Robust High-Dimensional Linear Model Method in Microarray Data Analysis
Xiao Ni, Kejun Liu, Stanley Young, Kevin Gaido

1 Department of Statistics, North Carolina State University, Raleigh, NC 27695-8203, USA,
2 GlaxoSmithKline, Five Moore Drive, Research Triangle Park, NC, 27709-3398, USA,
3 National Institute of Statistical Sciences, PO Box 14006, Research Triangle Park, NC 27709-4006, USA,
4 Hamner Institute for Health Sciences, Six Davis Drive, PO Box 12137, Research Triangle Park, NC 27709-2137, USA

ABSTRACT

Motivation The latest oligonucleotide microarray technologies allow scientists to measure 20,000–200,000 mRNA transcript levels in a single experiment. The sheer number of the data points produces two challenges for statistical inference. First, it currently takes many hours or even days to analyze a typical microarray dataset. This is the computational challenge. Second, each transcript may contain specific outliers and require systematic ways to detect and remove these outliers. This is the outlier challenge.

Results We have developed a robust high dimensional linear modeling (RHDLM) framework to solve the computational challenge and address the outlier challenge. Giving the same results as the industry standard (e.g. SAS® PROC GLM), our methods are 100–10,000 times faster. The statistical inference for 20,000 probesets and 300 arrays only takes 20 seconds on a regular laptop. We also developed a robust approach to efficiently filter outliers for each transcript and refit the model. Simulation studies show that the robust version is more powerful for detecting true signals in the presence of non-trivial outliers.

Availability The approach will be implemented in software called ArrayStudio Lite available at statgen.ncsu.edu/powermarker/aslite. The source code is not available.

1 INTRODUCTION

The use of high-density DNA microarrays for profiling expression of many thousands of transcripts in tissues and cells has increased dramatically in the decade since its introduction. Statistical inference for microarray experiments poses significant challenges for statisticians and computer scientists because of the high dimensionality, the novel structure of microarray data and multiple sources of variability in microarray experiments. For example, a typical microarray dataset generates 20,000 (e.g. GeneChips) - 200,000 (e.g. Exon Arrays) data points for each array and an experiment may contain tens to hundreds of arrays. Typically, efficient statistical analysis for such large datasets is restricted to very simple methods, whereas we extend it to more complex analysis strategies.

One of the most common biological or clinical questions in microarray experiments is to identify differentially expressed genes, whose expression levels are associated with some particular outcome. The techniques to detect significantly differentiated genes have evolved over time. Early microarray studies simply look for genes with a twofold or higher difference between the mean intensities for each group (de Risi et al. 1996). It has been reported that applying an arbitrary cutoff may mask biologically significant changes (Mills and Gordon 2001). From a statistical point of view, ignoring the variation is equivalent to ignoring the statistical significance. Single-array methods are natural extensions of the fold-change method by accommodating gene heterogeneity across all measurements on an array (Chen et al. 1997; Newton et al. 2000). Single-array methods are only applicable to two-color arrays. These early methods are no longer popular. In practice, it is not recommended to use a single replicate to perform microarray analysis (Lee et al. 2000).

Classic statistical approaches used to detect differences between groups include analysis of variance (ANOVA) and the non-parametric counterpart Kruskal-Wallis test (Kruskal and Wallis 1952). For a two-group case, these two methods are equivalent to the parametric t-test and nonparametric Wilcoxon rank sum (Snedecor and Cochran 1980). These methods are used extensively (e.g Tanaka et al. 2000; Arfin et al. 2000). Applying ANOVA to a microarray dataset is straightforward. We model each gene separately and calculate p-values of interesting comparisons. We then use a multiplicity adjustment to control the family wise false positive rate (Westfall and Young 1993) or false discovery rate (Benjamini and Hochberg 1995). This is the “univariate” approach. Univariate simply means the ANOVA model was fitted for each gene separately. This will completely overcome the gene heterogeneity problem. It is argued that the regular t-test or ANOVA approach is not efficient (Newton et al. 2000; Dudoit et al. 2002; Cui and Churchill 2003). First, the power will be low if the sample size is small. Second, the variance estimate for each gene may not be stable. In addition, the evaluation of the statistic for one gene does not use data from any other correlated genes. These arguments led to many extensions or variants of the classic t-test approach. The significance analysis of microarrays” (SAM) version of the t-test (Tusher et al. 2001) adds a small positive constant to the denominator of the gene-specific t test. This diminishes the variance stability problem. The regularized t-test (Baldt and Long 2001) uses weighted average of gene-specific and global variance estimates. The B statistic (Lonnstedt and Speed 2002) allows for gene-specific variances but also combines information across many genes.

