12-1-2009

Some Contributions in Statistical Discrimination of Different Pathogens Using Observations through FTIR

Dongmei Wang
Georgia State University

Some Contributions in Statistical Discrimination of Different Pathogens Using Observations Through FTIR

by

Dongmei Wang

Under the Direction of Yu-Sheng Hsu

ABSTRACT

Fourier Transform Infrared (FTIR) has been use to discriminate different pathogens by signals from cells infected with these versus normal cells as references. To do the statistical analysis, Partial Least Square Regression (PLSR) was utilized to distinguish any two kinds of virus-infected cells and normal cells. Validation using Bootstrap method and Cross-validations were employed to calculate the shrinkages of Area Under the ROC Curve (AUC) and specificities

corresponding to 80%, 90%, and 95% sensitivities. The result shows that our procedure can significantly discriminate these pathogens when we compare infected cells with the normal cells. On the height of this success, PLSR was applied again to simultaneously compare two kinds of virus-infected cells and the normal cells. The shrinkage of Volume Under the Surface (VUS) was calculated to do the evaluation of model diagnostic performance. The high value of VUS demonstrates that our method can effectively differentiate virus-infected cells and normal cells.

INDEX WORDS: FTIR, PLSR, AUC, Specificity, Sensitivity, VUS

Some Contributions in Statistical Discrimination of Different Pathogens Using Observations Through FTIR

by

Dongmei Wang

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science
in the College of Arts and Sciences
Georgia State University
2009

Some Contributions in Statistical Discrimination of Different Pathogens Using Observations Through FTIR

by

Dongmei Wang

Committee Chair: Dr. Yu-Sheng Hsu

Committee: Dr. Julia Hilliard

Dr. Jiawei Liu

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

December 2009

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to all those who helped me during the writing of this thesis.

My deepest gratitude goes first and foremost to Professor Yu-sheng Hsu, my supervisor, for his constant encouragement and guidance. He has walked me through all the stages of the writing of this thesis. Without his consistent and illuminating instruction, I could never have the precious opportunity to study at GSU, nor does the opportunity to do the research, and this thesis could not have reached its present form.

Secondly, I would like to express my heartfelt gratitude to the committee member Dr. Julia Hilliard who is from Biology Department and Dr. Jiawei Liu for taking the time to review my work and providing me with valuable feedback.

I also owe special debt of gratitude to my friends Baoying Yang, Ye Cui, Zhibo Wang, Huayu Liu, Shenjia Zhang, Ji Li, Tian Tang, Shan Luo and Yueheng An who gave me their unrequited help during the hardest time. Their concern and encouragement is my greatest driving force for progress.

Also, I would like to thank Ruili and Jing Guo for offering the original data and patient explanation the meaning of the data which is very important for the whole process of data analysis.

Last my thanks would go to my beloved family for their loving considerations and great confidence in me all through these years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

Early detection of diseases has become mainstream in the medical world. This is because many diseases can be controlled in the early stages. Furthermore, the treatment may be more effective at the early stage of the disease, too. Therefore, it will be of great significance for us to develop some approaches for fast and accurate identification of pathogens. Studies showed that specific pathogens have developed unique countermeasures, which can be interrupted by pathogens subverting the innate defenses of host cells. The sequence of interruptions can be detected and differentiated using Fourier Transform Infrared (FTIR) technology even for closely related viruses [1].

Fourier transform spectroscopy is a measurement technique for collecting spectra to measure the coherence of radiation resource. FTIR spectroscopy is one of the widely used applications of Fourier transform spectroscopy for collecting infrared spectra[2]. It uses Infrared (IR) light guided through an interferometer to pass through the sample to get interferogram of measured signal and then performs a Fourier transform on this signal data which is identical to that from conventional IR spectroscopy but more sensitive and has a much shorter sampling time than conventional one. In addition, FTIR technique allows collecting the information at all frequencies at the same time such that multiple samples can be collected and averaged together to improve the sensitivity. Plentiful research has indicated that FT-IR can be applied as spectral biodiagnosis method (Shan-Yang Lin, Mei-Jane Li and Wen-Ting Cheng, 2007; Salmn et al. 2002; Alam et al. 2004; Burattini et al. 2008).

However, most of this study only focused on the possibility of differentiating normal cells from infected cells without considering the time post cell infection. In this study the identification of pathogens within a shorter time post infection will be discussed not only for the differentiation of any two of normal cells with infected cells, but also for differentiation of three kinds of pathogens infecting the cells at the same time post infection.

Healthy monkey kidney cells (ATTC Lot#CCL-81) were grown in selected laboratory environments and then exposed to herpes simplex virus-1 (HSV-1), MacIntyre strain (ATTC Lot# VR-539), Coxsackie (ATTC Lot#, and Adenovirus (ATCC Lot#. Normal cells were denoted as mock infected. The absorbance sequence of IR spectra of these cells at 2, 4, 6, 12, and 24 hours post infection were measured. For each observation, 728 measurements were taken on the wave number range of 800-1500 cm-1.

Some work has been done by Tian Tang[3] and Shan Luo[4]. Tian Tang has shown that Partial Least Square Regression (PLSR) is suitable to do the data analysis in this study. Shan Luo has shown that 6 hours is sufficient to discriminate any two kinds of viruses except for discrimination between uninfected cells and adenovirus infected cells the comparison of which were ignored in this study. Since Shan Luo has done the two dimensional discrimination and find the 95% confidence interval by bootstrap method and 2-fold cross validation method for 6 hours data, we will continued her study by calculating the shrinkage to evaluate the diagnostic performance of the whole procedure. Furthermore, we studied the discrimination

among uninfected, HSV1, and Coxsackie virus infected cells simultaneously instead of pairwise comparisons. The results are encouraging.

The thesis is organized as follows: In chapter II, The methodologies utilized in the thesis are introduced which includes data processing, standardization, Wilcoxon Sign Rank Test (WSRT) , PLSR for discriminating two and three kinds of cells respectively, and evaluation of diagnostic performance of PLSR. In chapter III, the results and explanations of study are presented according to the procedure of data analysis. The conclusion and possible future work of the study will be given in chapter IV.

CHAPTER 2 METHODOLOGY

2.1 Data Construction (inner intra)

2.1.1 Standardization and Average

Overall, we have 21 data sets for Mock, 21 for HSV1, 20 for Adeno and 18 for Coxsackie which yield to 21 paired comparisons for Mock and HSV1 infected cells, 20 paired comparisons for HSV1 and Adeno infected cells, 17 paired comparisons for Adeno and Coxsackie infected cells, 18 paired comparisons for HSV1 and Adeno infected cells, 18 paired comparisons for Mock and Adeno infected cells, and 18 paired comparisons for Mock and Coxsackie infected cells. Each data set includes 30-80 observations.

It is found that the variation of each observation is very large. To make the data more comparable, we standardize each observation by the formula as following:

$$y_{ji} = \frac{x_{ji} - \bar{x}_j}{s_j} \qquad i = 1, 2, \ldots, 728$$

Where x_{ji} is the i th variable in j th observation, $\bar{x}_j = \frac{1}{728}\sum_{i=1}^{728} x_{ji}$ is the mean of 728 variables in j th observation, and $s_j = \sqrt{\frac{1}{728}\sum_{i=1}^{728} (x_{ji} - \bar{x}_j)^2}$ is the standard deviation of j th observation. To make smooth the data, the average of every 4 neighboring variables was taken using

$$c_i = \frac{1}{4}(\sum_{k=1}^{4} y_{4i+k-4}) \qquad i = 1, 2, \ldots, 182$$

where c_i is the new averaged variable, Therefore, for each data set we have 30-80 observation and 182 variables.

2.1.2 Data Constructing – inner and intra

From the preliminary study we found that the data sets of FTIR readings are inconsistent from one data to another, which is caused by the variability within and between each data set. To avoid the variability among different dates, we construct inner data set and intra data set based on the data from the same date. Take one Mock data set and one HSV1 data set as example. We randomly divide the Mock data and HSV1 data into two groups, respectively, which can be denoted as M1, M2 and H1, H2. Then we take the average of each data group, which can be denoted as m1, m2 and h1, h2. Subsequently, we construct inner and intra data by:

Inner1=m1-m2, Inner2=h1-h2,

Intra1=m1-h1, Intra2=m2-h2.

Since we can obtain 2 inner data sets and 2 intra data sets from each data set, we totally have 42 inner and intra observations for Mock and HSV1 infected cell comparison, 40 inner and intra observations for HSV1 and Adeno infected cell, 34 inner and intra observations for Adeno and Coxsackie infected cells, 36 inner and intra observations for HSV1 and Adeno, infected cells and 36 inner and intra observations for Mock and Coxsackie, infected cells. Moreover, since we are using averages of many absorbance, the inner and intra differences may be considered as having normal distributions by the Central of Limit Theorem.

2.2 Wilcoxon Sign Rank Sum Test (WSRT)

The purpose we applied WSRT on intra data sets is to obtain the frequencies where the inner and intra variables are significant different from zero. WSRT is a

non-parametric test, which is used to test the null hypothesis that the mean of a distribution is equal to some value. We use WSRT in place of a one-sample t-test because it is very robust to the influence of outlier and it does not require assumptions about the form of the distribution of the measurements. Furthermore, using ranks instead of the real values may have more discriminating powers.

Take Mock and HSV1 infected cells as example, for each variable the null hypothesis is that the mean of intra data, M is equal to zero, that is

$$H_0 : M = 0 \qquad H_a : M \neq 0 .$$

To obtain WSRT statistic the difference between each observation and the hypothesized mean M, $d_i = x_i - M = x_i$ were calculated (where x_i is the $i\ th$ observation). Secondly rank of the set $\{| d_i |;\ i = 1,\ \ldots,\ n\}$ and name them Y_i, $i = 1, \ldots,\ n$. Define $R_i = (signX_i)Y_i$. Then the Wilcoxon rank statistic W is defined to be the sum of all positive R_i s. It is well known that under the null hypothesis M=o, the mean and the standard deviation of W is given as:

$$\mu_w = n(n+1)/4 ,$$

$$\sigma_w = \sqrt{n(n+1)(2n+1)/24} .$$

where n is the sample size.

The null distribution of W when sample size is large ($n > 20$) can be approximated by a normal distribution. Therefore, the p-value is given by P(Z>|z|) where z= $(w - \mu_w)/\sigma_w$. Then, we can select the significant range of frequencies by the p-values. Since we have 182 frequencies, we have a multiple comparison case, Bonferroni method is used to calculate the adjusted probability of type I error. The

critical value should be $Z_{0.05/(2*182)}$ and the corresponding p-value is around 0.0002. Among 182 variables, the one whose p-value of WSRT is smaller than 0.0002 should be selected as significant one for the next step's analysis.

2.3 PLS for two and three dimensional differentiation

Since the numbers of predictors in our data are large (larger than the number of observations) and the predictors are highly correlated, PLS is used to find the discriminators, which is to distinguish two kinds of the cells on the selected significant variables. PLS, developed by Herman Wold at 1960's, is a popular effective method to build predictive models for variables with large number and high collinearity[5]. It extracts latent factors decomposed from both predictor X and response Y such that these latent factors can account for as much of covariance between X and Y as possible, and then use a certain number of these factors to build the predicted model. If denoted x_j as the *jth* significant variable selected from WSRT, the *jth* factors were extracted as

$$C = \sum_{j=1}^{n} \alpha_j x_j \quad , \qquad (\sum_{j=1}^{n} \alpha_j^2)^{1/2} = 1$$

such that

$$A = [corr(Y, C_i)]^2 \operatorname{var}(C_i)$$

is maximized, and all the factors are orthogonal. Specifically, the first factor will maximize A, the second factor will maximize A among all factors which are orthogonal to the first factors, etc.

The final regression model can be presented as following:

$$Y=\sum_{i=1}^{n} a_i x_i$$

where k is the number of factors used in the model which can be decided by various of criterion. In this study k is selected as the minimum number such that 95% of variance of both predicts and response variables can be explained.

In addition, response variables were coded as 0 and 1 to represent two kinds infect cells we are comparing. To do discrimination among Mock, HSV1, and Coxsackie, the mean of intra data of Mock- HSV1(m-h), intra data of Mock- Coxsackie (m-c), and inner data of Mock- Mock (m-m) were computed respectively. We searched for the frequencies where both intra differences are significant. Furthermore, all m-h values are larger than m-c values at these frequencies. Therefore, we can be coded HSV1 as 2, Coxsackie as 1 and Mock as 0. PLS method can be used to find the discrimination.

2.4 Evaluation of model diagnostic performance

2.4.1 AUC, sensitivity and specificity

Sensitivity and specificity are popular statistics used to evaluate the accuracy of a diagnostic or classification model with binary response. Sensitivity measures the proportion of correctly diagnostic positives among all true positives (TP), that is the percentage of objects who are identified as disease among all diseased objects. Specificity measures the proportion of correctly negatives among all true negatives (TN), that is the percentage of objects who are identified as non-disease among all non-diseased objects (Wall, 2001). To illustrate the definition of sensitivity and

specificity, we can construct following table:

Table- 1 2-dimensional diagnostic table

		Condition of disease	
		Positive	**Negative**
Test Result	**Positive**	True Positive(TP)	False Positive(FP)
	Negative	False Negative(FN)	True Negative(TN)

Then we have:

$$Sensitivity = \frac{Number \text{ of TP}}{\text{Number of TP} + \text{Number of FN}}$$

$$Specificity = \frac{Number \text{ of TN}}{\text{Number of TN} + \text{Number of FP}}$$

Generally, sensitivity and specificity are determined by the cutoff point for the discriminators. Unfortunately, the cutoff point that yields high sensitivity may produce low specificity, and vise versa. Therefore, we use a few specificities at fixed sensitivities to evaluate the discriminators.

Another popular statistic to gauge the goodness of the discriminators is Area under the ROC (Receiver Operating Characteristic) Curve (AUC)[9]. It presents the area under the graph of the sensitivity vs 1-specificity by varying the cut-off points for the discriminators. It is obvious that larger AUC presents better discrimination. Many researchers use the following guide for evaluating the accuracy of a diagnostic test:

Table- 2 rough criterion for accuracy evaluation

AUC	Performance of Discrimination
.90-1	Excellent
.80-.90	Good
.70-.80	Fair
.60-.70	Poor
.50-.60	Fail

Let X_1 be the random variable of the discriminators which is from disease population and X_2 be the one from benign population. Then for a cutoff point C we have:

$$Sensitivity = P(X_1 \leq C)$$

$$Specificity = P(X_2 \geq C)$$

Furthermore, DeLong (1988) Bamber (1957) showed:

$$AUC = P(X_1 \leq X_2)$$

X_1, X_2 are normally distributed with means μ_1, μ_2 and standard deviations σ_1, σ_2, respectively. then we have:

$$\frac{X_1 - \mu_1}{\sigma_1} \sim N(0,1)$$

$$\frac{X_2 - \mu_2}{\sigma_2} \sim N(0,1)$$

At $sensitivity = 1 - \alpha$, we have:

$$Sensitivity = P(X_1 \le C\ ) = P(\frac{X_1 - \mu_1}{\sigma_1} \le \frac{C\ - \mu_1}{\sigma_1}) = \phi(\frac{C\ - \mu_1}{\sigma_1}) = 1 - \alpha$$

This implies:

$$C\ = \phi^{-1}(1-\alpha)\sigma_1 + \mu_1,$$

Therefore,

$$Specificifity = P(X_2 \ge C\ ) = P(\frac{X_2 - \mu_2}{\sigma_2} \ge \frac{C\ - \mu_2}{\sigma_2}) = \phi(-\frac{C\ - \mu_2}{\sigma_2}) = \frac{\mu_2 - \mu_1 - \sigma_1\phi^{-1}(1-\alpha)}{\sigma_2}$$

and

$$AUC = P(X_1 \le X_2) = P(\frac{(X_1 - X_2) - (\mu_1 - \mu_2)}{\sqrt{\sigma_1^2 + \sigma_2^2}} \le \frac{\mu_2 - \mu_1}{\sqrt{\sigma_1^2 + \sigma_2^2}}) = \phi(\frac{\mu_2 - \mu_1}{\sqrt{\sigma_1^2 + \sigma_2^2}})$$

Therefore, to calculate the specificity corresponding to 80%, 90% and 95% sensitivity, we just need to replace α with .2, .1, and .05 in the above formulas. Since both data sets in the pairwise comparisons without Mock are from diseased population, the specificities will be meaningless. Therefore, we just calculate the AUC to do the evaluation without specificities.

