Singular Value Decomposition applied to the building of class predictor

Yi Ding
05138398
BIOC 218
Final Project

SINGULAR VALUE DECOMPOSITION APPLIED TO THE BUILDING OF CLASS PREDICTOR

1. INTRODUCTION:

Advances in microarray technology have made possible the measurement of gene expression data on a genomic scale (Spellman *et. al* 1998, Eisen *et. al* 1998, Golub *et. al* 1999). The outputs of the experiments are expression profiles either sampled at different times or from different sources (patients belonging to different phenotype). This has a profound impact on the study of human diseases. By comparing the differentially expressed profiles, we can find out the mechanism of gene expression, hence obtain information useful for clinical diagnosis and drug design.

In the study of cancer genomics, its two major goals are: 1. to find out informative genes of cancer (genes that are mostly differentially expressed between classes of phenotype).2. To build a class predictor from these informative genes that could be applied to other samples.

The typical approach to select informative genes is to rank genes according to their expression level using a test statistic (t-test, signal-to-noise ratio, etc.) (Golub et. al 1999, Tusher et. al 2001), and choose the top ranked genes as informative genes in building a class predictor using "weight vote"(Golub et. al 1999) or KNN method. However, because of the inter-regulation of gene expression and the deficiency of sample density, the gene expression data are highly correlated.
Predictors built by these methods may often lead to misclassification since they are based on inadequate amount of information (Jaeger et. al 2003, Ghosh, 2001).

To overcome this flaw, Jaeger proposed a method that first performs clustering, and then chooses the highest ranked genes from the different clusters to build a predictor (Jaeger *et. al* 2003). This method decorrelates the informative genes, and thus may achieve better classification accuracy. Unfortunately, however, the clustering algorithm dramatically adds to the computational expenses. In this paper, the author proposes a new decorrelating scheme using singular value decomposition (SVD).

SVD is often referred to as Principle Component Analysis (PCA) in statistics (Golub and Van Loan, 1996). In this paper, to de-correlate the highly ranked genes, SVD is performed to transform the data from gene*array space to the eigengene * eigenarray space, where the eigengene expressions are mutually orthogonal.(Alter *et. al* 2003). After de-correlation, a subset of the eigengene was selected to build a predictor with weighted voting method. And a comparison with the scoring method will evaluate the performance of the predictor.

2. MATERIALS AND METHODS:

2.1 Microarray

Two microarray data sets are used in the present study: 1. Classification of acute leukemias classification sample (Golub *et. al* 1999). 2. Prostate tumor and normal samples (Ramaswamy and Golub 2002). Both data sets are available in the Affymetrix format at http://www-genome.wi.mit.edu/cgi-bin/cancer/datasets.cgi

For the Leukemias data, the training set contains 27 samples of ALL (acute lymphoblastic leukemia) and 11 samples of AML (acute myeloid leukemia), the test set contain 24 samples of ALL and 11 samples of ALL. The prostate cancer data set contains 50 normal samples and 52 tumor samples. The author chose 20 normal sample and 20 tumor sample as the training set, the others forms a test set.

2.2 Data preprocessing

Because the microarray is often highly skewed, Log transform is recommended to smooth out the data.(Yang *et. al*). To remove the experimental artifacts and random noise, the frequently used techniques include scaling and regression (with regard to some 'housekeeping' genes or bases). However, as the White Head data are already transformed and filtered, this paper will not discuss detailed procedures.

2.3 Marker selection

A subset (1000 for instance) of the highly differentially expressed genes is chosen for SVD calculation. To locate the marker genes, a t-statistic or signal to noise ratio (given below) was calculated for each gene. The t-statistic is given by $\frac{\mu_1 - \mu_2}{(\sigma_1^2 + \sigma_2^2)^{\frac{1}{2}}}$

and the signal to noise ratio is given by $\frac{\mu_1 - \mu_2}{\sigma_1 + \sigma_2}$, where μ_1 and μ_2 are the mean expression in class 1, and class 2 while σ_1 and σ_2 are the respective variance within respective classes. The marker genes are those that rank highest in the statistics. .(Typically a permutation test is applied to randomly permute the samples and the statistics is calculated in each permutation, finally the genes with the smallest t-test p-value are chosen. Because of limited time and those genes ranked highest often coincides with the ones with the smallest p-value, we omit this procedure.)

2.4 SVD calculation

A $m \times n$ matrix A has the following Singular Value Decomposition

$$A = U\Sigma V^T$$

Where $U^T U = I_m$, $V^T V = I_n$ the columns of U is called left-eigenvector which correspond to an eigenarrays and the columns of V are called right-eigenvector which correspond to eigen genes (8). Since for microarrays we have m>n, we can make use of the so-called 'economic SVD'

$$A = \overline{U}\Sigma V^T$$

Where $\overline{U}$ is m-by-n. In this method we only have to calculate the first n column of $\overline{U}$, hence computational expenditure is greatly reduced. In this paper SVD is calculated via MATLAB.