* to whom correspondence should be addressed

While these regularized t-tests are theoretically useful and can enable information sharing to some extent, none of them are particularly popular in practice. For most microarray experiments with a good replicate number for each group, the univariate ANOVA approach is still the “gold standard”. There are many reasons why the univariate ANOVA approach is favored in practice. First, when replicate number is large (e.g. ≥ 5 for each group), most of the variants mentioned above will converge to the regular ANOVA (multiple-group case) or t-test (two-group case). The assumptions of regular t-test or ANOVA can usually be met, and the variance estimates are precise enough, which makes the need for information sharing across genes less compelling. Improvement of these variants over the regular method is usually small, and will be diminished further by the total sample size. Second, the improvement comes with increased computational burden. The step down permutation test is a good example. For a large dataset this computationally intensive approach is much more time-consuming. Third, all of these methods do not allow the user to specify continuous covariates or block covariates. More complicated designs including random effects, such as split-plot experiments and repeated measures, were not covered by these methods. Finally, a power calculation is usually not available for most of these ad hoc or Bayesian methods.

The ANOVA based approach described in this paper is not novel. The ANOVA model was first used by Kerr et al. (2005) as an illustration. The RHDLM method is assessed in Section 3 through simulations and introduces the new RHDLM method, where we first develop the concept of the high dimensional linear modeling (HDLM) in this paper. There are two novel features. First, it completely solves the computational challenge inherent in the high dimensional linear model by providing an implementation 100–10,000 faster than SAS® PROC GLM. Second, we propose a robust high-dimensional linear model (RHDLM) framework to incorporate outlier detection in the statistical inference process. This approach integrates outlier-detection and filtration into traditional ANOVA analysis. Unlike most of the current methods in the microarray literature, our method can be applied to essentially any experimental design and can incorporate experiment covariates such as block factors.

The rest of this article is organized as follows. Section 2 introduces the new RHDLM method, where we first develop the high-dimensional linear model framework, and then describe a sequential outlier detection algorithm for robust inference. The RHDLM method is assessed in Section 3 through simulations and real microarray data analysis. In Section 4 we apply the RHDLM to the DBP time series data Thompson et al. (2005) as an illustration. Section 5 concludes the article with a discussion.

2 ROBUST HIGH-DIMENSIONAL LINEAR MODEL METHOD

In this section, we introduce High-Dimensional Linear Models (HDLM), which significantly reduces the computation cost for microarray data analysis compared with traditional linear model methods. We further propose a robust high-dimensional linear model (RHDLM) method, which is robust against extreme observations and hence is more powerful in detecting significant genes for microarray experiments when model assumptions are not met. The RHDLM is implemented on the basis of some novel computing algorithms and therefore is highly efficient.

2.1 High-Dimensional Linear Models

Common design of microarray experiments involves typically one or multiple experimental factors or covariates. After preprocessing steps, the hypothesis testing for differential expression is often done in a general linear model (GLM) framework. Consider a microarray experiment with $n$ chips. For each gene, the microarray data can be represented in the following usual linear model:

$$ Y = X\beta + \epsilon, \quad (1) $$

where $Y$ is $n \times 1$ vector corresponding to gene expression levels, $X$ is $n \times p$ design matrix, $\beta$ is $p \times 1$ vector of fixed unknown coefficients, and $\epsilon$ is $n \times 1$ vector of independent normal random errors with mean 0 and unknown variance $\sigma^2$. Assume that $X$ is full column rank, then $X^T X$ is non-singular and hence the matrix inverse exists. Then the ordinary least squares estimator (OLS) is $\hat{\beta} = (X^T X)^{-1} X^T Y$. Direct inverse of $X^T X$ is computational costly and one of the more efficient ways is through QR decomposition of $X$. It is well known that the matrix $X$ has a QR decomposition

$$ X = QR, \quad (2) $$

where $R$ is $p \times p$ and upper-triangular matrix, and $Q$ is $n \times p$ orthogonal matrix, i.e. $Q^T Q = I$. Using this decomposition, we have $X^T X = R^T Q^T QR = R^T R$ and $(X^T X)^{-1} = (R^T R)^{-1} = R^{-1}(R^T)^{-1}$, where $R$ can be efficiently inverted by backward substitution. The estimator $\hat{\beta}$ can be rewritten as $\hat{\beta} = (R^T R)^{-1}R^T Q^T Y$. Notice that $(R^T R)^{-1}R^T Q^T$ is common for all genes and hence we only need to calculate it once.