2.4.2 Volume Under the Surface

To evaluate the diagnostic performance of simultaneous discrimination of Mock, HSV1 and Coxsackie, Volume Under the Surface (VUS) which is an extension of the AUC for multi-group discrimination is employed in the study. There are a few ways to generalize ROC curve to a k-dimensional surface. One way was proposed by Hsu and was studied in Cha's thesis [6] and Li's dissertation [7]. Denote data of Mock minus Mock as X_1, Mock minus HSV1 as X_2 and Mock minus Coxsackie as X_3, and a_1, a_2, a_3 be the probability in Mock, Coxsackie and HSV1 respectively, for $a_1 + a_2 + a_3 = 1$ we have the estimated VUS according to Yi Li(2009) [7]as:

$$\hat{VUS} = \frac{1}{n_1 n_2 n_3} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} \left(\frac{a_2}{a_2 + a_3} I(x_{2j} \le x_{3k}) + \frac{a_3}{2(a_2 + a_3)}\right)\left(\frac{a_2}{a_1 + a_2} I(x_{1i} \le x_{2j}) + \frac{a_1}{2(a_1 + a_2)}\right) I(x_{1i} \le x_{3k})$$

Furthermore, Li showed:

$$VUS_{\max} = \frac{(a_1/2 + a_2)(a_2 + a_3/2)}{(a_1 + a_2)(a_2 + a_3)},$$

and

$$VUS_{\min} = \frac{a_1 a_3/8 + a_2^2/6 + a_1 a_2/6 + a_2 a_3/6}{(a_1 + a_2)(a_2 + a_3)}$$

Since $VUS_{\max} \neq 1$ and $VUS_{\min} \neq 0.5$, it is hard to interpret the value of VUS. Li proposed the mapping methods to transform VUS into the interval (0.5, 1). The first mapping is linear mapping. The formulas are shown as bellow:

$$\frac{VUS - VUS_{\min}}{VUS_{\max} - VUS_{\min}} = \frac{VUS_{new} - .5}{1 - .5}$$

that is

$$VUS_{new} = .5 \frac{VUS - VUS_{\min}}{VUS_{\max} - VUS_{\min}} + .5$$

The second one is quadratic mapping. The formulas are:

$$\frac{\sqrt{VUS} - \sqrt{VUS_{\min}}}{\sqrt{VUS_{\max}} - \sqrt{VUS_{\min}}} = \frac{VUS_{new} - .5}{1 - .5}$$

So

$$VUS_{new} = .5 \frac{\sqrt{VUS} - \sqrt{VUS_{\min}}}{\sqrt{VUS_{\max}} - \sqrt{VUS_{\min}}} + .5$$

The reason for quadratic mapping is that the dimension is 4 for discriminating 3 classes in comparing with dimension of ROC being 2 (see[7]).

2.5 Bootstrap method and 2-fold cross validation method

Although we have used the measurement talked above to evaluate the accuracy

of our procedure, they need to be validated. The validation involves the estimation of the shrinkages of the evaluation statistics discussed in section 2.4. In this study, we adopt Bootstrap method and 2-fold Cross validation to estimate the shrinkages.

Bootstrap was proposed by Efron in 1979 [8]. The original article was using this numerical technique to estimate the bias of an estimator. Since the shrinkage can be considered as a bias, we certainly can use it as a method to estimate the shrinkages. To do this, we draw a bootstrap sample and use the discussed discrimination method in section 2.3 to obtain a discriminator, then validate it in the original sample. From this procedure, we obtain one group of accuracy measurement from training and validation data, respectively. The difference between result of accuracy measurement of training data and that of validation data is an estimate of the shrinkage. Repeat the procedure for 100 times and take the average of the 100 shrinkages which is final estimate of the shrinkages.

For k-fold cross-validation, we select k=2. This is because we do not have a large sample. The validation set need to be large enough to find sensitivity, specificity etc. The process of 2-fold cross-validation is similar to the Bootstrap validation. We randomly divide the original data into two equal size groups, one of which is used as training data and the other one is treated as validation data. The average of 100 shrinkages is obtained to do the evaluation.

In general, the shrinkage is a decreasing function of the sample size. The 2-fold cross-validation method used half of the original sample size. Therefore, we expect it over-estimates the shrinkages. However, Bootstrap sample in the average contains

around 62% of the original sample. The 62% of similarity with the original sample (or validation sample) will under-estimate the shrinkages. We believe the true shrinkages should be between these two estimators.

CHAPTER 3 RESULTS AND INTERPRETATION

3.1 Result for comparison of Mock vs HSV1

As mentioned before, we have 21 groups of data for Mock and HSV1, respectively. So we obtain 42 groups of Inner data which are shown as blue, and 42 groups of Intra data shown as red in the Figure-1. As in Figure-1, Inner and Intra difference overlay with each other for most of the wavenumbers. However, there are certain ranges over which Inner and Intra data separated. Figure-2 is the graph of P-value obtained from WSRT. To differentiate Mock from HSV1 infected cells, those frequencies of which P-values are less or equal than 0.0002 are selected as significant variables to do PLSR. The P-value is determined through the Bonferroni's multiple test. There are 72 selected frequencies whose locations are shown in Figure-3. From Figure-3 we can see that the variables we chose are located at the wavenumbers where the two kinds of curves are somewhat separated where we can visually see the significant difference. Using 95% variation guideline on numbers of PLS Factors, we select the first 10 factors to obtain the discriminators. Table-3 shows us the coefficient of each variable. Figure-4 is the box-plot of predicted values from PLSR. Since there is no overlay between Inner and Intra data, we can see the procedure gives excellent discrimination of Mock and HSV1.

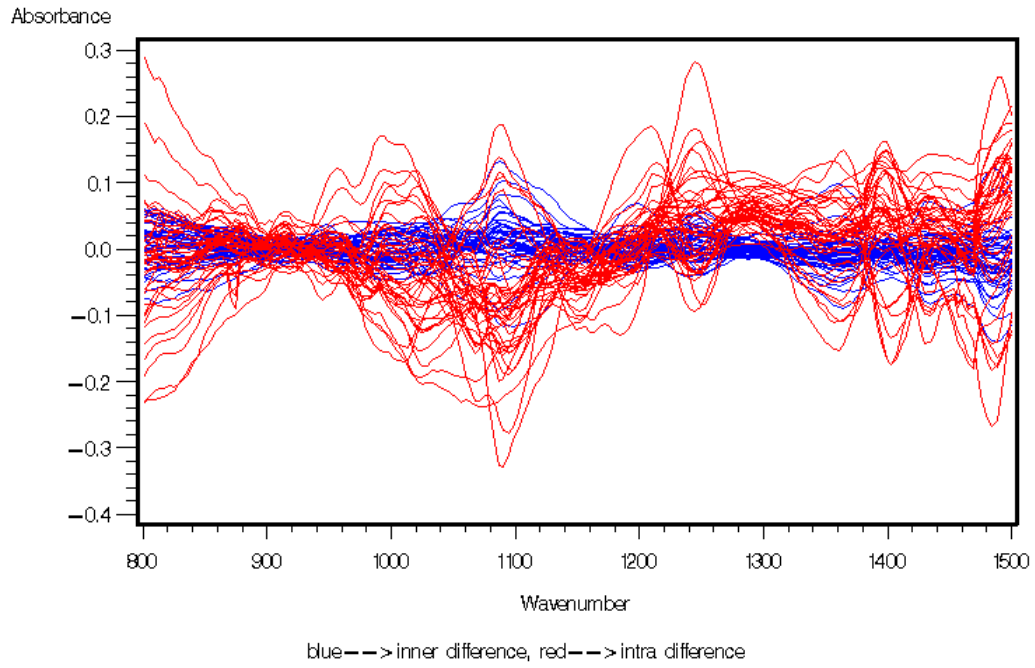


Figure- 1 Inner & Intra data for Mock vs HSV1 Infected Cells

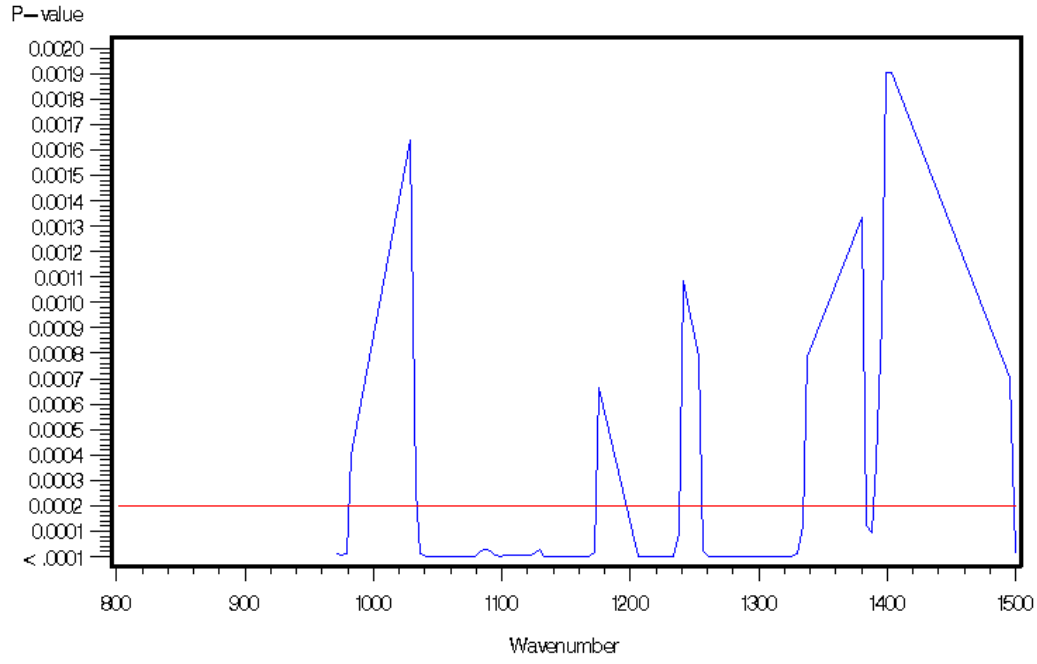


Figure- 2 P-values obtained from WSRT

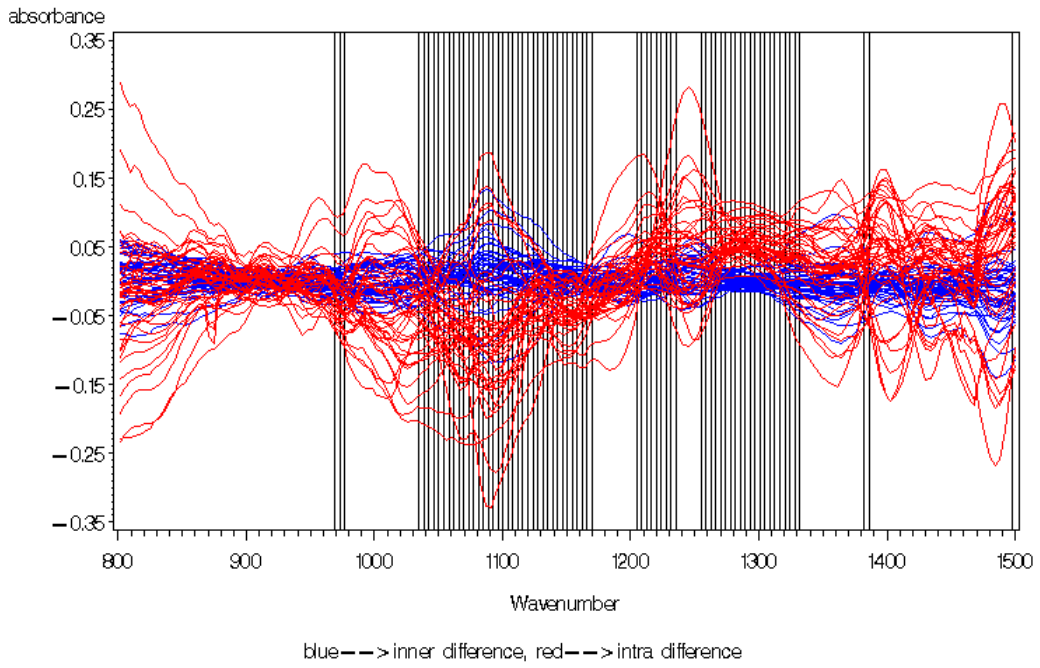


Figure- 3 Location of selected variables for Mock & HSV1 Infected Cells

Table- 3 Percent Variation Accounted for by PLS Factors

Number of Extracted variables	Independent Variables		Dependent Variables	
	Current	Total	Current	Total
1	56.9113	56.9113	65.4694	65.4694
2	21.0183	77.9296	6.7177	72.1872
3	6.9845	84.9142	5.3509	77.5380
4	3.9435	88.8576	6.2306	83.7687
5	6.1763	95.0339	3.0464	86.8151
6	2.1166	97.1505	1.1429	87.9580
7	0.5531	97.7036	1.9126	89.8706
8	0.9969	98.7005	1.0629	90.9335
9	0.3922	99.0926	2.6901	93.6236
10	0.3017	99.3943	1.2417	94.8653

Table- 4 Coefficients of significant variables for Mock vs HSV1 Infected Cells

variable	coefficient	variable	coefficient	variable	coefficient	variable	coefficient
969.2278	2.40571	1092.672	-0.80202	1162.109	4.141104	1277.838	2.573121
973.0854	-1.7279	1096.529	-0.46591	1165.967	-7.80916	1281.695	4.011748
976.943	0.851569	1100.387	0.402074	1169.824	-10.9246	1285.553	4.117925
1034.807	-4.31488	1104.245	1.118464	1204.543	-4.50405	1289.41	5.401583
1038.665	-4.52704	1108.102	1.363855	1208.4	-3.45379	1293.268	6.135614
1042.523	-3.27098	1111.96	-0.25496	1212.258	3.796854	1297.126	4.812678
1046.38	0.357261	1115.817	-1.03622	1216.116	4.615034	1300.983	1.108285
1050.238	2.570739	1119.675	-1.59614	1219.973	4.173997	1304.841	-0.88234
1054.095	4.896635	1123.533	-2.96474	1223.831	0.256281	1308.699	-3.21023
1057.953	4.399465	1127.39	-0.142	1227.688	-0.8532	1312.556	-4.19975
1061.811	1.632553	1131.248	2.96959	1231.546	-1.78863	1316.414	-1.86935
1065.668	0.014091	1135.106	4.038975	1235.404	-0.8378	1320.271	-1.06322
1069.526	-1.64993	1138.963	0.348473	1254.692	-3.67702	1324.129	-0.50471
1073.384	-1.76568	1142.821	2.256314	1258.549	-2.93105	1327.987	-1.66381
1077.241	-1.29508	1146.678	1.299303	1262.407	-1.68277	1331.844	-3.17146
1081.099	-0.29131	1150.536	3.815472	1266.265	-1.07098	1381.993	-2.76318
1084.956	-0.30766	1154.394	1.563418	1270.122	0.772767	1385.851	2.599533
1088.814	-0.28679	1158.251	-0.2684	1273.98	1.304376	1497.722	0.919644

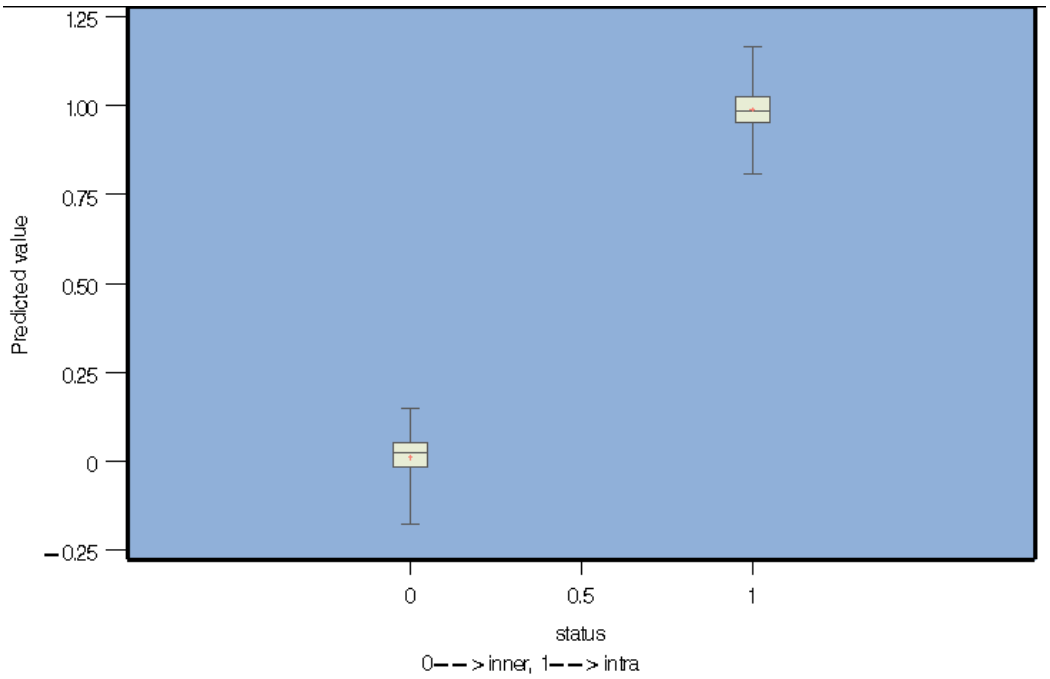


Figure- 4 Box-plot of predicted values of Mock and HSV1 Infected Cells

Table-5 includes the results of original evaluation and their shrinkages from Bootstrap (BT) method and 2-fold Cross-validation (CV). As shown in the Table-5, the original value of AUC and specificities are all equal to 1 which represents excellent discrimination of Mock and HSV1. Moreover, the shrinkages obtained by Cross-validation are very small (<10%), as well as those from Bootstrap method which is even smaller (<0.2%). As we mentioned before, the Cross-validation overestimate the shrinkage; whereas the Bootstrap method underestimate the shrinkage. We calculate the average of these two estimates to show perhaps more reasonable estimate. Therefore, we can conclude that the shrinkages of our procedure are very small, too. The small shrinkages yield to very high value of final AUC and specificities which are all larger than 95%. Overall, our procedure can

excellently distinguish Mock and HSV1 infected cells.