After performing SVD, we have n eigen genes that are mutually orthogonal. Then we can choose a subset of these eigen genes to form a feature set, before classifying and predicting in the space spanned by this eigen genes.

Using SVD, dimension reduction and orthogonal feature extraction are accomplished at the same time(see figure 1.).

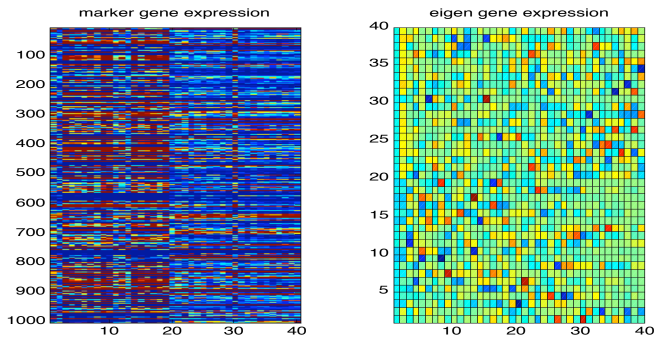

Figure 1. The color image of the expression profiles of 1000 marker genes and the pseudo color map of the 40 eigen genes in the prostate cancer data set

2.5 Building a predictor via weighted voting

In the weighted voting algorithm, each input (gene) is assumed independent and each contributes a weight or vote for a class; the class receiving the greatest number of votes is the predicted class. The decision boundary is chosen to be half way between the class means. There are many sophisticated methods for classification, such as Self-Organizing –Map (Golub *et. al* 1999), support vector machine (Jaeger et. al 2003) multilayer perceptron (Mateos *et. al* 2002). Because of these algorithms uses nonlinear decision boundary, they are more powerful in classification. Yet in the presence of outlier, these methods tend to be overfit. Since the purpose of this paper is to study the effects of SVD as a preprocessing procedure, the author will focus on weighted voting method.

$b_i = \frac{1}{2}(\mu_{1i} + \mu_{2i})$ for each gene in the feature set.

The Vote of each gene is given by $V_i = S_i(g_i - b_i)$

where g_i is the expression level and S_i is the signal to noise ratio.

Finally, the class is given by $sign(\sum_i V_i)$ and the prediction strength is given by

$\frac{V_{win} - V_{lose}}{V_{win} + V_{lose}}$ (Golub *et. al* 1999).

3. RESULTS AND DISCUSSION

For both data sets, we performed SVD to the marker genes(selected) to generate the eigen genes, weighted voting was then used to generate a decision rule that can be applied to both the training set and test set. We then compare the accuracy of this method (eigen gene) to the commonly used marker gene method.

For both data sets, higher accuracy was achieved in almost all cases by eigen gene method when using same number of features. (See figure 2 and 3). We also notice that eigen gene predictor performs much better than the marker gene in the test set. This is because more information is contained in the de-correlated eigen genes hence is more accurate and robust.

In the experiment results we also find that adding more features will not necessarily increase accuracy, especially for the marker gene method. As for eigen gene method, a very small subset of eigen genes are required to achieve a certain level of accuracy , for the leukemias example, 6 eigen genes gives prediction of 100% accuracy within the training set and 80% within the test set. Adding more eigen genes will not significantly improve the accuracy, since the those eigen genes may represent noise or experimental artifacts.

One defect of the eigen gene predictor is that it’s prediction strength (Golub et. al 1999) is not as high as the marker genes. The mean prediction strength of marker gene in the leukemia data is above 60% for both the training and testing set while the prediction strength of the eigen gene is around 47% in the training set and 34% in the testing set . But this could be overcome if we have a training set large enough.