In this GLM framework, the null hypotheses of interest are in the following form: $H_0 : L\beta = 0$, where $L$ is a vector or matrix of estimable contrast coefficients. Therefore we test for each gene the hypothesis $H_0$ by forming the following sum of squares

$$ SS(H_0) = (L\hat{\beta})^T (L(X^T X)^{-1}L^T)^{-1}L\hat{\beta} = Y^T_g [X(X^T X)^{-1}L^T(L(X^T X)^{-1}L^T)^{-1}] L(X^T X)^{-1}X^T \hat{\epsilon}_g. \quad (3) $$

From linear model theory, $SS(H_0)$ follows $F$ distribution under the null hypothesis. Notice here in (3) we use $\hat{\epsilon}_g$ to emphasize the fact that only the response vector is gene-specific, whereas matrices inside the brackets are common to all genes. The main computational overhead or bottleneck is the matrix inverse, and indeed it only needs to be computed once. We store the common matrices or their decompositions and share them for computing matrix products for all genes. This is a simple fact but particularly useful for high dimensional data in practice. All other inferential quantities such as Least Squares Means, Type I/III Sum of Squares can be efficiently computed in this high-dimensional linear model framework. For typical microarray experiments with thousands of genes, our HDLM method is extremely efficient. We provide the comparison of computing time in Section 3.

Our method is fundamentally different from the SWEEP operator in SAS PROC GLM (Goodnight 1979). In their method, the
computation always involves \(X\) and \(Y\) simultaneously, whereas our method computes quantities of common \(X\) once and uses \(Y\) only when necessary. The computational challenge posed by modern microarray data has become a bottleneck for using traditional PROC GLM in gene expression analysis. While the SWEEP operator does allow the user to specify multiple \(Y\)s, the algorithm is not designed for more than 1000 \(Y\)s. Our method has tremendous computational advantage to handle an essentially unlimited number of \(Y\)s.

### 2.2 Robust High-Dimensional Linear Models

Microarray data often contain many extreme observations due to multiple sources of variability and stochastic errors introduced in both the experimental and pre-processing stages. This sometimes leads to poor power in discovery. We propose the Robust High-Dimensional Linear Models (RHDLM) which adds robustness to the fast computation benefit. The RHDLM features an outlier detection algorithm which makes our strategy more powerful when the data are contaminated by outliers.

#### 2.2.1 Computation of Residuals

Residuals will be the main tools for outlier detection. The vector of residuals is given by \(e = Y - \hat{Y} = (I - H)Y\), where \(H = X^T(X^TX)^{-1}X\) is the hat matrix. Let \(s^2 = e^Te/(n-p)\) be the usual estimator for \(\sigma^2\). Then the studentized residual (STUDENT in SAS) is given by

\[
r_i = \frac{e_i}{s_i \sqrt{(1-h_i)}},
\]

where \(h_i = x_i^T(X^TX)^{-1}x_i\) is the leverage for the \(i\)-th observation. It can be shown that \(r_i^2/(n-p)\) has a \(\beta\text{eta}(1/2, 1/2(n-p-1))\) distribution (Cook and Weisberg 1982, p.18). The studentized residuals \(r_i\) are identically (but not independently) distributed. The jackknife, or deletion residual (RSTUDENT in SAS) is

\[
t_i = \frac{e_i}{s(i) \sqrt{(1-h_i)}},
\]

where \(s(i)\) denotes the estimator for \(\sigma^2\) with the \(i\)-th observation deleted. It can be shown that jackknife residuals are identically distributed with a \(t\) distribution with \(n-p-1\) degrees of freedom. There is a one-to-one linear relationship between \(r_i^2\) and \(t_i^2\):

\[
r_i^2 = \frac{r_i^2(n-p-1)}{n-p-r_i^2} \sim F(1, n-p-1).
\]

Equivalently we get

\[
r_i^2 = \frac{(n-p)r_i^2}{n-p-1 + r_i^2},
\]

and hence \(r_i^2/(n-p) \sim \beta\text{eta}(1/2, 1/2(n-p-1)).\) Computing the jackknife residuals can be convenient using the following identities:

\[
s(i) = \frac{(n-p)s^2 - e_i^2(1-h_i)}{n-p-1},
\]

\[
t_i = \frac{e_i}{s(i) \sqrt{(1-h_i)}}.
\]

#### 2.2.2 The Sequential Outlier Detection Algorithm

We propose the Sequential Outlier Detection Algorithm (SODA) for sequentially eliminating potential outliers in the sample. The absolute jackknife residuals \(|t_i|\) are identically but not independently distributed. Denote the maximum absolute jackknife residual by \(R = \max|t_i|\). Then we compare \(R\) to a pre-specified critical value to check whether it is a potential outlier. To account for multiplicity, a Bonferroni adjustment is used for the critical value. Let \(n^*\) denote the number of observations in the current sample. Then the Bonferroni-adjusted critical value is \(R^* = t(1 − \frac{1}{2}\alpha/n^*, n^* − p − 1).\) Beckman and Cook (1983) pointed out that the Bonferroni adjustment is essentially exact for maintaining a size \(\alpha\) test, whenever the residual correlations are not excessively large. This conclusion is supported by our simulations (See Section 2.3).