Table- 5 Comparison of Mock vs HSV1 Infected CElls

Measureme -nt	Original estimate	BT shrinkage	CV shrinkage	BT final	CV final	Average
AUC	1	0.0015669	0.0157860	0.9984331	0.9842140	0.9913236
Sp (sen=95%)	1	0.0004556	0.0882122	0.9995444	0.9117878	0.9556661
Sp (sen=90%)	1	0.0000142	0.0193804	0.9999858	0.9806196	0.9903026
Sp (sen=80%)	1	0.0000001	0.0012577	0.9999999	0.9987423	0.9993711

3.2 Result for comparison of Mock vs Coxsackie

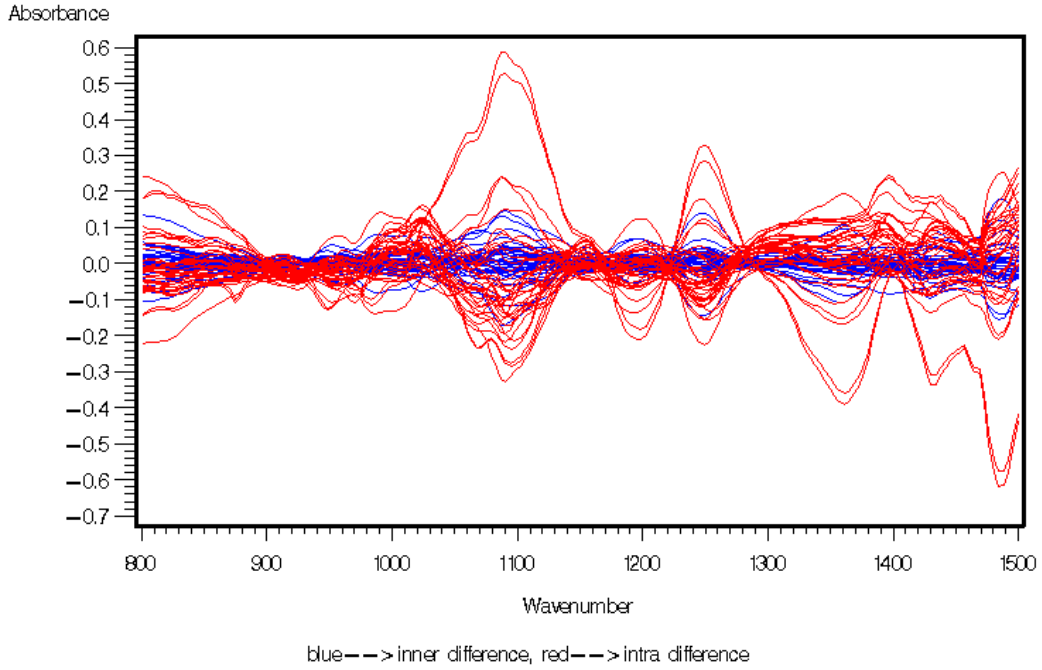


Figure- 5 Inner & Intra data for Mock vs *Coxsackie* Infected Cells

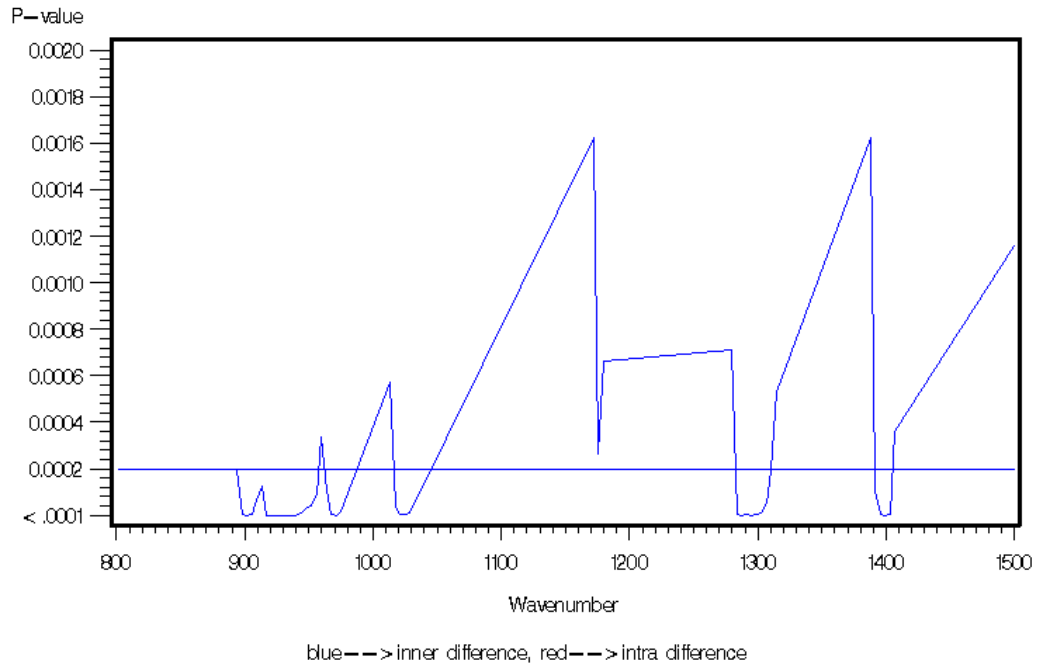


Figure- 6 P-values obtained from WSRT for Mock vs Coxsackie Infected Cells

Table- 6 Percent Variation Accounted by PLS Factors for Mock vs Coxsackie Infected Cells

Number of Extracted Factors	Moel Effects		Dependent Variables	
	Current	Total	Current	Total
1	45.2461	45.2461	57.4882	57.4882
2	20.2041	65.4503	4.4442	61.9323
3	9.8070	75.2572	6.3680	68.3003
4	15.4898	90.7470	1.2849	69.5852
5	2.4711	93.2181	4.4070	73.9922
6	3.2884	96.5065	3.0521	77.0444
7	2.6453	99.1518	2.8007	79.8450
8	0.2937	99.4456	5.4630	85.3080
9	0.2878	99.7333	2.9562	88.2642
10	0.0908	99.8242	2.8545	91.1187
11	0.0550	99.8791	3.3061	94.4248
12	0.0580	99.9372	0.7073	95.1320
13	0.0078	99.9450	2.1014	97.2335
14	0.0267	99.9717	0.2453	97.4788
15	0.0049	99.9766	0.3923	97.8710

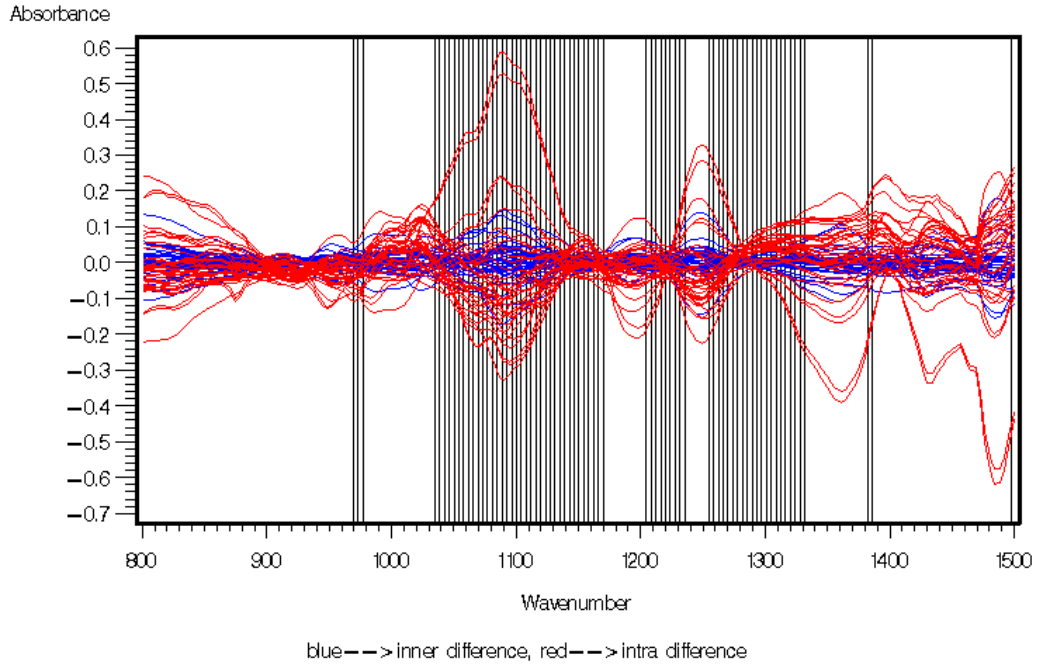


Figure- 7 Location of selected variables for Mock & Coxsackie Infected Cells

Table- 7 Coefficients of significant variables for Mock vs Coxsackie Infected CEIls

Variable	coefficient	variable	coefficient	variable	coefficient	variable	coefficient
894.0041	16.10422	932.5803	-14.2163	975.0142	-1.62225	1302.912	-3.03676
897.8617	2.318229	936.438	14.92157	1017.448	-0.73084	1306.77	-4.67095
901.7194	-7.43914	940.2956	6.164857	1021.306	1.900459	1310.627	-4.83333
905.577	-14.9607	944.1532	4.91767	1025.163	2.918608	1391.637	-2.61931
909.4346	7.106428	948.0108	4.439454	1029.021	-0.47518	1395.495	0.796733
913.2922	21.62438	951.8685	0.360445	1283.624	1.859231	1399.353	2.659035
917.1499	6.916377	955.7261	1.01493	1287.482	2.964753	1403.21	1.619077
921.0075	-6.15804	963.4413	4.643868	1291.339	8.385121		
924.8651	-16.3421	967.2989	-0.64618	1295.197	3.641702		
928.7227	-20.3733	971.1566	-4.28043	1299.055	-0.61215		

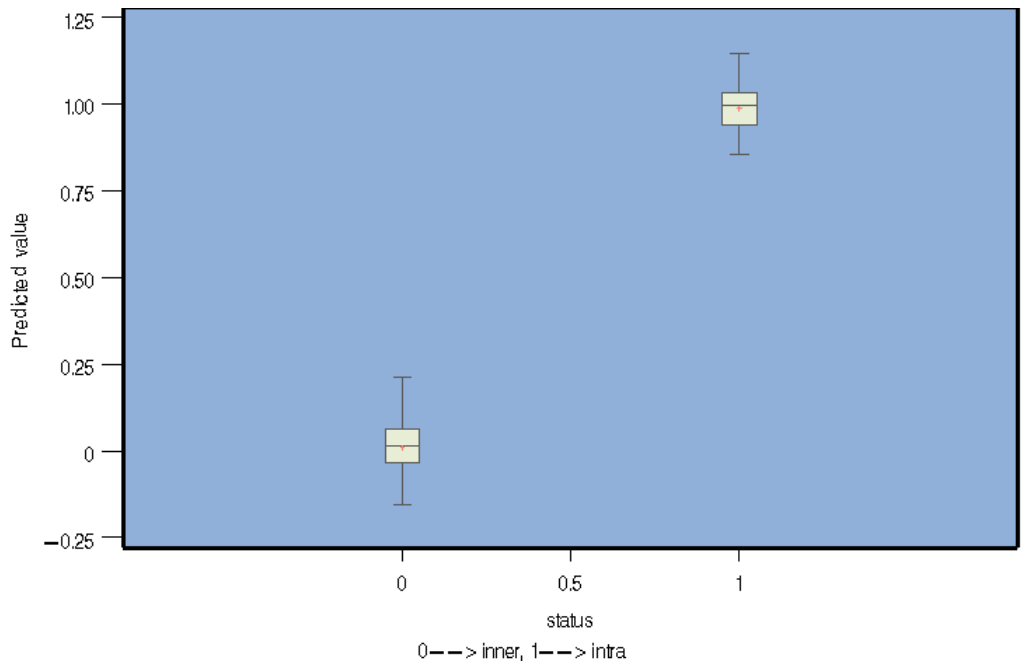


Figure- 8 Box-plot of predicted values of Mock and Coxsackie Infected Cells

Table- 8 Comparison of Mock vs Coxsackie Infected Cells

Measurements	Original estimate	BT shrinkage	CV shrinkage	BT final	CV final	Average
AUC	1	0.0057682	0.0557993	0.9942318	0.9442007	0.9692163
Sp (sen=95%)	1	0.0162822	0.3605640	0.9837178	0.6394360	0.8115769
Sp (sen=90%)	1	0.0021444	0.1736157	0.9978556	0.8263843	0.9121200
Sp (sen=80%)	1	0.0001343	0.0423241	0.9998657	0.9576759	0.9787708

3.3 Result for comparison of HSV1 vs Coxsackie Infected Cells

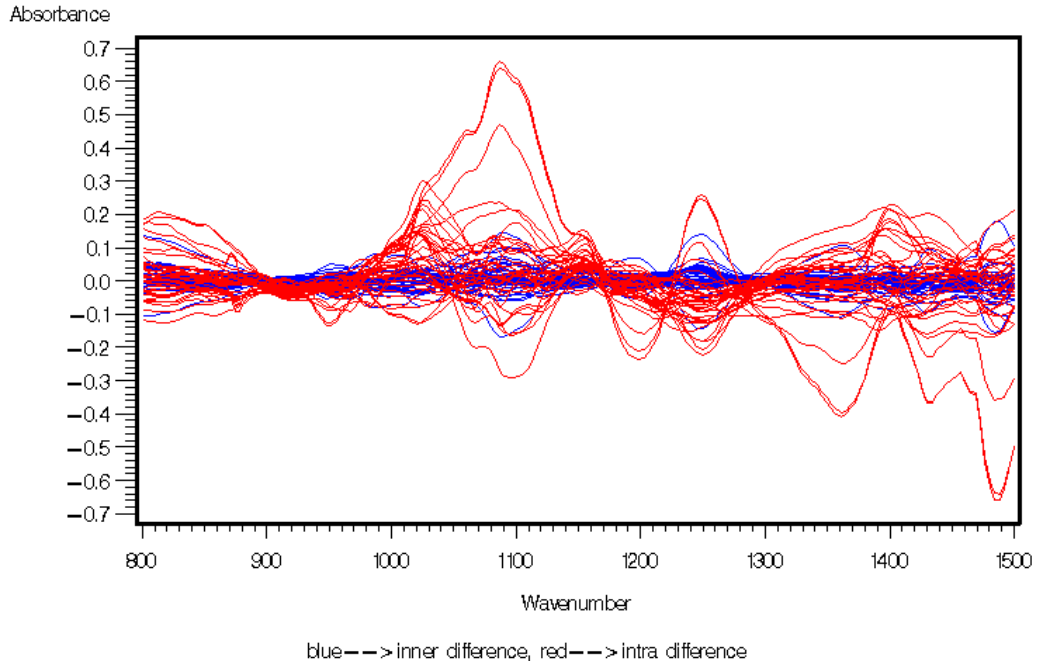


Figure- 9 Inner & Intra data for HSV1 vs Coxsackie

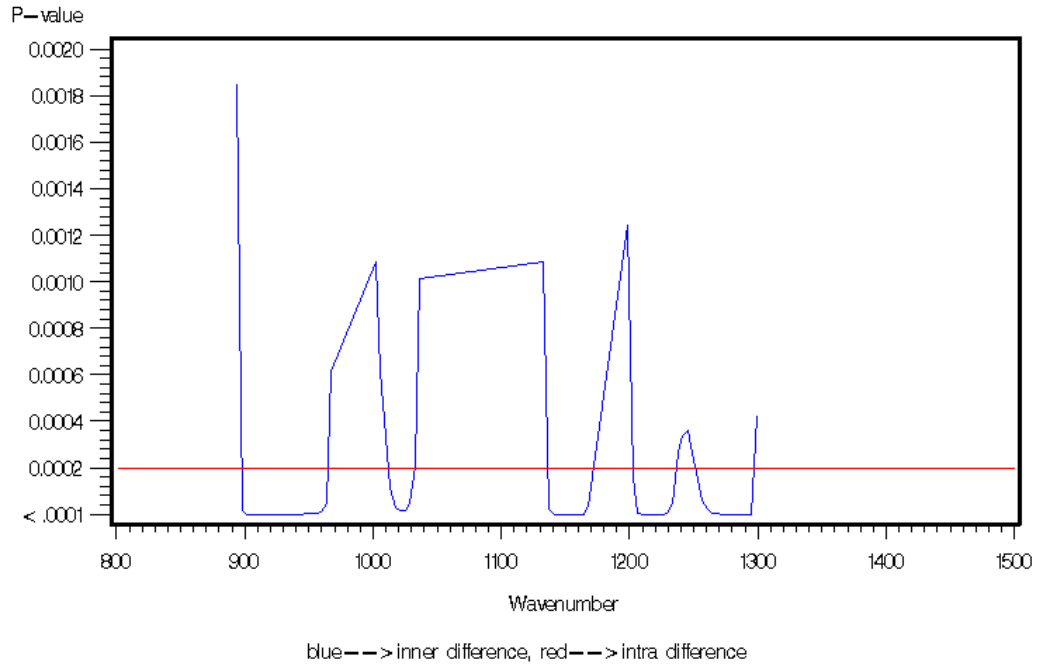


Figure- 10 P-values obtained from WSRT for HSV1 vs Coxsackie Cells

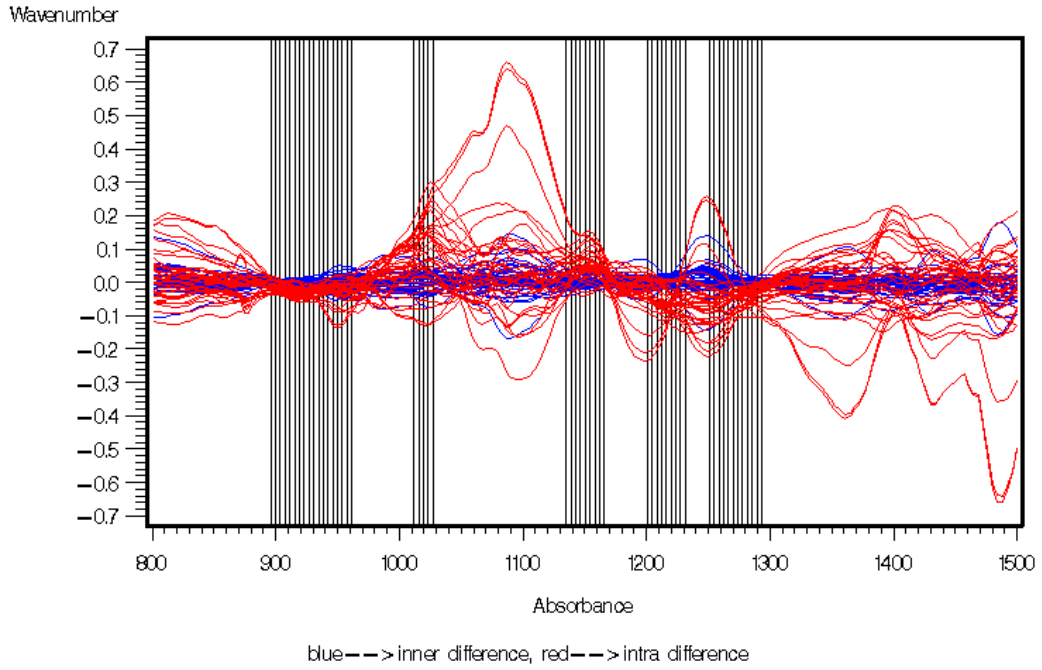


Figure- 11 Location of selected variables for Mock vs Coxsackie Infected Cells

Table- 9 Percent Variation Accounted by PLS Factors for HSV1 vs Coxsackie Infected Cells