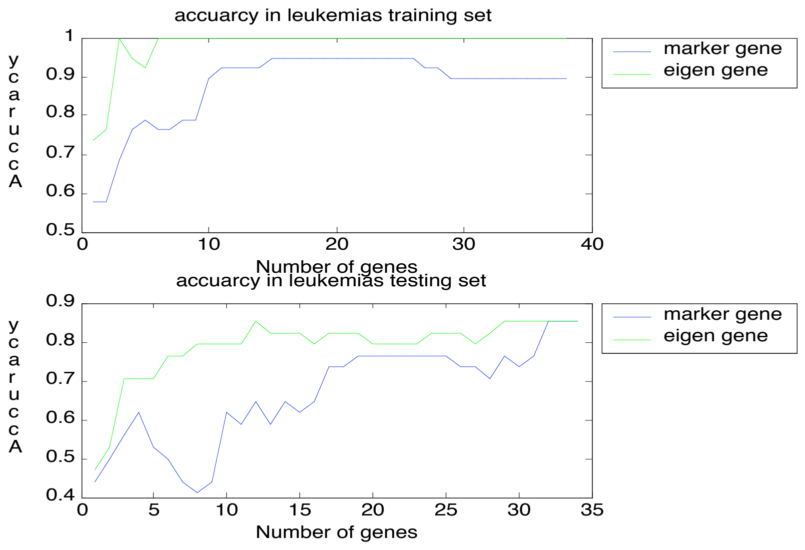

Figure 2. Accuracy Vs. feature plot for the Leukemia data sets

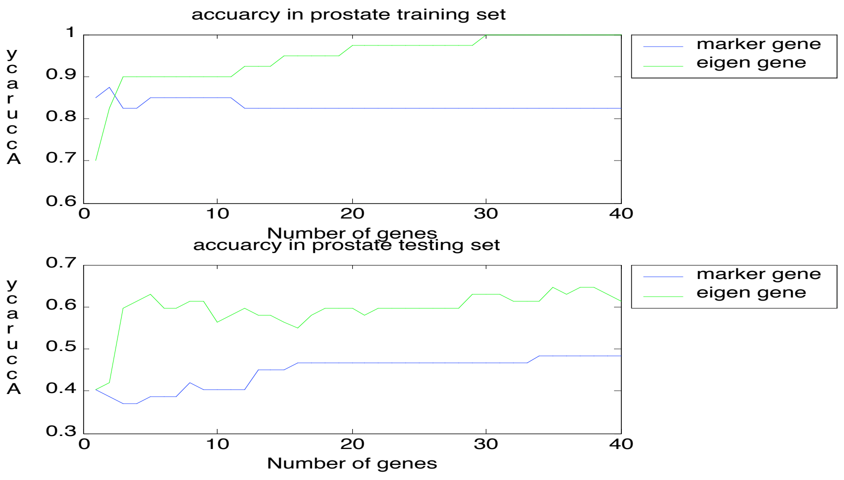

Figure 3. Accuracy Vs. Feature plot of the Prostate cancer data sets, the marker genes are selected according to the signal to noise ratio.

4. CONCLUSION

In this paper, we have developed a singular value decomposition method to generate eigen genes that can be used for classification of clinical phenotypes. The ability of de-correlating and feature extraction makes SVD a superior tool in data preprocessing is testified.

REFERENCES

1. Alter ,O., Brown, P.O. and Botstein , D. 2000) *Proc. Natl. Acad. Sci. USA,* 97: 10101-10106
2. Eisen, M.B. Spellman, P. T., Brown, P. O., and D. Bostein. 1998. *Proc.Natl.Acad.Sci. USA,* 95: 14683-14868.
3. Ghosh, D. "Singular value decomposition regression models for Classification of tumors from micro array experiments"
4. Golub T. R. et. al. 1999. "Molecular classification of cancer: Class Discovery and class prediction by gene expression monitoring." *Science,* 286: 531-537.
5. Golub, G.H. and Van Loan, C.F. 1996 Matrix *Computation* Johns Hopkins Univ. Press, Baltimore, 3rd Ed.
6. Jaeger,J. Sengupta, R. and Ruzzo, W. L. 2003. "Improved gene selection for classification of microarrays." *Pac Symp Biocomput*: 53-64.
7. Nadon, R. and Shoemaker, J., "Statistical issues with microarrays: processing and analysis." *Trends in Genetics,* 18: 265-271,2002
8. Ramaswamy,S., and Golub, T.R., 2002 "DNA Microarrays in Clinical Onltology." *Journal of Clinical Oncology,* Vol 20, No.7, 2002: 1932-1941
9. Spellman, P. T., Sherlock, G., Zhang, ,M. Q., Iyer,V.R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., and Futcher ,B. 1998 *Mol.Biol.Cell,* 9: 3273-3297.
10. Tusher, V.G., Tibshirani, R. and Chu, G. 2001 "Significance analysis of microarrays applied to ionizing response." *Proc Nat Acad Sciences,* 98: 5116-5121
11. Yang, Y.H., Dudoit,S., Luu,P. and Speed,T.S. 2002 "Normalization for cDNA microarray data." *Nucleic Acids Res,* 2002 Feb 15; 30(4):e15.
12. Mateos A, Dopazo J, Jansen R, Tu Y, Gerstein M, Stolovitzky G. Systematic learning of gene functional classes from DNA array exp ression data by using multilayer perceptrons, Genome Res. 2002 Nov;12(11):1703-15.