Also note the special treatment to deal with ties among absolute jackknife residuals. It is possible that there are ties among \(|t_i|\), for example, when two jackknife residuals have opposite signs, i.e., \(t_j = -t_i\). This occurs in the situation where two observations with the same \(x\) covariates \((x_j = x_i)\) have the predicted value exactly in the middle point of \(y_j\) and \(y_i\), so that \(t_j + t_i = 0\). For example, in one-way analysis of variance (ANOVA), this could occur in a treatment group with only two replicates whereas the observations are far away from each other.

The algorithm is described below. Beginning with the full sample, set \(n^* = n\).

**Step 1** Fit the GLM and obtain jackknife residuals \(t_i\) and absolute residuals \(|t_i|\).

**Step 2** Sort the sample by \(|t_i|\) and find the largest one, say \(t^* = \max|t_i|\).

**Step 3** Compare \(t^*\) to the critical value \(R^*\), where \(R^* = t(1 − \frac{1}{2}\alpha/n^*, n^* − p − 1)\). If \(t^* > R^*\) then there are no outliers in the current sample and the outlier detection and removal procedure stops. On the other hand, if \(t^* \geq R^*\), then there are two possibilities. (i) If there are no ties observations with the same \(t^*\). Suppose it is the \(k\)-th observation with \(|t_k| = t^*\). Declare \(y_k\) an outlier, and then leave the \(k\)-th observation out, set \(n^* = n^* - 1\) and go to Step 1. (ii) If there are ties, say \(|t_j| = |t_l| = t^*\) and \(t_j + t_l = 0\), stop the procedure.

#### 2.2.3 Efficient Implementation of SODA using Matrix Downdating

The SODA is based on sequential leave-one-out exclusion of outliers, and thus we use matrix downdating technique to make this algorithm highly efficient. Suppose that the \(i\)-th observation \(z\) is identified as an outlier, and we leave it out and re-fit the linear model. It follows that

\[
X^T_{(-i)}X_{(-i)} = X^T X - z^T z.
\]

Now by the new QR decomposition \(X_{(-i)} = \tilde{Q}\tilde{R}\), we have

\[
\tilde{R}^T\tilde{R} = R^T R - z^T z.
\]

This is referred to as a downdating problem. There exist fast algorithms that efficiently downdates \(R\) to \(\tilde{R}\), and therefore we only need to compute one QR decomposition for the full data and avoid recomputing new QR decompositions. In our implementation, we use the LINPACK algorithm due to Saunders (1972) and Stewart (1979). As a result, our robust method is very efficient.

### 2.3 False Outlier Rate

Our SODA procedure detects and then leaves out extreme observations in the sample. We use Bonferroni multiplicity
adjustment in our algorithm, and therefore it is capable of safeguarding the sample from excessive deletion when there are few outliers. We use simple simulations to assess the false outlier rate (FOR) by using the SODA procedure. The FOR is the type I error which is defined as the probability of falsely detecting any observation as an outlier using SODA procedure. Suppose that the sample contains no outliers and we choose the filtration \( \alpha = 5\% \), then the SODA procedure is expected to delete at most 5% of the data. Our simulation results are consistent with this finding. For simplicity, we simulated data from the following model:

\[
Y_{ij} = \tau_i + \varepsilon_{ij}, \quad i = 1, 2, \ j = 1, \ldots, n, \tag{10}
\]

where \( n \) is the number of replicates in each group and \( \varepsilon_i \)'s are independent normal(0,1). Set \( \tau_1 = \tau_2 = 0 \) as their actual values have no effect in this simulation. Three sample size settings are considered, that is, \( n = 5, 10, \) or \( 20. \) The filtration \( \alpha = 1\% \) or \( 5\%. \) We set the Monte Carlo (MC) sample size \( N = 1000. \) After applying the proposed procedure on a simulated random data set (without intentionally introducing outliers into the data), we can estimate the FOR as follows:

\[
FOR = \frac{1}{N} \sum_{i=1}^{N} I(\text{outliers detected in the } i\text{-th MC sample}), \tag{11}
\]

where \( I(\cdot) \) is the indicator function. The FORs are summarized in Table 1. The empirical outlier rate is close to the nominal level.