Number of Extracted factors	Model Effects		Dependent Variables	
	Current	Total	Current	Total
1	44.9579	44.9579	75.4549	75.4549
2	19.4883	64.4462	4.7542	80.2091
3	12.4480	76.8942	3.0451	83.2542
4	7.5435	84.4376	2.2286	85.4828
5	1.6714	86.1091	7.8812	93.3640
6	9.8054	95.9145	0.6627	94.0267
7	1.5915	97.5060	0.8004	94.8270
8	1.6350	99.1411	0.3981	95.2252
9	0.1942	99.3353	1.0984	96.3236
10	0.3807	99.7160	0.2598	96.5834

Table- 10 Coefficients of significant variables for HSV1 vs Coxsackie infected cells

variable	coefficient	variable	coefficient	Variable	Coefficient	variable	coefficient
895.9329	10.5515	949.9396	1.383823	1154.394	0.829049	1254.692	-0.13805
899.7905	7.352137	953.7973	2.180964	1158.251	1.444732	1258.549	-0.29153
903.6482	2.073188	957.6549	3.396522	1162.109	2.564662	1262.407	-0.40727
907.5058	-1.10856	961.5125	4.650166	1165.967	3.313081	1266.265	-0.41144
911.3634	-3.27855	1011.662	0.358282	1200.685	-1.52828	1270.122	-0.58546
915.221	-4.74862	1015.519	0.368065	1204.543	-1.7593	1273.98	-0.66289
919.0787	-4.71022	1019.377	0.323084	1208.4	-1.8836	1277.838	-0.48853
922.9363	-5.3993	1023.234	0.239398	1212.258	-2.04895	1281.695	-0.03246
926.7939	-5.37332	1027.092	0.127122	1216.116	-1.66405	1285.553	0.65095
930.6515	-4.09006	1135.106	-1.38547	1219.973	-0.77283	1289.41	1.46524
934.5092	-3.02199	1138.963	-1.32705	1223.831	0.452887	1293.268	2.124589
938.3668	-1.67272	1142.821	-0.82344	1227.688	1.059691		
942.2244	-0.0034	1146.678	-0.09856	1231.546	1.00258		
946.082	0.672056	1150.536	0.355427	1250.834	0.052065		

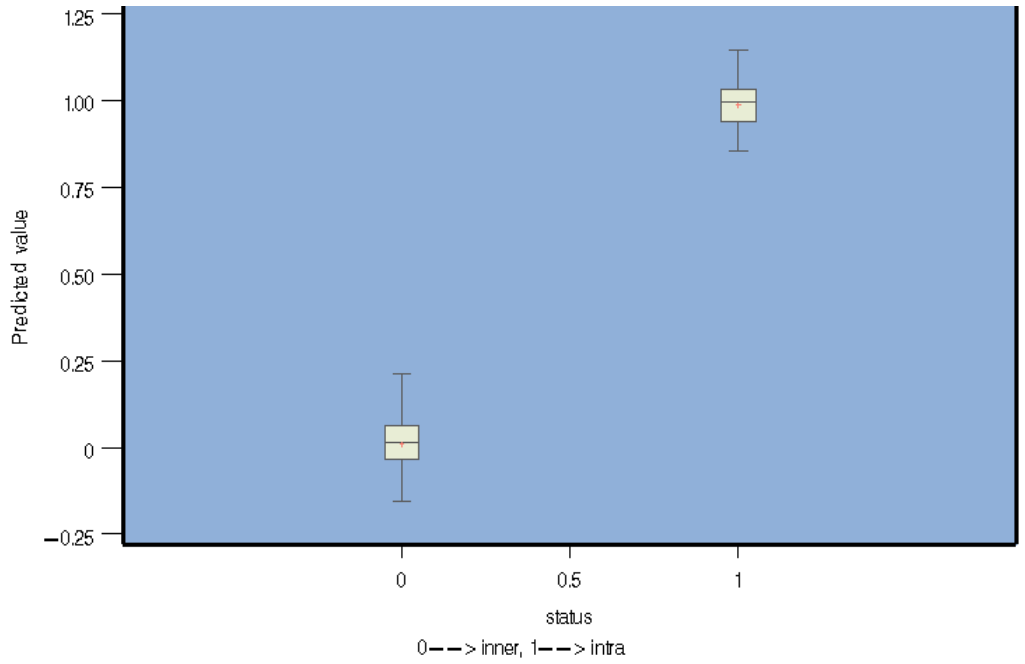


Figure- 12 Box-plot of predicted values of HSV1 vs Coxsakie infected cells

Table- 11 Comparison of HSV1 vs Coxsackie infected cells

Measuremen -t	Original estimate	BT shrinkage	CV shrinkage	BT Final	CV final	Average
AUC	1	0.00090796	0.03078348	0.9990920	0.9692165	0.984154

3.4 Result for comparison of HSV1 vs Adeno infected cells

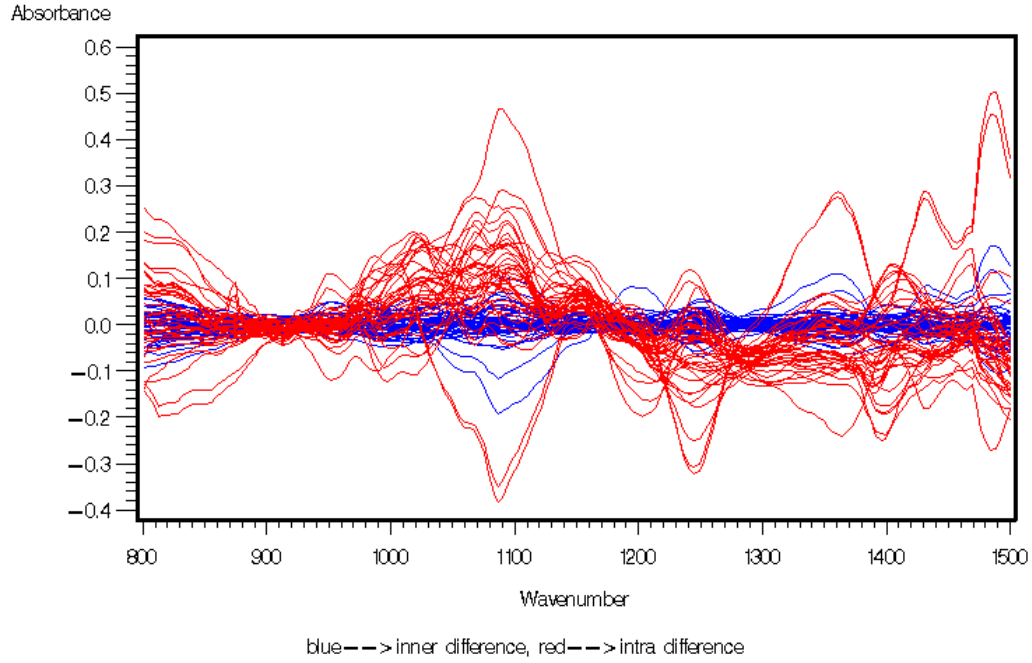


Figure- 13 Inner & Intra data for HSV1 vs Adeno infected cells

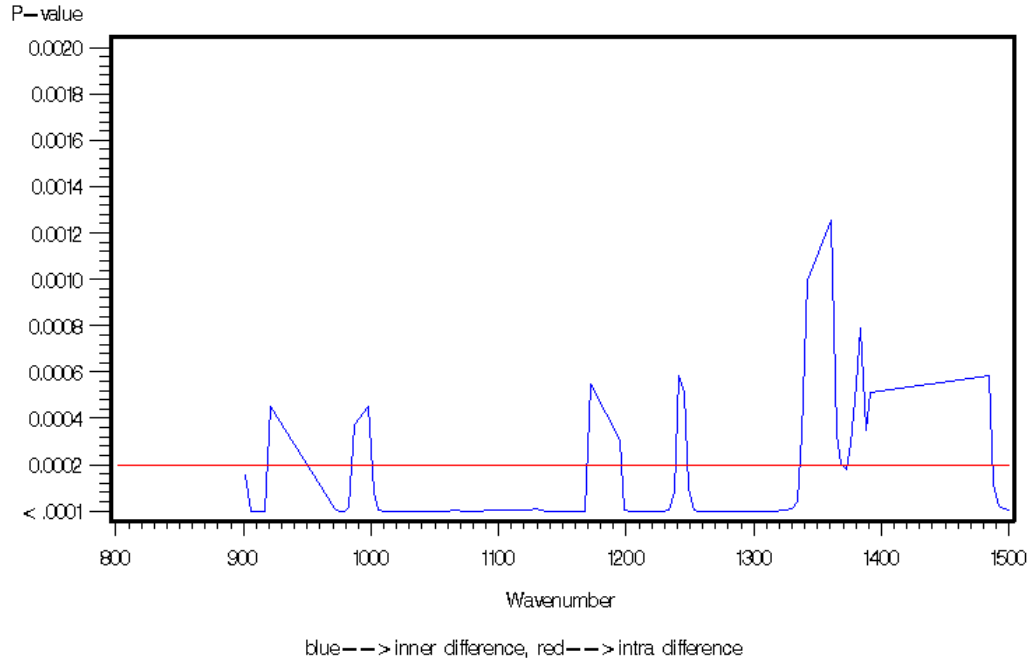


Figure- 14 P-values obtained from WSRT for HSV1 vs Adeno cells

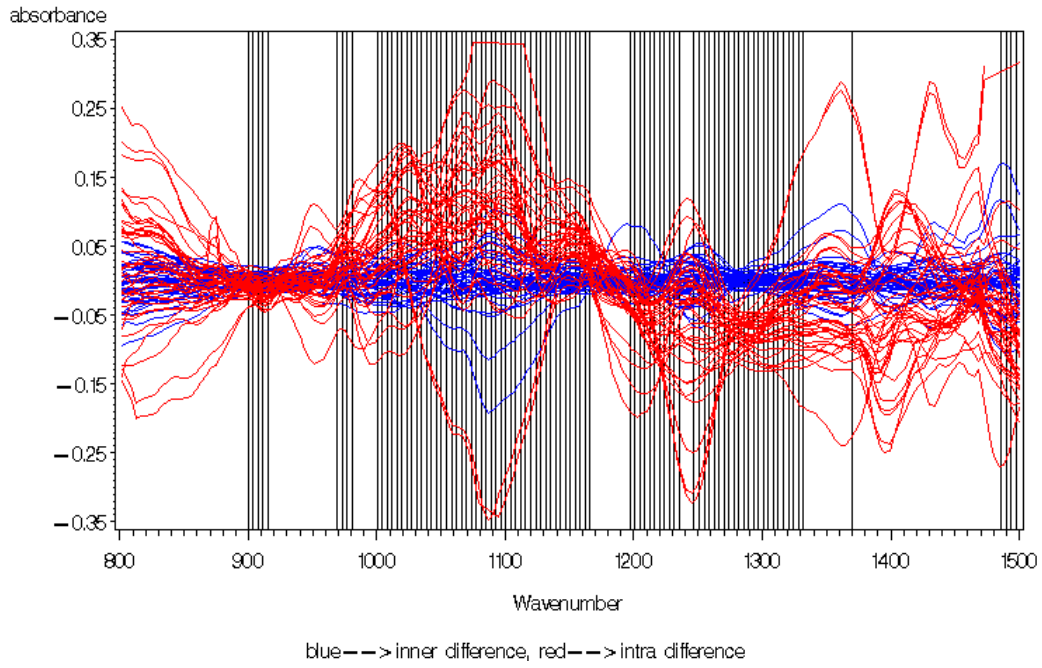


Figure- 15 Location of selected variables for HSV1 vs Adeno infected cells

Table- 12 Percent Variation Accounted by PLS Factors for HSV1 vs Adeno infected cells

Number of extracted Factors	Model Effects		Dependent Variables	
	Current	Total	Current	Total
1	48.9946	48.9946	71.1989	71.1989
2	26.4531	75.4477	12.3020	83.5010
3	10.6421	86.0898	6.6828	90.1837
4	4.2682	90.3579	2.4808	92.6645
5	3.2101	93.5680	1.5116	94.1762
6	1.7331	95.3011	0.5724	94.7485
7	1.1659	96.4670	0.5092	95.2577
8	2.2227	98.6898	0.1040	95.3617
9	0.4052	99.0950	0.2562	95.6179
10	0.4810	99.5760	0.1614	95.7793

Table- 13 Coefficients of significant variables for HSV1 vs Adeno infected cells

variable	coefficient	variable	coefficient	variable	coefficient	variable	coefficient
899.7905	0.894496	1054.095	0.078171	1142.821	-0.1437	1266.265	-0.62877
903.6482	-0.681	1057.953	0.042874	1146.678	0.139795	1270.122	-0.75658
907.5058	-2.07676	1061.811	0.036097	1150.536	0.389834	1273.98	-0.83068
911.3634	-2.9635	1065.668	0.028762	1154.394	0.673103	1277.838	-0.85471
915.221	-3.15268	1069.526	0.032307	1158.251	0.966435	1281.695	-0.87637
969.2278	-0.32206	1073.384	0.058846	1162.109	1.164495	1285.553	-0.87085
973.0854	-0.24103	1077.241	0.06661	1165.967	1.285578	1289.41	-0.8082
976.943	-0.46555	1081.099	0.066696	1196.828	-0.10511	1293.268	-0.73246
980.8006	-0.61484	1084.956	0.070272	1200.685	-0.20277	1297.126	-0.59998
1000.089	-0.29246	1088.814	0.084217	1204.543	-0.31724	1300.983	-0.40394
1003.946	-0.18273	1092.672	0.086161	1208.4	-0.30543	1304.841	-0.17862
1007.804	-0.13647	1096.529	0.057802	1212.258	-0.42495	1308.699	0.041111
1011.662	-0.08581	1100.387	0.009833	1216.116	-0.41842	1312.556	0.22759
1015.519	-0.01117	1104.245	-0.02932	1219.973	-0.35313	1316.414	0.310752
1019.377	0.035466	1108.102	-0.08506	1223.831	-0.27331	1320.271	0.322096
1023.234	0.043286	1111.96	-0.14726	1227.688	-0.17738	1324.129	0.259901
1027.092	0.041343	1115.817	-0.22914	1231.546	-0.13375	1327.987	0.210695
1030.95	-0.01163	1119.675	-0.31719	1235.404	-0.10381	1331.844	0.194287
1034.807	-0.03298	1123.533	-0.41987	1246.977	-0.13914	1370.421	0.245327
1038.665	0.012979	1127.39	-0.56325	1250.834	-0.18358	1486.149	-0.13627
1042.523	0.047211	1131.248	-0.67549	1254.692	-0.24854	1490.007	-0.20273
1046.38	0.097071	1135.106	-0.68389	1258.549	-0.34912	1493.864	-0.30124
1050.238	0.103459	1138.963	-0.45368	1262.407	-0.49132	1497.722	-0.39644