Appendix:

A Matlab script for this project:

```
% Define problem size
m=7129; n=38;c1=27;c2=11;
R=[ones(1,c1),-ones(1,c2)];
% I/O operations
yd1=fopen('train38.txt','r');
[A,count]=fscanf(yd1,'%8f',[n,m]);
A=A';
[m,n]=size(A);
% Marker gene selection
%===============================================================
========%
% compute the test statistics
stat4=zeros(m,4);
stat4(:,1)=mean(A(:,1:c1),2);
stat4(:,2)=mean(A(:,c1+1:n),2);
stat4(:,3)=std(A(:,1:c1)')';
stat4(:,4)=std(A(:,c1+1:n)')';
t_stat=(stat4(:,1)-stat4(:,2))./(stat4(:,3))+sqrt(stat4(:,4));
s2n_stat=(stat4(:,1)-stat4(:,2))./sqrt(stat4(:,3).^2+stat4(:,4).^2);

[y1,i1]=sort(abs(t_stat));
[y2,i2]=sort(abs(s2n_stat));

m1=1000; %chose # of marker
marker_1=i1(m-m1+1:m); % retrieve marker index
marker_2=i2(m-m1+1:m);
A_bar1=A(marker_1,:);
A_bar2=A(marker_2,:);


% svd computation
%===============================================================
=====%
[U1,S1,V1]=svd(A_bar1,0);
[U2,S2,V2]=svd(A_bar2,0);
% color image generation
subplot(121)
image(A_bar1)
title('marker gene expression')
```

```
subplot(122)
pcolor(V1)
title('eigen gene expression')


% Building a class predictor
%==========================================================================%
NPC=20; % choose number of principle components
PC1=V1(:,1:NPC)';
PC2=V2(:,1:NPC)';
% Using weighted voting to build predictor
% compute the decision boundary
stat4=zeros(NPC,n);
stat4(:,1)=mean(PC1(:,1:c1),2);
stat4(:,2)=mean(PC1(:,c1+1:n),2);
stat4(:,3)=std(PC1(:,1:c1)')';
stat4(:,4)=std(PC1(:,c1+1:n)')';
D_bs=.5*(stat4(:,1)+stat4(:,2)); % decision boundary
s2n_stat_S=(stat4(:,1)-stat4(:,2))./(stat4(:,3)+stat4(:,4)); % compute s2n ratio
RESULT_S=zeros(1,n); PS_S=zeros(1,n);
for i=1:n
   Vote=s2n_stat_S.*(PC1(:,i)-D_bs);
   RESULT_S(i)=sign(sum(Vote));
   i_n=find(Vote<0);i_p=find(Vote>0);
   PS_S(i)=abs(sum(Vote(i_p))-
abs(sum(Vote(i_n))))/(abs(sum(Vote(i_n)))+sum(Vote(i_p)));
end



% comparison with the highest-ranked gene method
% retrieve top NPC genes
top1=i1(m-NPC+1:m);
sub1=A(top1,:);
% compute correlation between these genes
COR_top=corrcoef(sub1');
COR_top=tril(COR_top);
mean_COR_top=mean(mean(COR_top),2);


%% Using weighted voting to build predictor
%% compute the decision boundary
stat4=zeros(NPC,n);
stat4(:,1)=mean(sub1(:,1:c1),2);
stat4(:,2)=mean(sub1(:,c1+1:n),2);
```

```
stat4(:,3)=std(sub1(:,1:c1)')';
stat4(:,4)=std(sub1(:,c1+1:n)')';
% compute decision boundary

% cross validation

%
D_b=.5*(stat4(:,1)+stat4(:,2));
s2n_stat=(stat4(:,1)-stat4(:,2))./(stat4(:,3)+stat4(:,4)); % compute s2n ratio
RESULT=zeros(1,n);PS=zeros(1,n);
for i=1:n
   Vote=s2n_stat.*(sub1(:,i)-D_b);
   RESULT(i)=sign(sum(Vote));
   i_n1=find(Vote<0);i_p1=find(Vote>0);
   PS(i)=abs(sum(Vote(i_p1))-
abs(sum(Vote(i_n1))))/abs(abs(sum(Vote(i_n1)))+sum(Vote(i_p1)));
end
% Display results
%==============================================================
==========%
mean_COR_top              % correlation matrix
RESULT                    % prediction of marker gene
RESULT_S                  % prediction of eigen gene
corr=sum((RESULT==R),2);     % total number of TP using marker
corr_S=sum((RESULT_S==R),2);  % total numbe of TP using eigen
PS_mean=mean(PS,2)           % mean prediction strength of marker gene
PS_S_mean=mean(PS_S,2)        % mean prediction strength of eigen gene
```