Table 1. False Outlier Rate for Two-sample case

<table>
<thead>
<tr>
<th>Filtration ( \alpha )</th>
<th>Sample size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td>0.01</td>
<td>0.008</td>
</tr>
<tr>
<td>0.05</td>
<td>0.052</td>
</tr>
</tbody>
</table>

3 NUMERICAL STUDIES

We conducted simulation studies to assess the performance of the RHDL method. For simplicity, we considered two-group data, where model parameters were estimated from the DBP time series data (Thompson et al. 2005) obtained from a real microarray experiment. We compared the SODA-based RHDL method with traditional methods such as Student t and Wilcoxon rank sum tests. To compare the performance of different methods, we plotted the receiver operating characteristic curve (ROC, described later) for visualization and also compute the partial area under the curve (pAUC) as a numerical measure.

3.1 Estimation of Parameters from the DBP Time Series Data

The DBP time series data set contains information for 15923 probe sets in a \( 2 \times 4 \) factorial experiment. There are treatment and control groups, with four time points, and there are three replicates for each time-treatment combination. The response variables are log-2 transformed fold changes. Among the total 15923 probe sets, we identified 1289 that were significantly differentially expressed, after we adjusted for multiplicity using the false discovery rate (FDR), controlled at 0.05. For the 1289 significant probe sets, we estimated treatment effects \( \tau \) and error variances \( \sigma^2 \) and stored them in a data set, from which \( (\tau, \sigma^2) \) were randomly drawn as a pair when we simulated significant genes later. We estimated the \( \sigma^2 \) for all the other genes and pooled them together for sampling purposes.

3.2 Data Simulation and Study Design

We simulated data from the following model:

\[
Y_{gij} = \mu_g + \tau_{gi} + \varepsilon_{gij}, \quad g = 1, \ldots, G; \ i = 1, 2; \ j = 1, \ldots, J, \tag{12}
\]

where \( g \) is the gene index, \( i = 1, 2 \) corresponding to sample 1 and 2, and for each sample there are \( J \) replicates. Since \( \mu_g \) is a nuisance parameter and is not relevant to our analysis, we set \( \mu_g = 0 \) for all genes. We used parameter estimates from the DBP time series data for other model parameters.

We used a factorial experiment with three factors: sample size, percentage of differentially expressed genes, and percentage of outliers. Specifically, we had 1000 genes (\( G = 1000 \)), and for each gene we had two samples, each with sample size \( J = 3, 5, 8 \) or \( 10. \) There were five levels for percent of differentially expressed (significant) genes: 1%, 2.5%, 5%, 10%, and \( 20\%. \) For those significant genes, the treatment effects \( \tau \) and the error variance \( \sigma^2 \) were sampled (with replacement) as a pair from estimation results of the 1289 genes in the DBP time series data; whereas for the insignificant genes, we set \( \tau = 0 \) and sample \( \sigma^2 \) from the results of all 15923 probe sets. We simulated data with various percentage of outliers: 0%, 1%, 2.5%, 5%, 10% and 20%. Since the response variable \( Y_{gij} \) here indicates log-2 transformed fold change as in usual microarray experiments, we simulated an outlier by randomly adding 2 to or subtracting 2 from normally generated \( Y_{gij} \), corresponding to fold change of 4. The total number of settings was \( 4 \times 5 \times 6 = 120. \)

3.3 Comparison of Various Methods

We compared the following four methods for detecting significant genes: (i) \( t \): Regular Student \( t \) test, (ii) \( W \): Wilcoxon rank-sum test, (iii) Robust05: Robust \( t \) test with outlier filtration, filtration \( \alpha = 0.05 \), (iv) Robust10: Robust \( t \) test with outlier filtration, filtration \( \alpha = 0.10 \). We use the receiver operating characteristic (ROC) curves for method comparison.

3.3.1 Receiver Operating Characteristic Curves

The ROC curve has become a popular tool for evaluating the performance of diagnostics medical tests. For diagnostic tests with binary outcomes, conventionally we use positive to indicate presence of disease. Let \( Y \) be the test result and larger values of \( Y \) indicates stronger evidence of disease. Figure 1(a) depicts two hypothetical distributions of \( Y \) for both normal and diseased population. A test with cutoff threshold \( u \) partitions area under the two curves into four parts: true positive (TP), false negative (FN), false positive (FP), and true negative (TN). As shown in Figure 1(b), the ROC curve is the plot of TP rate (or \( 1 - F N \) against FP rate for all possible choices of threshold \( u \). By definition, the ROC curve is a monotone increasing function, whose domain and range are both \([0,1]\). The 45 degree diagonal line corresponds to a random classifier with equal probability of classification to either outcome. For an ideal diagnostic rule, \( TP = 1 \) and \( FP = 0 \); therefore higher ROC curves indicates better tests. In Figure 1(b), ROC I is associated with a better test. In our simulation, all tests are dichotomous, i.e., each gene is classified as either differentially expressed or not. This
significant effect on the performance of different methods. The Wilcoxon test does not achieve sufficient power when sample size is small, therefore it has the lowest pAUCs almost all the time for \( n = 5 \), whereas its power increases when \( n = 10 \). We observe from Table 2 that Robust05 outperforms Robust10 when \( n = 5 \) and vice versa when \( n = 10 \). We find that the percentage of significant genes had little effect.