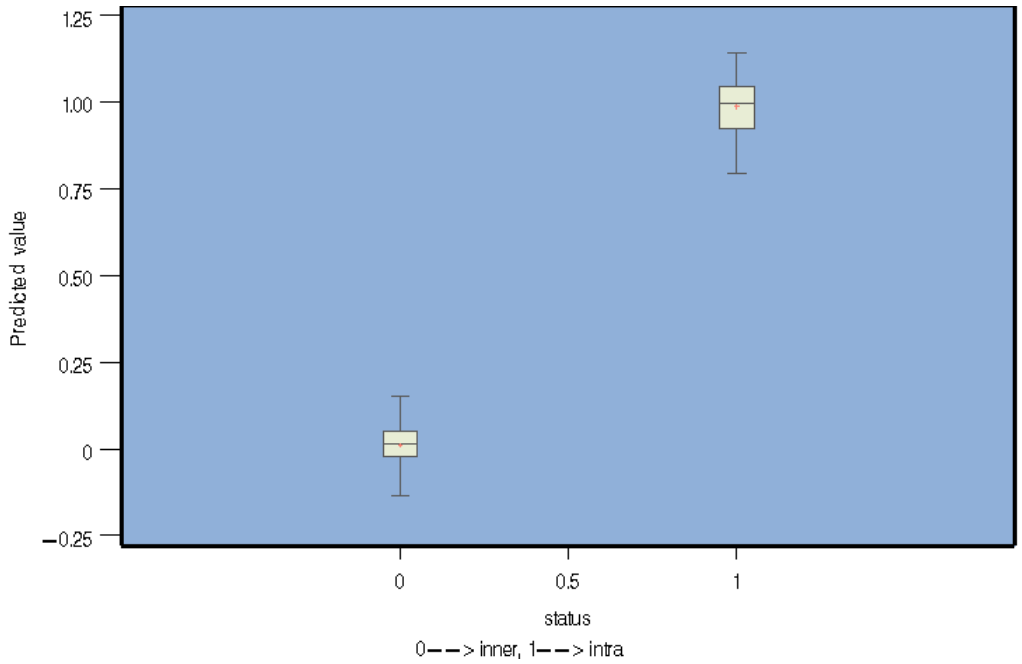


Figure- 16Box-plot of predicted values of HSV1 vs Adeno infected cells

Table- 14 Comparison of HSV1 vs Adeno infected cells

Measurement	Original estimate	BT shrinkage	CV shrinkage	BT final	CV final	Average
AUC	0.999999999	0.0000289	0.02082459	0.9999711	0.97917541	0.98957326

3.5 Result for comparison of Adeno vs Coxsackie cells

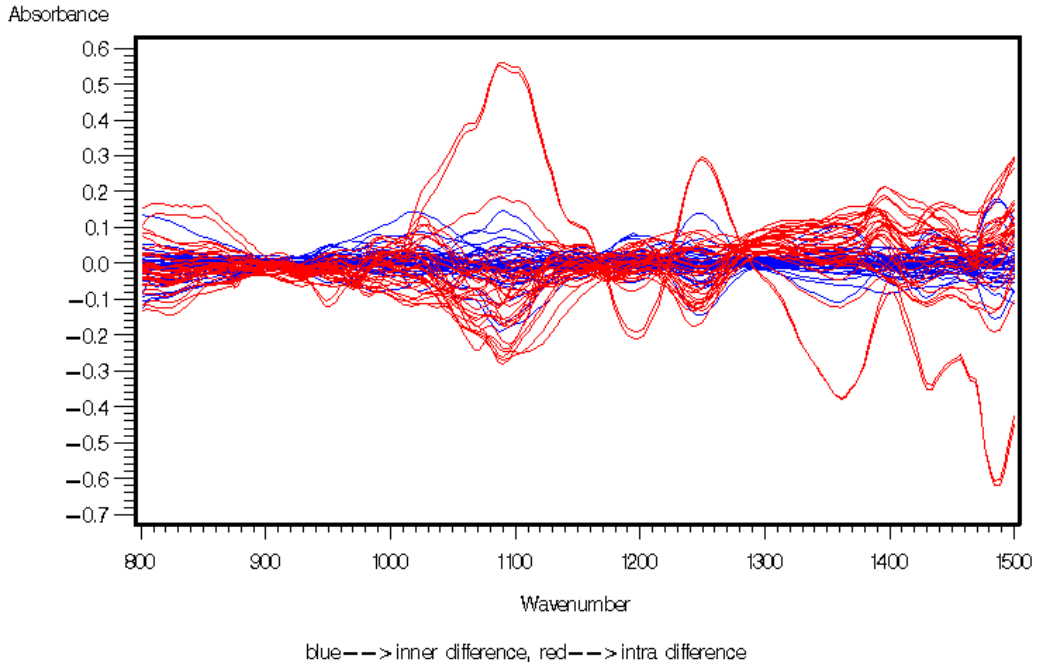


Figure- 17 Inner & Intra data for Adeno vs Coxsackie infected cells

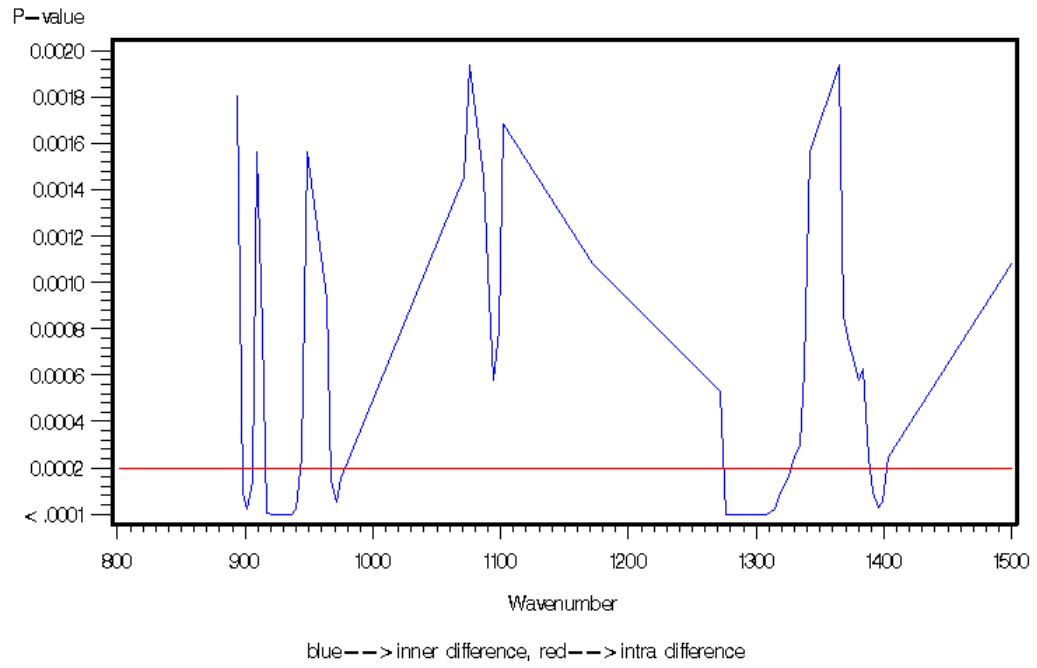


Figure- 18 P-values obtained from WSRT for Adeno vs Coxsackie infected cells

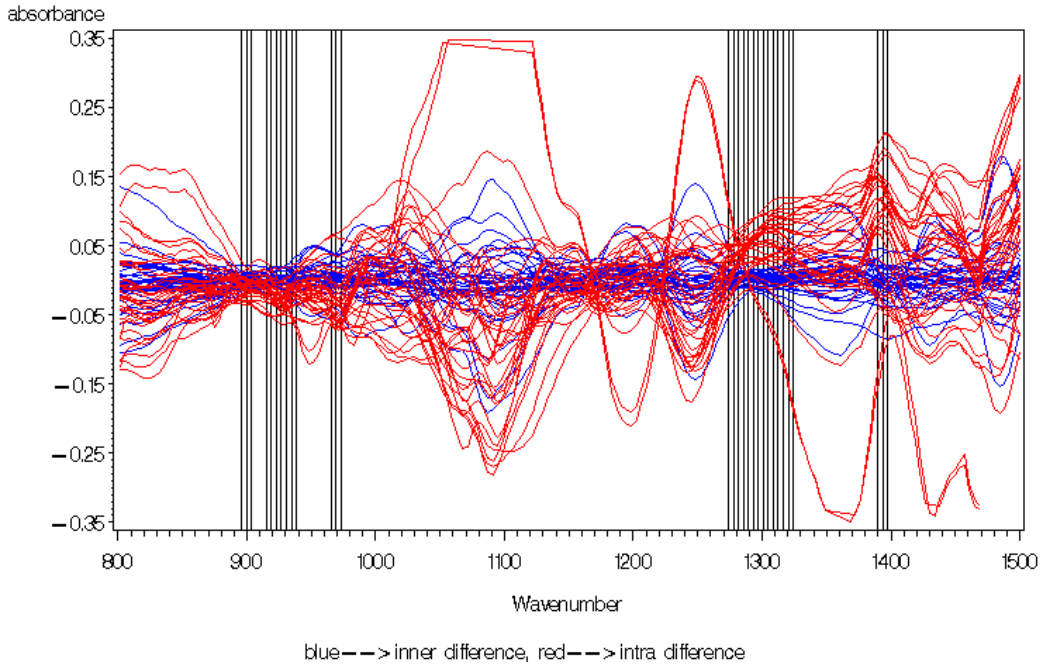


Figure- 19 Location of selected variables for Adeno vs Coxsackie infected cells

Table- 15 Percent Variation Accounted by PLS Factors for Adeno vs Coxsackie infected cells

Number of Extracted Factors	Model Effects		Dependent Effects	
	Current	Total	Current	Total
1	58.4046	58.4046	53.1031	53.1031
2	17.9649	76.3695	6.1215	59.2246
3	7.5538	83.9233	6.3637	65.5882
4	2.3291	86.2524	9.4914	75.0797
5	12.3827	98.6351	0.7889	75.8686
6	0.3167	98.9518	9.6675	85.5361
7	0.4911	99.4428	0.9185	86.4546
8	0.1592	99.6021	1.3677	87.8223
9	0.2719	99.8740	0.6125	88.4348
10	0.0322	99.9062	3.8530	92.2878
11	0.0269	99.9331	1.3191	93.6068
12	0.0242	99.9572	0.4396	94.0464
13	0.0155	99.9727	0.2752	94.3216
14	0.0102	99.9829	0.1754	94.4969
15	0.0043	99.9871	0.4566	94.9536

Table- 16 Coefficients of significant variables for Adeno vs Coxsackie infected cells

variable	coefficient	variable	coefficient	variable	coefficient	variable	coefficient
895.9329	-32.373	934.5092	14.6213	1285.553	10.57296	1316.414	-2.41311
899.7905	28.55549	938.3668	5.570504	1289.41	9.409508	1320.271	-5.31319
903.6482	16.94194	965.3701	-5.90303	1293.268	3.796267	1324.129	-8.70611
915.221	12.33587	969.2278	-6.721	1297.126	-0.32284	1389.709	-0.29298
919.0787	1.910436	973.0854	14.59353	1300.983	2.767476	1393.566	-0.56368
922.9363	-28.6364	1273.98	-15.8304	1304.841	-0.40561	1397.424	-0.54517
926.7939	-12.7679	1277.838	-2.4026	1308.699	2.60186		
930.6515	-8.45968	1281.695	11.43226	1312.556	-0.11929		

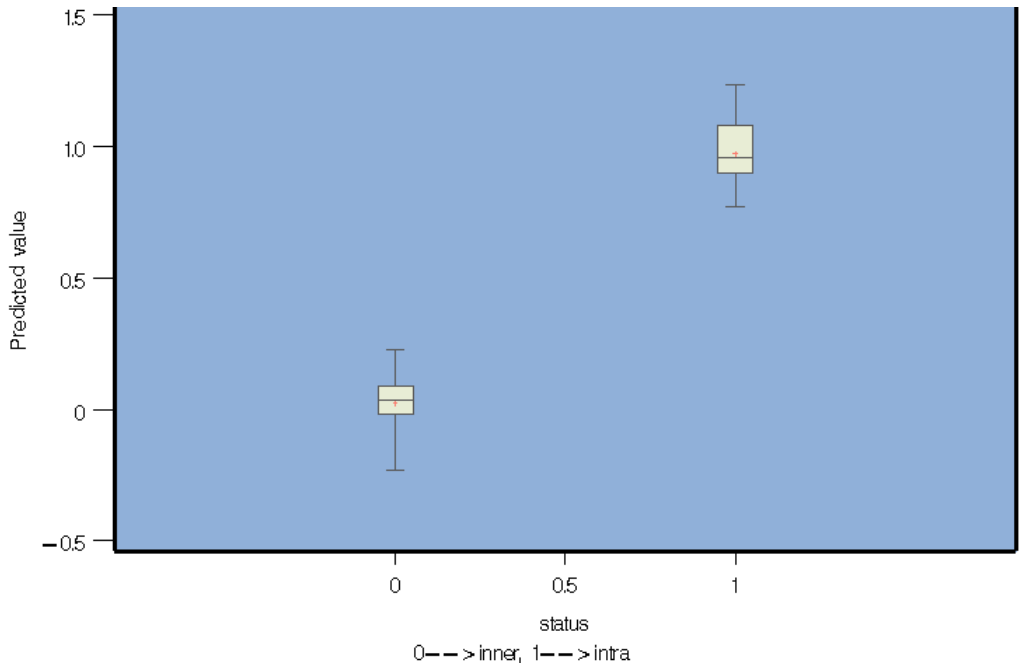


Figure- 20 Box-plot of predicted values of Adeno vs Coxsackie infected cells

Table- 17 Comparison of Adeno vs Coxsackie infected cells

Measureme-nt	Original estimate	BT shrinkage	CV shrinkage	BT final	CV final	Average
AUC	0.999999999	0.00537195	0.0374756	0.9946280	0.96252439	0.9785762

3.2 Results for 3-dimensional discrimination

The Figure-20 shows data of Inner and Intra difference for Mock, HSV1 and Coxsackie. It seems that the discrimination among three cells simultaneously is harder to achieve in compare with 2-dimensional discrimination since most of the data overlay with each other. But Figure-21 shows that there are common ranges for HSV1 and Coxsackie where the P-values are smaller than 0.0002. The locations of these 14 significant frequencies are shown by Figure-22. It can be seen that most significant frequencies concentrate around 970-980 and around 1300 cm-1. If we use first three PLS factors scores to plot three cells in a 3-dimensional space, as shown in Figure 24, 25 and 26 with different angles. Mock can be discriminated from infected cells easily. The others are harder to differentiate visually. The first 9 PLS factors from these 14 variables are used to obtain the discriminators. The coefficients of PLSR are shown by the Table-18. We compute the discriminators values for three differences, Mock-Mock, Mock-Coxsackie, and Mock-HSV1. The resulted box-plots are shown in Figure 27. The plots in Figure 27 reveal good discriminations among three cells. A few overlapped points are somewhat expected in the high-dimensional discriminations.

