c(rep(c(6, 12, 18, 24),3),rep(c(4, 8, 12, 16),3),
rep(c(6,10,20,50),3)),

                    method = rep(rep(c("t-test", "eBayes",    "SAM"), each=4), 3))



xyplot(npogr.0.75~subset | data, groups = method, type="l", data=plot.df,

      auto.key=list(columns=3,title="nPOGR score form list1 to list2 (cutoff=
0.75)"),layout=c(3,1),

      scales=list(x=list(at=c(4,8,12,18,24,50) )) )


## nPOGR2 0.75

nPOGR2.DMD.0.75 <- read.csv("npogr2 0.75 DMD.sub.csv")

nPOGR2.Lung.0.75 <- read.csv("npogr2 0.75 lung.sub.csv")

nPOGR2.Prostate.0.75 <- read.csv("npogr2 0.7 prostate.sub.csv")

## Plot all using lattice - xyplot

npogr2.DMD.0.75.mean <- rowMeans(nPOGR2.DMD.0.75)

npogr2.Lung.0.75.mean <- rowMeans(nPOGR2.Lung.0.75)

npogr2.Prostate.0.75.mean <- rowMeans(nPOGR2.Prostate.0.75)


plot.df <- data.frame(npogr2.0.75 = c(npogr2.DMD.0.75.mean, npogr2.Lung.0.75.mean,
npogr2.Prostate.0.75.mean),

                    data = rep(c("DMD", "Lung cancer", "Prostate cancer"), each=12),

                    subset = c(rep(c(6, 12, 18, 24),3),rep(c(4, 8, 12, 16),3),
rep(c(6,10,20,50),3)),

                    method = rep(rep(c("t-test", "eBayes",    "SAM"), each=4), 3))
```

```
xyplot(npogr2.0.75~subset | data, groups = method, type="l", data=plot.df,

      auto.key=list(columns=3,title="nPOGR score form list2 to list1 (cutoff=
0.75)"),layout=c(3,1),

      scales=list(x=list(at=c(4,8,12,18,24,50) )) )

#########################################################################
##                                The End
#########################################################################
```

Codes for Lung Cancer and DMD studies are similar to the Prostate Cancer data sets.

CURRICULUM VITAE

Dake Yang
E-mail: D0yang03@cardmail.louisville.edu
Address: 725 E Madison Street Louisville, KY, 40202

EDUCATION

2009.08 – Current	MS (candidate, expected in May 2011), Biostatistics University of Louisville (UofL), Louisville, Kentucky, USA
2003.09 – 2007.06	BS, Mathematical Statistics Beijing Institute of Technology (BIT), Beijing, China

RESEARCH EXPERIENCE

2010.08--- Now	Graduate Student (Guy N. Brock) Graduate Thesis—Consistency of Differentially Expressed Gene Rankings Based on Subsets of Microarray Data Dept of Bioinformatics and Biostatistics, School of Public Health and Information Sciences, UofL
2009.08—2010.7	Data Analyst and Bioinformatics (Eugenia Wang) Bioinformatics Data Analysis Gheens Center on Aging, School of Medicine, UofL