3.4.2 ROC Plots  Figure 2 shows the ROC curves for four different combinations of sample size, percentage of significant genes and outlier percentage. Plot (a) depicts the case where

\[
pAUC(\delta) = \int_0^\delta ROC(t)dt, \tag{13}
\]

where \( \delta \) is some small false positive rate. As with the ROC curve, higher pAUCs are favored. We recorded the pAUC for each of the 120 experiments. We chose \( \delta = 0.01 \) to focus on the region \([0, 0.01]\), such that the number of false positives ranges from 8 to 9.9, with 10 to 200 true positives. The FDR was controlled between 0.04 and 0.50. We also reported the percentage of genes that were diagnosed to have outliers. The outlier gene percentage should be roughly 5% and 10%, corresponding to the two filtration \( \alpha \) levels.

3.4 Simulation Results

3.4.1 pAUCs  We reported a representative subset of all results from the 120 experiments for illustration and analysis purpose. Table 2 gives pAUCs from the four methods when \( n = 5 \) or 10, and the significant gene percentage was 1%, 5% or 10%. We use abbreviations “R05” and “R10” to label the Robust05 and Robust10 methods respectively.

From Table 2, it can be seen that our SODA based robust methods yield higher pAUCs than the Student’s \( t \) test and the Wilcoxon rank sum test for data with some amount of outlying observations (≥ 2.5% for \( n = 5 \), ≥ 1% for \( n = 10 \)). For small number of outliers (e.g. ≤ 1% for \( n = 5 \)), the false positives produced by the robust methods moderately drags down the ROC curve and also the pAUC. When sample size doubles (\( n = 10 \)), even with 1% of outliers, the Robust05 and Robust10 methods still lead to notable improvement over the \( t \) test. The Robust05 and Robust10 methods had better improvement over the regular \( t \) test when the outlier percentage increased to 10%. For example, for 5% expressed genes and \( n = 5 \), we compute the percent increase for pAUC for 5% outliers as follows: \((R05 – t)/t = (0.608 – 0.563)/0.563 = 24\%\). When outlier percentage increases to 10%, we have percent of improvement \((0.564 – 0.393)/0.393 = 43.5\%\). Sample size has

![ROC curves](image)

Fig. 1. Illustration of ROC curves (a) hypothetical distributions of normal and disease population (b) typical ROC curves
Table 2. pAUC comparison of four methods based on 1000 Monte Carlo samples