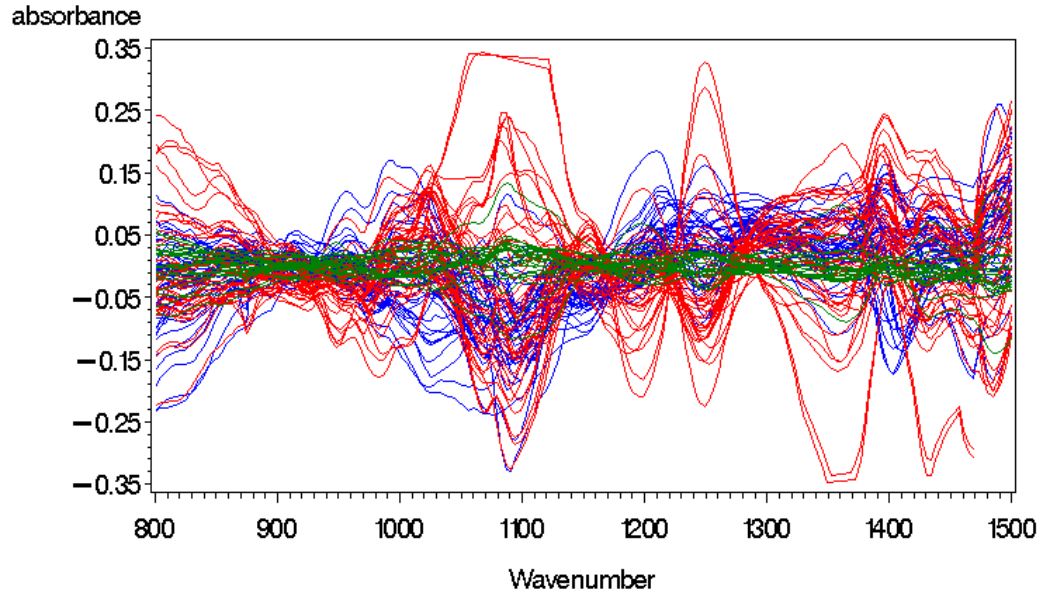


Figure- 21 Inner & Intra differences for Mock, HSV1 & Coxsackie

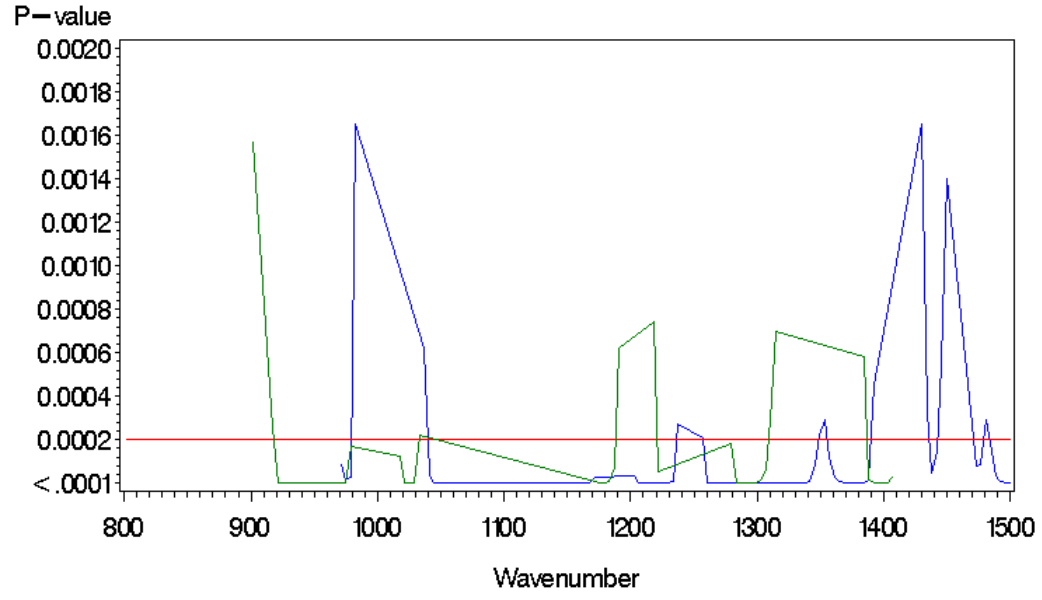


Figure- 22 P-value from WSRT for Mock-HSV1 & Mock-Coxsackie infected cells

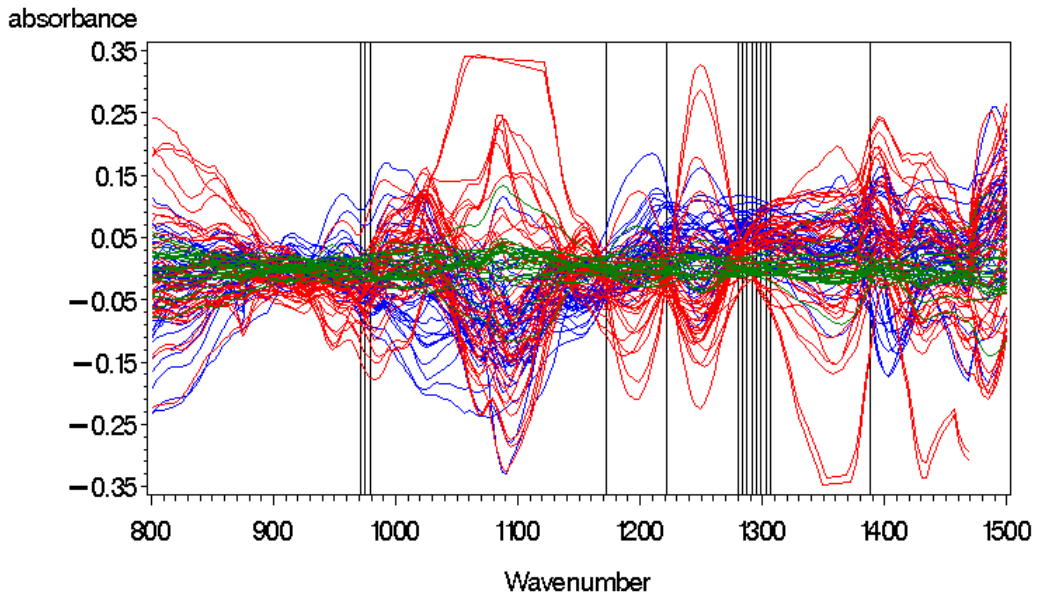


Figure- 23 Location of significant variables for Mock, HSV1 & Coxsackie infected cells

Table- 18 Percent Variation Accounted by PLS Factors for Mock, HSV1 and Coxsackie infected cells

Number of Extracted Factors	Dependent Variables			
	Current	Total	Current	Total
1	53.9804	53.9804	53.0736	53.0736
2	14.8356	68.8160	8.6209	61.6945
3	23.1883	92.0042	1.1512	62.8457
4	4.8950	96.8992	3.0527	65.8984
5	1.3388	98.2379	1.8178	67.7162
6	0.4708	98.7087	3.2611	70.9773
7	1.1270	99.8357	0.3437	71.3210
8	0.1486	99.9843	1.4746	72.7955
9	0.0064	99.9906	3.4591	76.2546
10	0.0046	99.9952	1.8920	78.1467
11	0.0017	99.9969	0.3464	78.4931
12	0.0014	99.9983	0.1537	78.6468
13	0.0010	99.9992	0.0073	78.6541
14	0.0008	100.0000	0.0001	78.6542

Table- 19 Coefficients from PLSR for selected variables

variables	971.1566	975.0142	978.8718	1171.753	1221.902	1279.766	1283.624
coefficient	37.55268	-74.9173	37.64728	-5.58801	-1.28176	-0.01432	12.04739
variables	1287.482	1291.339	1295.197	1299.055	1302.912	1306.77	1387.78
coefficient	-121.772	137.5957	52.63678	-11.4476	-14.5495	-41.9682	-1.35066

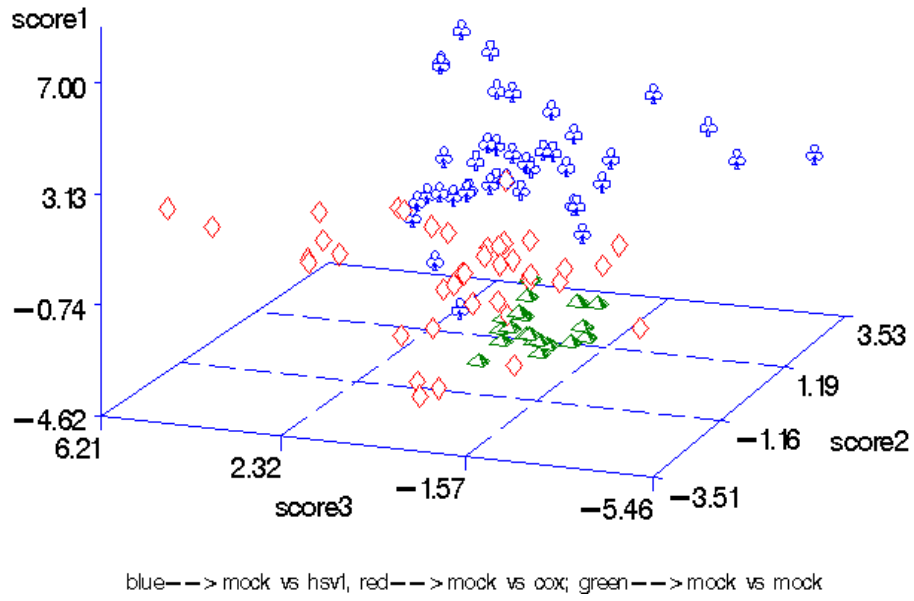


Figure- 24 Score plot for Mock, HSV1 and Coxsackie (1) infected cells

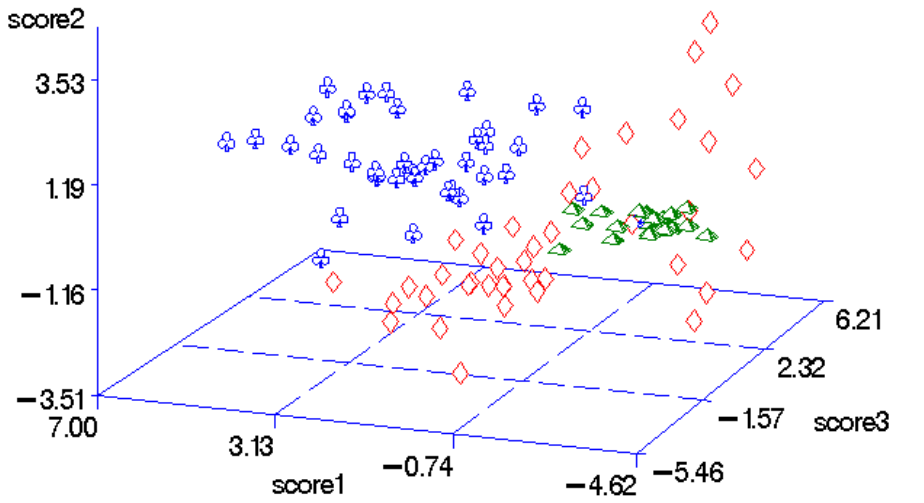
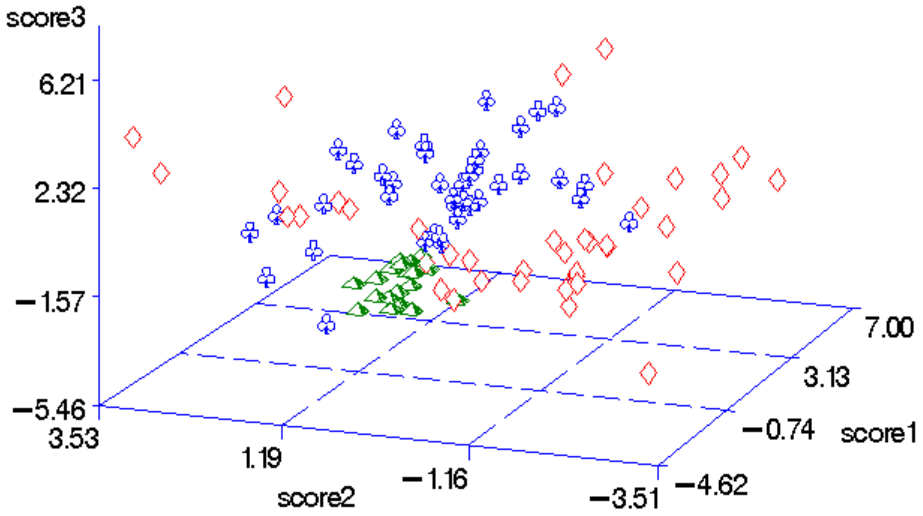
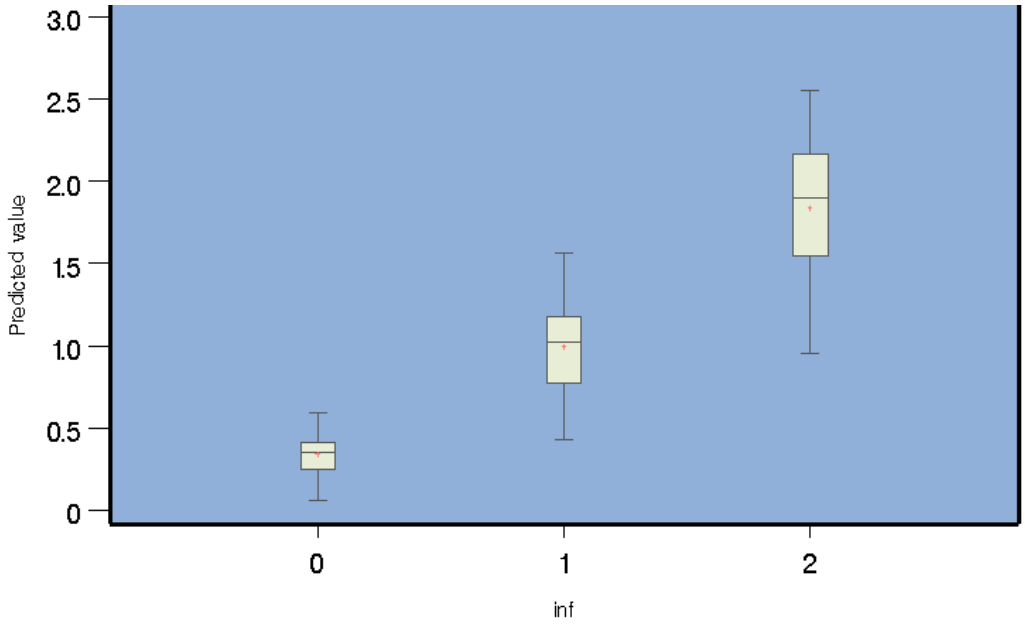


Figure- 25 Score plot for Mock, HSV1 and Coxsakie (2) infected cells



blue−−>mock vs hsv1, red−−>mock vs cox; green−−>mock vs mock

Figure- 26 Score plot for Mock, HSV1 and Coxsackie (3) infected cells



0−−>inner, 1−−>intra of mock vs cox; 2−−>intra of hsv vs mock

Figure- 27 Box-plot for Mock, HSV1 and Coxsackie infected cells

Also the multi-discrimination table is constructed by selecting two cutoff points to discriminate different cells. In order to make cutoff points more reasonable, we only pick the cutoff points to one decimal place as an example. For instance, we select the interval between 0.1 and 1 as Coxsackie; less than 0.1 as Mock and greater than 1 as HSV1. The result is shown as in Table-21. The count number represents the number of predicted values and the percentage number is the percentage of total data in different cells.

Table- 20 Multi-discrimination Table

		Status		
		mock	**coxackie**	**HSV1**
diagnosis	**mock**	17(85%)	0(0%)	0(0%)
	coxackie	3(15%)	34(85%)	4(10%)
	HSV1	0(0%)	6(15%)	36(90%)

Now, we use VUS to measure the discrimination power, the VUS and its shrinkages are shown at the Table-22. From Table-22 we can see both the linear mapping and quadratic mapping of original VUS are above 80%, which indicate 'Good' discrimination. Bootstrap method and Cross-validation method yield that the VUS shrinkages of our procedure is between 2% and 5%. The final VUS of linear mapping is about 0.80, which indicate 'Good' discrimination, and the final VUS quadratic mapping is 0.832556, also indicate 'Good' discrimination. In general, the

discrimination cannot yield as good results as pairwise comparison. However, it is still considered to be very good.

Table- 21 Comparison of Mock, HSV1 & Coxsackie infected cells

	original VUS	BT shrinkage	CV shrinkage	BT final	CV final	Average
linear mapping	0.824889	0.0194	0.045318	0.805489	0.779571	0.79253
quadratic mapping	0.862152	0.017373	0.041819	0.84478	0.820333	0.832556

Chapter 4 Conclusion

To conclude, all of the AUC and specificities of pair comparisons are larger than 95% except for the comparison of Mock (uninfected) vs Coxsackie infected cells which still have AUC and specificities larger than 90%. Therefore, we can say that the procedure of 2-dimensional comparison can discriminate any two kinds of cells excellent.

The 3-dimensional discrimination is more difficult than the 2-dimensional one since it is 4-dimensional. From the score graph, Box-plot and the VUS we can see that though the three cells are separated, there are some points where they overlay with each other. However, the result is still encouraging.

It also proves the FTIR microspectroscopy to be an effective technique in discrimination between normal and virus-infected cells at early stages of infection. In other words, using FTIR microscopy for the viruses' infection diagnosis is worth to continue for researchers.

Further studies can be improved to do the multi-dimensional discrimination. A sequential procedure may be worth to investigate. Take 3-dimensional as example, we can discriminate non-diseased cells with diseased cells firstly using the 2-dismensional discrimination procedure. Then we can apply 2-dimensional discrimination again to distinguish the rest diseased cells.

REFERENCE

1. Microbiol, V. (2007) RAPID IDENTIFICATION OF BIOLOGICAL AGENTS USING VIBRATIONAL SPECTRAL MICROSCOPY. Aug 31;123(4):305-19.

2. Beekes, M. Lasch, P. & Naumann, D. Analytical applications of Fourier transform-infrared (FT-IR) spectroscopy in microbiology and prion research.

3. Tang, T (2008) Infrared Spectroscopy In Combination With Advanced Statistical Methods For Distinguishing Viral Infected Biological Cells. Georgia State University.

4. Luo, S (2009) ADVANCED METHODOLOGY OF DISCRIMINATION IN SELECTING MOST EFFICIENT MEASUING PERIOD FOR BIOLOGICAL CELLS. Georgia State University.

5. Johnson, R. A. & Wichern, D. W. (1982). Applied Multivariate Statistical Analysis. University of Wisconsin-Madison.

6. Cha, E. (2005), *The volume under the ROC surface for multi-ordered classes*, thesis. Georgia State University.

7. Li, Y. (2009), A Generalization of AUC to and Ordered Multi-class Diagnosis to Longitudinal Data Analysis on Neurobehavioral Outcome in Pediatric Brain-Tumor Patients. Georgia State University.

8. Efron, B. (1979). Bootstrap Methods: Another Look at the Jackknife". The Annals of Statistics.

9. Swets J. A. (1995). Signal detection theory and ROC analysis in psychology and diagnostics: Collected papers. *Lawrence Erlbaum Associates.*

10. SAS Institute Inc. 2002-2005 SAS OnlineDoc. Version 9.1.3.

11. Juhasz, F.(1989), On the theoretical backgrounds of cluster analysis based on the eigenvalue problem of the association matrix. Statistics, 20: 572-581.

12. Croux, C. and Joossens, K. (2005). Influence of Observations on the Misclassification Probability in Quadratic Discriminant Analysisi', Journal of Multivariate Analysis.