<table>
<thead>
<tr>
<th>Percent of significant genes</th>
<th>Outlier percent</th>
<th>pAUC for n = 5</th>
<th>t</th>
<th>pAUC for n = 10</th>
<th>W</th>
<th>R05</th>
<th>R10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>0.887</td>
<td>0.617</td>
<td>0.804</td>
<td>0.761</td>
<td>0.949</td>
<td>0.939</td>
</tr>
<tr>
<td>1.0</td>
<td>0.820</td>
<td>0.582</td>
<td>0.795</td>
<td>0.749</td>
<td>0.888</td>
<td>0.932</td>
<td>0.948</td>
</tr>
<tr>
<td>2.5</td>
<td>0.700</td>
<td>0.526</td>
<td>0.769</td>
<td>0.732</td>
<td>0.806</td>
<td>0.900</td>
<td>0.941</td>
</tr>
<tr>
<td>5.0</td>
<td>0.611</td>
<td>0.475</td>
<td>0.731</td>
<td>0.697</td>
<td>0.707</td>
<td>0.831</td>
<td>0.905</td>
</tr>
<tr>
<td>10.0</td>
<td>0.424</td>
<td>0.375</td>
<td>0.580</td>
<td>0.558</td>
<td>0.606</td>
<td>0.688</td>
<td>0.756</td>
</tr>
<tr>
<td>20.0</td>
<td>0.245</td>
<td>0.223</td>
<td>0.315</td>
<td>0.305</td>
<td>0.403</td>
<td>0.452</td>
<td>0.472</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Percent of significant genes</th>
<th>Outlier percent</th>
<th>pAUC for n = 5</th>
<th>t</th>
<th>pAUC for n = 10</th>
<th>W</th>
<th>R05</th>
<th>R10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0.865</td>
<td>0.609</td>
<td>0.773</td>
<td>0.723</td>
<td>0.946</td>
<td>0.935</td>
<td>0.943</td>
</tr>
<tr>
<td>1.0</td>
<td>0.800</td>
<td>0.573</td>
<td>0.770</td>
<td>0.717</td>
<td>0.864</td>
<td>0.922</td>
<td>0.943</td>
</tr>
<tr>
<td>2.5</td>
<td>0.700</td>
<td>0.539</td>
<td>0.738</td>
<td>0.689</td>
<td>0.783</td>
<td>0.893</td>
<td>0.935</td>
</tr>
<tr>
<td>5.0</td>
<td>0.563</td>
<td>0.460</td>
<td>0.698</td>
<td>0.657</td>
<td>0.68</td>
<td>0.828</td>
<td>0.896</td>
</tr>
<tr>
<td>10.0</td>
<td>0.393</td>
<td>0.363</td>
<td>0.564</td>
<td>0.529</td>
<td>0.531</td>
<td>0.669</td>
<td>0.724</td>
</tr>
<tr>
<td>20.0</td>
<td>0.213</td>
<td>0.214</td>
<td>0.288</td>
<td>0.272</td>
<td>0.322</td>
<td>0.427</td>
<td>0.400</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Percent of significant genes</th>
<th>Outlier percent</th>
<th>pAUC for n = 5</th>
<th>t</th>
<th>pAUC for n = 10</th>
<th>W</th>
<th>R05</th>
<th>R10</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.0</td>
<td>0.856</td>
<td>0.600</td>
<td>0.756</td>
<td>0.702</td>
<td>0.944</td>
<td>0.933</td>
<td>0.943</td>
</tr>
<tr>
<td>1.0</td>
<td>0.795</td>
<td>0.583</td>
<td>0.754</td>
<td>0.699</td>
<td>0.861</td>
<td>0.924</td>
<td>0.941</td>
</tr>
<tr>
<td>2.5</td>
<td>0.694</td>
<td>0.538</td>
<td>0.728</td>
<td>0.674</td>
<td>0.767</td>
<td>0.889</td>
<td>0.931</td>
</tr>
<tr>
<td>5.0</td>
<td>0.560</td>
<td>0.462</td>
<td>0.681</td>
<td>0.635</td>
<td>0.649</td>
<td>0.819</td>
<td>0.892</td>
</tr>
<tr>
<td>10.0</td>
<td>0.380</td>
<td>0.365</td>
<td>0.551</td>
<td>0.521</td>
<td>0.502</td>
<td>0.662</td>
<td>0.718</td>
</tr>
<tr>
<td>20.0</td>
<td>0.201</td>
<td>0.218</td>
<td>0.279</td>
<td>0.258</td>
<td>0.297</td>
<td>0.431</td>
<td>0.388</td>
</tr>
</tbody>
</table>

a t and W indicate student t and Wilcoxon rank sum tests respectively; R05 and R10 correspond to robust GLM with filtration $\alpha = 0.05$ and 0.10 respectively.

Moving from (a) to (b), we doubled the sample size and kept the other factors unchanged. Similarly, from (a) to (c) we increased the percentage of significant genes from 5% to 20%; and in (d) we considered 20% outliers. Robust05 and Robust10 yield ROC curves that dominate the other two methods t and W for false positive rate in [0, 0.02], except for a very small region at early stage. This provides more empirical evidence that, compared to traditional methods, the RHDLM is a more powerful method for finding differentially expressed genes at controlled false positive rates. We use the plots in Figure 2 to examine the effects of different factors. Compared with (a), plot (b) shows that the robust methods are more effective – their ROC curves quickly hike close to 1. There is little difference between plots (a) and (c), which suggests that percent of significantly expressed genes has little effect on the power of the analysis. Contrasting (a) with (d), we see that all four ROC curves are significantly lower when outliers dramatically increase. In summary, the RHDLM method is robust and more powerful when outliers are present.

Next we present in Figure 3 two ROC plots for n = 5 and n = 10 with 5% significant genes and no outliers. Plot (a) shows that when no outliers are introduced and sample size is small (n = 5), the t test yields slightly higher ROC curve than the RHDLM. However, in (b) where n = 10, there is essentially no loss by using RHDLM.