13. Abdi, H. (2003). *Partial least squares (PLS) regression*. *The University of Texas at Dallas.*

14. Harz , M. Rösch , P. & Popp. J. Vibrational spectroscopy - A powerful tool for the rapid identification of microbial cells at the single-cell level. Institute of Physical Chemistry. Institute of Photonic Technology

APPENDIX

APPENDIX 1: DATA PROCESS

```
libname boots 'C:\Dongmei Wang\Research\Bootstrap\mock vs HSV1'; run;
/********** import data, divide into 2 group, take mean **********/
%macro import(data,number1,number2);
data &data;
        set &data(firstobs=5);
       var1=Wavenumber+0;
       drop xlabel Wavenumber;
run;
data &data;
        set &data;
        %do i=3 %to &number2;
              varr&i=var&i;
              drop var&i;
     %end;
data &data;
      set &data(rename=(varr3-varr&number2=var2-var&number1));
run;
data &data;
      set &data;
run;
%mend;

%macro standardize(data,number);
proc means data=&data;
        var var1-var&number;
        output out=mean mean(var1-var&number)=var1-var&number;
        output out=std std(var1-var&number)=var1-var&number;
run;
data mean; /*mean*/
        set mean;
        drop _freq_ _type_;
run;
data std; /*std*/
        set std;
        drop _freq_ _type_;
run;

data std;
        set &data mean std;
```

```
run;
proc transpose data=std out=stdtr prefix=v;
        var var1-var&number;
run;
data stdtr;
        set stdtr;
        rename v729=mean v730=std;
run;
%macro std;
        %do i=1 %to 728;
        data stdtr;
           set stdtr;
           var&i=(v&i-mean)/std;
          drop v&i;
        run;
        %end;
%mend;
%std
data &data;
        set stdtr;
       drop mean std;
run;
%mend;

/************ sumby 4 ************************************/
%macro sumby4(data1,data2); /****data1 is the data to deal with, data2 is the data
reserved ***/
data sumby4;
        set &data1;
              %do i=1 %to 182;
                    c&i=0;
                     %do j=0 %to 3;
                           %let m=%sysevalf(1+4*(&i-1)+&j, integer);
                                  c&i=c&i+var&m;
                     %end;
                    c&i=c&i/4;
              %end;
run;
data &data2;
        set sumby4;
       keep c1-c182;
run;
%mend;
```

```
/************ randomly divide into 2 group *****************/
%macro divide(data1,data2,data3,number1);
data data1;
      set data1;
      index=uniform(&number1);
run;
proc sort data=data1 out=data1; by index; run;
/************ divide into 2 group ***************/
data &data2;
      set &data1 ;
      if _n_<=&number1/2;
      drop index;
run;
data &data3;
      set &data1;
      if _n_>&number1/2;
      drop index;
run;
/************ take mean ******************/
proc means data=&data2;/***** goup 1 ****/
      output out=&data2;
run;
data &data2;
      set &data2;
      if _n_=4;
      drop _freq_ _stat_ _type_;
run;
proc means data=&data3;/******** gourp 2 *********/
      output out=&data3;
run;
data &data3;
      set &data3;
      if _n_=4;
      drop _freq_ _stat_ _type_;
run;
%mend;

%macro final(data1,data2,data3,data4,number1,number2);
proc transpose data=&data1 out=&data1 prefix=m1;run;
proc transpose data=&data2 out=&data2 prefix=m2;run;
proc transpose data=&data3 out=&data3 prefix=h1;run;
proc transpose data=&data4 out=&data4 prefix=h2;run;
/********* inner ******************/
data inner&number1;
```

```
      merge &data1 &data2;
      inner&number1=m11-m21;
      drop m11 m21;
run;
data inner&number2;
      merge &data3 &data4;
      inner&number2=h11-h21;
      drop h11 h21;
run;
/********** intra ***********/
data intra&number1;
      merge &data1 &data3;
      intra&number1=m11-h11;
      drop m11 h11;
run;
data intra&number2;
      merge &data2 &data4;
      intra&number2=m21-h21;
      drop m21 h21;
run;
%mend;
/***************** 1st group 032608 ********************/
proc import datafile='E:\data\Shan Luo\data
set\1-3group\032608\v-mock-human-t-nofix-6hpi-032808-2cm-1_2000-700
filter.csv'
      replace out=data1;
run;
%import(data1,58,59)
%standardize(data1,58)
%sumby4(data1,data1)
%divide(data1,mock1_1,mock1_2,58)

/***************** import HSV #1**********************/
proc import datafile='E:\data\Shan Luo\data
set\1-3group\032608\v-HSV1-human-t-nofix-6hpi-033108-2cm-1_2000-700 filter.csv'
      replace out=data1;
run;

%import(data1,71,72);
%standardize(data1,71)
%sumby4(data1,data1);
%divide(data1,HSV1_1,HSV1_2,71);

%final(mock1_1,mock1_2,HSV1_1,HSV1_2,1,2)
```

```
/********** merge all the 42 inner and 42 intra **********/
data boots.inner;
     if 1=1 then delete;
run;
data boots.intra;
     if 1=1 then delete;
run;

%macro merge;
%do i=1 %to 42;
     data boots.inner;
          merge boots.inner inner&i;
     run;
%end;
%do i=1 %to 42;
     data boots.intra;
          merge boots.intra intra&i;
     run;
%end;
%mend;
%merge;

data graph;
     merge boots.inner boots.intra boots.x1;
run;
data plot;
     merge boots.x1 boots.inner;
run;

footnote ' blue-->inner difference, red-->intra difference';
title 'mock vs HSV1';
symbol1 color=blue i=j line=1 w=1 h=2.5 repeat=42;
symbol2 color=red i=j line=1 w=1 h=2.5 repeat=42;
proc gplot data=graph;
plot    (inner1-inner42 intra1-intra42)*col1 / overlay;
run;
quit;
```

APPENDIX 2 Calculate the shrinkage

APPENDIX 2.1 Use bootstrap method to do the shrinkage of MOCK VS HSV1

```
proc transpose data=boots.inner out=inner;run;
data boots.inner_1; /********* m-m **********/
```

```
    set inner(rename=(c1-c182=a1-a182));
    if (mod (_n_,2)=1);
run;

run;data boots.inner_2; /********** h-h ***************/
    set inner(rename=(c1-c182=b1-b182));
    if (mod (_n_,2)=0);
run;

proc transpose data=boots.intra out=intra;run;
data boots.intra_1; /************* m1-h1 *****************/
    set intra;
    if (mod (_n_,2)=1);

run;data boots.intra_2; /************** m2-h2 ****************/
    set intra(rename=(c1-c182=d1-d182));
    if (mod (_n_,2)=0);
run;
libname boots 'C:\Dongmei Wang\Research\Bootstrap\mock vs HSV1'; run;

/************************************************/
/************ Do the analysis 100 times **********/
/***************** sample ************************/

data sampling;
    merge boots.inner_1 boots.inner_2 boots.intra_1 boots.intra_2;
    drop _name_;
    id=_n_;
run;

%macro sampling(a);
%let m=%sysevalf(&a*21+1);
%let n=%sysevalf(&a*21+21);
%do i=&m %to &n;
    %let j=%sysevalf(&i*666+666);
    data one;
        set sampling;
        index=ranuni(&j);
    run;
proc sort data=one out=one;by index;run;
data sample&i;
    set one;
    by index;
    if _n_=1;
```

```
run;
data boots_sample&a;
	set boots_sample&a sample&i;
run;
%end;
%mend;

data boots.model;
	if 1=1 then delete;
run;
data boots.validation;
	if 1=1 then delete;
run;
data boots.percent_number;
	if 1=1 then delete;
run;
data boots.percent_number;
	if 1=1 then delete;
run;

%macro repeat;
%do a=0 %to 99;

data boots_sample&a;
	if 1=1 then delete;
run;
%sampling(&a);

data boots_sample&a;
	set boots_sample&a;
	drop index;
run;

data boots.inner1; set boots_sample&a(keep=a1-a182
rename=(a1-a182=v1-v182));run;
data boots.inner2; set boots_sample&a(keep=b1-b182
rename=(b1-b182=v1-v182));run;
data boots.intra1; set boots_sample&a(keep=c1-c182
rename=(c1-c182=v1-v182));run;
data boots.intra2; set boots_sample&a(keep=d1-d182
rename=(d1-d182=v1-v182));run;

/********** sign rank test ***************/
```

```
data sign_rank;
        set boots.intra1 boots.intra2;
run;

ods output    TestsForLocation=t3;
proc univariate data=sign_rank;run;
*ods trace off;
data t4;
      set t3;
      if Testlab="S" then output;
run;

/************ count the # of significant variable *************/
data depend;
set t4(keep=pValue);
retain z 0;
if pValue<=0.0002 then z=z+1;
if _n_=182 then call symput("counterx",z);
run;

%let counterxzero=%sysevalf(&counterx+0);
%let counterxone=%sysevalf(&counterx+1);
%let counterxtwo=%sysevalf(&counterx+2);

/************* record the selected variables ***************/
data a&a; a=&counterxzero;run;
data boots.number_var;
      if 1=1 then delete;
run;
data boots.number_var;
      set boots.number_var a&a;
run;

/******* select significant value for inner 1 **********/
proc transpose data=boots.inner1 out=a_inner prefix=v;run;
data select1;
      merge t4(keep=pvalue) a_inner;
run;
data select1;
      set select1;
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select1 out=inner1 prefix=a;run;
```

```
/******* select significant value for inner 2 **********/
proc transpose data=boots.inner2 out=b_inner prefix=v;run;
data select2;
      merge t4(keep=pvalue) b_inner;
run;
data select2;
      set select2;
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select2 out=inner2 prefix=a;run;

/******* select significant value for intra 1 **********/
proc transpose data=boots.intra1 out=a_intra prefix=v;run;
data select3;
      merge t4(keep=pvalue) a_intra;
run;
data select3;
      set select3;
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select3 out=intra1 prefix=a;run;

/******* select significant value for intra 2 **********/
proc transpose data=boots.intra2 out=b_intra prefix=v;run;
data select4;
      merge t4(keep=pvalue) b_intra;
run;
data select4;
      set select4;
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select4 out=intra2 prefix=a;run;


/********** pls use random sample data as model data ***************/
data inner_pls;
      set inner1 inner2;
      inf=0;
run;
data intra_pls;
```

```
      set intra1 intra2;
      inf=1;
run;
data pls;
      set inner_pls intra_pls;
run;

/*********** pls use original data as validation data *************/
proc transpose data=boots.inner out=inner prefix=v;run;
data inner;set inner;drop _name_;run;
proc transpose data=boots.intra out=intra prefix=v;run;
data intra;set intra;drop _name_;run;

/********** select significant value for original data inner **********/
data select3;
      merge t4(keep=pvalue) boots.inner;
run;
data select3;
      set select3;
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select3 out=inner prefix=a;run;

/********** select significant value for original data inner **********/
data select4;
      merge t4(keep=pvalue) boots.intra;
run;
data select4;
      set select4;
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select4 out=intra prefix=a;run;

/********************** count number of components
**********************************/
data pls1;
      set pls inner intra;
run;

ods output    PercentVariation=percent;
proc pls data =pls1;/*pls is selected intra*/
      model inf=a1-a&counterxzero;
```

```
output out=no PREDICTED=no;
run;

data percent;
      set percent(keep=TotalYVariation rename=(TotalYVariation=y));
      retain p 0;
      if y<=94.5 then p=p+1;
        call symput("percent",p);
run;
%let percentnumber=%sysevalf(&percent+1);

/***************** record number of components ***********************/
data p; p=&percentnumber;run;

data boots.percent_number;
      set boots.percent_number p;
run;
/*****************************************************************
******/
proc pls data =pls1 nfac=&percentnumber;/*pls is selected intra*/
      model inf=a1-a&counterxzero;
output out=one PREDICTED=p;
run;

data boots.percent_number;
      set boots.percent_number p;
run;
/*****************************************************************
*********/
data validation;
        set one;
       if _n_>84;
run;
data model;
        set one;
       if _n_<=84;
run;

/************* shan luo spec *****************/
data tinner;
      set model(keep=p rename=(p=x));
      if _n_<=42;
run;
data tintra;
```

```
      set model(keep=p rename=(p=y));
      if _n_>42;
run;
data ptinner;
      set validation(keep=p);
      if _n_<=42;
run;
data ptintra;
      set validation(keep=p rename=(p=q));
      if _n_>42;
run;

/*************************************************/
/*calucalte specificity for sample and population */
/*************************************************/

%macro spec(datainner,dataintra,c1,c2,normal,result);
proc means noprint data=&datainner mean std; var &c1; output out=xnormal
mean=meanx std=stdx;run;
proc means noprint data=&dataintra mean std; var &c2; output out=ynormal
mean=meany std=stdy;run;

data spec;merge xnormal ynormal;run;
data spec;set spec(drop=_type_ _freq_);run;

      /*(3) compute spec1, spec2, spec3, AUC*/
      data norm;/*PROBIT() and PROBNORM()*/
      set spec;
      cutpt1=meany+(-1.6448536)*stdy;tr1=(cutpt1-meanx)/stdx;spec1=probnorm(tr1
);/*95% sensitivity*/
      cutpt2=meany+(-1.2815516)*stdy;tr2=(cutpt2-meanx)/stdx;spec2=probnorm(tr2
);/*90% sensitivity*/
      cutpt3=meany+(-0.8416212)*stdy;tr3=(cutpt3-meanx)/stdx;spec3=probnorm(tr3
);/*80% sensitivity*/
      se=sqrt(stdy*stdy+stdx*stdx);meandiff=meany-meanx;tile=meandiff/se;
      AUC=probnorm(tile);
      run;
   data &normal;set norm (keep=spec1 spec2 spec3 AUC);run;

data &result;set &result &normal;run;

%mend;

%spec(tinner,tintra,x,y,orig&a,resultorig);/*sample*/
```

```
%spec(ptinner,ptintra,p,q,validate&a,resultvalidate);/*original population*/

data boots.model;
    set boots.model orig&a;
run;
data boots.validation;
    set boots.validation validate&a;
run;

%end;
%mend;

%repeat;

proc means data=boots.model;
    output out=model;
run;
data model;
    set model;
    if _n_=4;
    drop _stat_ _freq_ _type_;
run;

proc means data=boots.validation;
    output out=validation;
run;
data validation;
    set validation;
    if _n_=4;
    drop _stat_ _freq_ _type_;
run;

data shrinkage;
    set model validation;
run;

proc transpose data=shrinkage    out=boots.shrinkage_mock_HSV1;run;
data boots.shrinkage_mock_HSV2;
    set boots.shrinkage_mock_HSV1;
    shrinkage=col1-col2;
run;

/************ detail of shrinkage ****************/
```

```
data one;
    set boots.model(rename=(spec1=m1 spec2=m2 spec3=m3 auc=am));
run;
data two;
    merge one boots.validation;
run;

data boots.shrinkage_detail;
    set two;
    spec_1=m1-spec1;
    spec_2=m2-spec2;
    spec_3=m3-spec3;
    auc_=am-auc;
    drop spec1 m1 spec2 m2 spec3 m3 auc am;
run;
```