3.4.3 Comparison of Computation Overhead In addition to robustness, RHDLM is of huge computational advantage with minimal cost compared with traditional methods. We compared our method to SAS PROC GLM, which is commonly used on for gene analysis in microarray data analysis. Four methods were compared: PROC GLM (non SODA), PROC GLM with SODA, HDLM (non SODA), and RHDLM. For illustration, we recorded the running times for scenarios of n = 5, 5% significant genes and 5% outliers, using the four different methods. The results are summarized in Table 3. The high-dimensional methods significantly reduced the computational time, using less than 1% of the time used by the counterpart methods. By using efficient downdating algorithms, the RHDLM adds robustness to HDLM with only a slight increase in computing time, compared to nearly tripled computing time for GLM with SODA.

Table 3. Computing time (sec.) comparison of four computational methods based on simulated data.

<table>
<thead>
<tr>
<th>Method</th>
<th>non SODA</th>
<th>SODA</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLM</td>
<td>138.4</td>
<td>377.5</td>
</tr>
<tr>
<td>HDLM</td>
<td>1.3</td>
<td>1.6</td>
</tr>
</tbody>
</table>

4 APPLICATION TO DBP TIME SERIES DATA We applied the proposed RHDLM method to the DBP time series data (Thompson et al. 2005). We computed the treatment-versus-control contrast at each time level, conducting a total of four hypothesis tests for each of the 15923 genes. Instead of using FDR for multiplicity as in Section 3.1, here we used the Bonferroni adjustment. A gene is considered to be statistically significant if any of the four adjusted p values is less than or equal to 0.05. Similar to the simulations, we look at two aspects regarding the performance of RHDLM: discovery power and computing time. Table 4 reports the number of significantly expressed genes found by four methods. The computing time (sec.) of four methods is included in parentheses.
Table 4. Number of significant genes found in the DBP time series data using four computational methods

<table>
<thead>
<tr>
<th>Method</th>
<th>non SODA</th>
<th>SODA</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLM</td>
<td>89 (3394.6)</td>
<td>116 (7772.5)</td>
</tr>
<tr>
<td>HDLM</td>
<td>89 (5.7)</td>
<td>116 (21.3)</td>
</tr>
</tbody>
</table>

a Thompson et al. (2005)
b Computing time (sec.)

Table 4 reveals the significant computational advantage of HDLM. For example, the computing time of GLM non SODA is 3394.6 seconds, whereas the HDLM takes less than 0.2% of that. Furthermore, the filtration of outliers leads to 116 significant genes, up from 89 in the no filtration case. The significant genes found by RHDLM are of biological interest, according to the results in Thompson et al. (2005). This coincides with our simulation findings and provides strong evidence that the RHDLM is more powerful and at the same time maintains low false positive rates.

5 SUMMARY AND DISCUSSION

In this article we proposed a new method for analyzing high-dimensional data. Based on the special structure of microarray data, we developed a high-dimensional linear model framework to significantly reduce the computation cost. We proposed a sequential outlier detection algorithm (SODA) based on leave-one-out of jackknife residuals. The SODA was efficiently implemented using matrix downdating technique and hence added minimal computing cost. Combining the SODA and HDLM, we arrived at a powerful and efficient RHDLM method. Compared with traditional methods, the RHDLM is faster and more powerful for detecting differentially expressed genes among tens of thousands of genes.

We demonstrated the effectiveness of RHDLM through Monte Carlo simulations and real data analysis. Empirical results show that our method has overall better performance than traditional methods in terms of ROC and pAUC. In addition, the RHDLM costs less than 1% of the computing time of traditional methods. In our simulations, the SODA works more favorably when total sample size is ≥ 10. This could be due to the fact that we compute the critical values according to a student t distribution with the degree of freedom determined by the total sample size. Therefore we recommend to use this feature for analyzing microarray data sets with at least 10 arrays (which is usually the case). Analysis of the DBP time series data (with 24 samples) using the RHDLM method not only significantly reduced the computing time, but also returns more significant genes that have been confirmed to be biologically meaningful. This new method has been used to analyze dozens of experiments in a major pharmaceutical company and has been applied to over ten different experimental designs. Although our method is designed for Affymetrix oligonucleotide microarray technology, it can be applied to any omic platforms directly.

In summary, the RHDLM is a new powerful and efficient method that is particularly useful for analyzing typically noisy microarray data. Since this method is based on general linear models, it is capable of accommodating more complex microarray experiments. The huge computational advantage makes it a desirable discovery method for biologists and bioinformaticians.

ACKNOWLEDGEMENT

We are indebted to our colleagues David Cooper, Lei Zhu and Katja Remlinger. Kejun (Jack) Liu was a joint postdoc at National Institute of Statistical Sciences and Hammer Institute for Health Sciences. Xiaojun Ni is a Graduate Industrial Trainee at GlaxoSmithKline.

REFERENCES