APPENDIX 2.2 Use 2-fold cross validation method to do the shrinkage of MOCK VS HSV1

```
/********** select significant value for original data inner **********/
data boots.validation_cross;
    if 1=1 then delete;
run;
data boots.model_cross;
    if 1=1 then delete;
run;
data boots.percent_number_cross;
    if 1=1 then delete;
run;
data boots.var_cross; if 1=1 then delete; run;

%macro cross;

%do a=0 %to 99;
data inner1;
    set boots.inner_1;
    id=ranuni(&a*888);
run;
data id;set inner1;keep id;run;
proc sort data=inner1 out=inner1;by id;run;
data inner1_1; set inner1; by id; if _n_<=11;drop id;run;
data inner1_2; set inner1; by id; if _n_>11;drop id;run;
```

```
data inner2;
      merge boots.inner_2(rename=(b1-b182=a1-a182)) id;
run;
proc sort data=inner2 out=inner2;by id;run;
data inner2_1; set inner2; by id; if _n_<=11;drop id;run;
data inner2_2; set inner2; by id; if _n_>11;drop id;run;

data intra1;
      merge boots.intra_1(rename=(c1-c182=a1-a182)) id;
run;
proc sort data=intra1 out=intra1;by id;run;
data intra1_1; set intra1; by id; if _n_<=11;drop id;run;
data intra1_2; set intra1; by id; if _n_>11;drop id;run;

data intra2;
      merge boots.intra_2(rename=(d1-d182=a1-a182)) id;
run;
proc sort data=intra2 out=intra2;by id;run;
data intra2_1; set intra2; by id; if _n_<=11;drop id;run;
data intra2_2; set intra2; by id; if _n_>11;drop id;run;

data inner1;/*************** model data inner *****************/
      set inner1_1 inner2_1;
      *inf=0;
run;

data inner2;/*************** validation data inner *****************/
      set inner1_2 inner2_2;
run;

data intra1;/*************** model data intra *****************/
      set intra1_1 intra2_1;
      *inf=1;
run;

data intra2;/*************** validation data intra *****************/
      set intra1_2 intra2_2;
run;


/****************** WSRT ********************/
data sign_rank; set intra1; run;
ods output    TestsForLocation=t3;
```

```
proc univariate data=sign_rank;run;
*ods trace off;
data t4;
	set t3;
	if Testlab="S" then output;
run;

data depend;
set t4(keep=pValue);
retain z 0;
if pValue<=0.0002 then z=z+1;
call symput("counterx",z);
run;

%let counterxzero=%sysevalf(&counterx+0);
%let counterxone=%sysevalf(&counterx+1);
%let counterxtwo=%sysevalf(&counterx+2);
/****************** count selected variables **********************/
data a; a=&counterxzero;run;
data boots.var_cross;
	set boots.var_cross a;
run;
/*********************************************************************
**/
/********** select significant value for original data inner **********/
proc transpose data=inner1 out=inner1;run;
data select3;
	merge t4(keep=pvalue) inner1;
run;
data select3;
	set select3;
	if pvalue>0.0002 then delete;
	drop pvalue;
run;
proc transpose data=select3 out=inner1 prefix=a;run;
data inner1;set inner1; inf=0;run; /******** inner1 as model ********/

/*****************************************************************/
proc transpose data=inner2 out=inner2;run;
data select3;
	merge t4(keep=pvalue) inner2;
run;
data select3;
	set select3;
```

```
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select3 out=inner2 prefix=a;run;
/*************************************************************/
proc transpose data=intra1 out=intra1;run;
data select3;
      merge t4(keep=pvalue) intra1;
run;
data select3;
      set select3;
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select3 out=intra1 prefix=a;run;
data intra1;set intra1;inf=1;run;/******** intra1 as model ********/
/*************************************************************/
proc transpose data=intra2 out=intra2;run;
data select3;
      merge t4(keep=pvalue) intra2;
run;
data select3;
      set select3;
      if pvalue>0.0002 then delete;
      drop pvalue;
run;
proc transpose data=select3 out=intra2 prefix=a;run;
/*************************************************************
/

data cross_pls;
      set inner1    intra1 inner2 intra2;
run;

ods output    PercentVariation=percent_cross;
proc pls data =cross_pls;/*pls is selected intra*/
      model inf=a1-a&counterxzero;
output out=no PREDICTED=no;
run;

data percent_cross;
      set percent_cross(keep=TotalYVariation rename=(TotalYVariation=y));
      retain p 0;
      if y<=94.5 then p=p+1;
```

```
        call symput("percent",p);
run;
%let percentnumber=%sysevalf(&percent+1);
/******************** count number of components
************************/
data p; p=&percentnumber;run;
data boots.percent_number_cross;
      set boots.percent_number_cross p;
run;

proc pls data =cross_pls nfac=&percentnumber;/*pls is selected intra*/
      model inf=a1-a&counterxzero;
output out=one PREDICTED=p;
run;


data model;
        set one;
       if _n_<=44;
run;
data validation;
        set one;
       if _n_>44;
run;

/************* shan luo spec *****************/
data minner;
      set model(keep=p rename=(p=x));
      if _n_<=22;
run;
data mintra;
      set model(keep=p rename=(p=y));
      if _n_>22;
run;
data vtinner;
      set validation(keep=p);
      if _n_<=20;
run;
data vtintra;
      set validation(keep=p rename=(p=q));
      if _n_>20;
run;
%spec(minner,mintra,x,y,orig&a,resultorig);/*sample*/
%spec(vtinner,vtintra,p,q,validate&a,resultvalidate);/*original population*/
```

```
data boots.model_cross;
      set boots.model_cross orig&a;
run;
data boots.validation_cross;
      set boots.validation_cross validate&a;
run;

%end;
%mend;
%cross;


/**************** calculate the shrinkage ********************/
proc means data=boots.model_cross noprint;
      output out=one;
run;
data one;
      set one;
      if _n_=4;
      drop _freq_ _stat_ _type_;
run;
proc transpose data=one out=one prefix=model_cross;run;

proc means data=boots.validation_cross noprint;
      output out=two;
run;
data two;
      set two;
      if _n_=4;
      drop _freq_ _stat_ _type_;
run;
proc transpose data=two out=two prefix=validation_cross;run;

data boots.shrinkage_new_cross;
      merge one two;
      shrinkage=model_cross1-validation_cross1;
run;
```

Appendix 2.3 Plot the location of selected variables for PLSR

```
/********************* graph for selected variables of mock vs HSV1
**********************/
libname graph 'C:\Dongmei Wang\Research\graph for separate mock HSV cox';run;
data mh;
```

```
    merge graph.intra_m_h graph.inner_m_h graph.x1;
run;

footnote h=2 j=1    ' blue-->inner difference, red-->intra difference';run;
axis1 label=(c=black"absorbance")order=(-0.35 to 0.35 by 0.1);
axis2 label=(c=black"Wavenumber" )order=(800 to 1500 by 100);
title 'Location of selected variables on inner and intra for mock vs HSV1';
symbol1 color=blue i=j line=1 w=1 h=2.5 repeat=42;
symbol2 color=red i=j line=1 w=1 h=2.5 repeat=42;
proc gplot data=mh;
plot    (inner1-inner42 intra1-intra42)*col1 /overlay
                                                haxis=axis2
                                                vaxis=axis1
                                                      href=969.22776 973.085383
    976.943006  1034.80735  1038.664973     1042.522595     1046.380218
    1050.237841     …    1057.953087     1381.993413     1385.851036
    1497.722101;
run;
quit;

/******************** take average for inner intra respectively
*********************/
proc transpose data=graph.intra_m_h out=intra;run;
proc means data=intra noprint;
    output out=mean;
run;
data mean_intra;
    set mean;
    if _n_=4;
    drop _stat_ _freq_ _type_;
run;

proc transpose data=graph.inner_m_h out=inner;run;
proc means data=inner noprint;
    output out=mean;
run;
data mean_inner;
    set mean;
    if _n_=4;
    drop _stat_ _freq_ _type_;
run;

data graph;
    set mean_inner mean_intra;
```

```
run;

proc transpose data=graph out=graph prefix=v;run;
data graph;
     merge graph graph.x1;
run;

footnote h=2 j=1 ' blue-->inner difference, red-->intra difference';
axis1 label=(c=black"absorbance");
axis2 label=(c=black"Wavenumber" )order=(800 to 1500 by 100);
title 'Location of selected variables on the mean of inner and intra for mock vs HSV1';
symbol1 color=blue i=j line=1 w=1 h=2.5 repeat=1;
symbol2 color=red i=j line=1 w=1 h=2.5 repeat=1;
proc gplot data=graph;
plot    (v1 v2)*col1 /overlay
                          haxis=axis2
                        vaxis=axis1
                          href=969.22776 973.085383    976.943006  1034.80735
     1038.664973      1042.522595      1046.380218      1050.237841
...  1385.851036      1497.722101;
run;
quit;
```

APPENDIX 3: COMPARISON OF MOCK, HSV1 AND COXSACKIE

APPENDIX 3.1 Bootstrap validation for the shrinkage of VUS

```
/**** Use bootstrap method to generate sample and do the shrinkage********/
/******for 100 times to evaluate the diagnostic performance          ********/

/****************** bootstrap sample ************************/
libname class 'C:\Dongmei Wang\Research\vus bootstrap'; run;proc transpose
data=class.inner out=one prefix=mm;run;
data class.inner_m_m;
     set one;
     if (mod(_n_,2)=1);
run;

proc transpose data=class.intra_m_h out=mh prefix=mh;run;
data class.mh1 class.mh2;
     set mh;
     if (mod(_n_,2)=1) then output class.mh1;
     else output class.mh2;
run;
```

```
proc transpose data=class.intra_m_c out=mc prefix=mc;run;
data class.mc1 class.mc2;
	set mc;
	if (mod(_n_,2)=1) then output class.mc1;
	else output class.mc2;
run;

data class.mh_pop;/********* pop for mock-HSV1 **********/
	merge class.mh1(rename=(mh1-mh182=m1-m182))
class.mh2(rename=(mh1-mh182=h1-h182));
	drop _name_;
	id=_n_;
run;

data class.mc_pop;/********* pop for mock-HSV1 **********/
	merge class.mc1(rename=(mc1-mc182=m1-m182))
class.mc2(rename=(mc1-mc182=c1-c182));
	drop _name_;
	id=_n_;
run;

proc transpose data=class.inner_m_m out=one;run;/***** original data as
validation *******/
data class.originaldata;
	merge class.intra_m_h class.intra_m_c(rename=(intra1-intra36=mc1-mc36))
one;
run;
proc transpose data=class.originaldata out=class.originaldata prefix=v;run;

/***************** generate bootstrap sample ********************/
%macro sampling(data1,a,n);/** data1=population data2=sample**/
%let m=%sysevalf(&a*&n+1);
%let k=%sysevalf(&a*&n+&n);
%do i=&m %to &k;
	%let j=%sysevalf(&i*666+666);
	data one;
		set &data1;
		index=ranuni(&j);
	run;
proc sort data=one out=one;by index;run;
data sample&i;
	set one;
	by index;
	if _n_=1;
```

```
run;
data sample&a;
	set sample&a sample&i;
run;
data sample&a;
	set sample&a;
	drop index;
run;
%end;
%mend;

/********************* calculate the mean std for VUS
**********************/
%macro calculate(data);
/*********** sign rank test ***************/
ods trace on;
ods output    TestsForLocation=p;
proc univariate data=&data;run;
data p;
	set p;
	if Testlab="S" then output;
run;
%mend;

/************ find the common range ***************/
%macro range(data);
proc transpose data=&data out=&data prefix=v;run;
data one;
	merge &data class.p;
run;
data one;
	set one;
	if p1<=0.0002;
	if p2<=0.0002;
	drop p1 p2;
run;
proc transpose data=one out=one prefix=v;run;
%mend;

%macro mean_std(n);
data two;
	set one(keep=p inf);
	if inf=&n;
	drop inf;
```

```
run;
proc means data=two noprint;
	output out=mean;
run;
data mean;
	set mean;
	if _n_>=4;
	drop _stat_ _type_ _freq_;
run;
proc transpose data=mean out=mean prefix=v;run;
%mend;

data class.percent_number;
	if 1=1 then delete;
run;
data class.var_number;
	if 1=1 then delete;
run;
data class.mean_std_mh;
	if 1=1 then delete;
run;
data class.mean_std_mc;
	if 1=1 then delete;
run;
data class.mean_std_mm;
	if 1=1 then delete;
run;
data class.mean_std_mh_v;
	if 1=1 then delete;
run;
data class.mean_std_mc_v;
	if 1=1 then delete;
run;
data class.mean_std_mm_v;
	if 1=1 then delete;
run;

%macro repeat;
%do a=0 %to 99;
data sample&a;/************* sample for mock-HSV1 21+21***************/
	if 1=1 then delete;
run;
%sampling(class.mh_pop,&a,21);
```

```
data mh_sample1; set sample&a(keep=m1-m182
rename=(m1-m182=mh1-mh182));run;
data mh_sample2; set sample&a(keep=h1-h182
rename=(h1-h182=mh1-mh182));run;

data mh_sample;
    set mh_sample1 mh_sample2;
run;

data sample&a;/************* sample for mock-cox 18+18 ***************/
    if 1=1 then delete;
run;
%sampling(class.mc_pop,&a,18);

data mc_sample1; set sample&a(keep=m1-m182
rename=(m1-m182=mc1-mc182));run;
data mc_sample2; set sample&a(keep=c1-c182 rename=(c1-c182=mc1-mc182));run;

data mc_sample;
    set mc_sample1 mc_sample2;
run;

data sample&a;/************* sample for mock-mock 21 ***************/
    if 1=1 then delete;
run;
%sampling(class.inner_m_m,&a,21);
data mm_sample;
    set sample&a;
run;

%calculate(mh_sample);/************* sign rank sum test ************/
data p1;set p(keep=pvalue rename=(pvalue=p1));run;
%calculate(mc_sample);
data p2;set p(keep=pvalue rename=(pvalue=p2));run;

data class.p;
    merge p1 p2;
run;

%range(mh_sample);/*************** find the common range
*********************/
data mh;
    set one;
    inf=2;
```

```
run;

%range(mc_sample);
data mc;
      set one;
      inf=1;
run;

%range(mm_sample);
data mm;
      set one;
      inf=0;
run;
data original_data; set class.originaldata;run;
%range(original_data);/********** select range for original data **********/
data original_data;
      set one;
run;

data depend;/************** count the number of variables used *************/
      set class.p;
      retain p 0;
      if p1<=0.0002 & p2<=0.0002 then p=p+1;
      call symput("counterxzero",p);
run;
%let counter=%sysevalf(&counterxzero+0);

data a; a=&counter; run;
data class.var_number;
      set class.var_number a;
run;

data pls;
      set mh mc mm original_data;
run;

ods output    PercentVariation=percent;/************** count the percentage
**************/
proc pls data =pls;
      model inf=v1-v&counter/solution;
output out=no PREDICTED=no;
run;

data percent;
```

```
      set percent(keep=TotalYVariation rename=(TotalYVariation=y));
      retain p 0;
      if y<=94.5 then p=p+1;
      call symput("percent",p);
run;
%let percentnumber=%sysevalf(&percent+0);
/***************** record number of components **********************/
data p; p=&percentnumber;run;

data class.percent_number;
      set class.percent_number p;
run;
/*****************************************************************
******/
proc pls data =pls nfac=&percentnumber;/*pls is selected intra*/
      model inf=v1-v&counter;
output out=one PREDICTED=p;
run;

%mean_std(2); /************* mean std for VUS of bootstrap data
***************/
data class.mean_std_mh;
      set class.mean_std_mh mean;
run;
%mean_std(1);
data class.mean_std_mc;
      set class.mean_std_mc mean;
run;
%mean_std(0);
data class.mean_std_mm;
      set class.mean_std_mm mean;
run;

data one; set one; if _n_>99;run;
data one;
      set one;
      if _n_<=42 then inf=2;
      if _n_<=78 & _n_>42 then inf=1;
      if _n_>78 then inf=0;
run;

%mean_std(2); /************* mean std for VUS ***************/
data class.mean_std_mh_v;
      set class.mean_std_mh_v mean;
```

```
run;
%mean_std(1);
data class.mean_std_mc_v;
      set class.mean_std_mc_v mean;
run;
%mean_std(0);
data class.mean_std_mm_v;
      set class.mean_std_mm_v mean;
run;

%end;
%mend;
%repeat;

/*********************Theoretical VUS************************/
libname vus 'C:\Dongmei Wang\Research\vus bootstrap';run;
proc iml;
use vus.mean_std_mh;
read all into mh;
close vus.mean_std_mh;

use vus.mean_std_mc;
read all into mc;
close vus.mean_std_mc;

use vus.mean_std_mm;
read all into mm;
close vus.mean_std_mm;

vus=j(100,1,0);

do i=1 to 100;
a1=0.333333;a2=0.333333;a3=1-a1-a2;/**probability**/
n1=21;n2=36;n3=42;
mu1=mm[i,1];mu2=mc[i,1];mu3=mh[i,1];/*mu1<mu2<mu3*/
sigma1=mm[i,2];sigma2=mc[i,2];sigma3=mh[i,2];
iteration=300;*simulation times;
success=0;
*find real VUS using Gaussian quadrature;
LL1=mu1-3*sigma1;UL1=mu1+3*sigma1;
LL2=mu2-3*sigma2;UL2=mu2+3*sigma2;
LL3=mu3-3*sigma3;UL3=mu3+3*sigma3;
w1=sqrt(5-2*sqrt(10/7))/3;w2=-sqrt(5-2*sqrt(10/7))/3;
w3=sqrt(5+2*sqrt(10/7))/3;w4=-sqrt(5+2*sqrt(10/7))/3;w5=0;
```

```
k11=(UL1-LL1)/2*w1+(LL1+UL1)/2;k12=(UL1-LL1)/2*w2+(LL1+UL1)/2;
k13=(UL1-LL1)/2*w3+(LL1+UL1)/2;k14=(UL1-LL1)/2*w4+(LL1+UL1)/2;k15=(UL1-
LL1)/2*w5+(LL1+UL1)/2;
k21=(UL2-LL2)/2*w1+(LL2+UL2)/2;k22=(UL2-LL2)/2*w2+(LL2+UL2)/2;
k23=(UL2-LL2)/2*w3+(LL2+UL2)/2;k24=(UL2-LL2)/2*w4+(LL2+UL2)/2;k25=(UL2-
LL2)/2*w5+(LL2+UL2)/2;
k31=(UL3-LL3)/2*w1+(LL3+UL3)/2;k32=(UL3-LL3)/2*w2+(LL3+UL3)/2;
k33=(UL3-LL3)/2*w3+(LL3+UL3)/2;k34=(UL3-LL3)/2*w4+(LL3+UL3)/2;k35=(UL3-
LL3)/2*w5+(LL3+UL3)/2;
f1=a2**2/((a1+a2)*(a2+a3))*probnorm((k21-mu1)/sigma1)*(1-probnorm((k21-
mu3)/sigma3))*pdf('normal',k21,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k11-mu2)/sigma2))*(1-probnorm((k11-
mu3)/sigma3))*pdf('normal',k11,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k31-mu1)/sigma1)*probnorm((k31-
mu2)/sigma2)*pdf('normal',k31,mu3,sigma3)*(UL3-LL3)/2;
f2=a2**2/((a1+a2)*(a2+a3))*probnorm((k22-mu1)/sigma1)*(1-probnorm((k22-
mu3)/sigma3))*pdf('normal',k22,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k12-mu2)/sigma2))*(1-probnorm((k12-
mu3)/sigma3))*pdf('normal',k12,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k32-mu1)/sigma1)*probnorm((k32-
mu2)/sigma2)*pdf('normal',k32,mu3,sigma3)*(UL3-LL3)/2;
f3=a2**2/((a1+a2)*(a2+a3))*probnorm((k23-mu1)/sigma1)*(1-probnorm((k23-
mu3)/sigma3))*pdf('normal',k23,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k13-mu2)/sigma2))*(1-probnorm((k13-
mu3)/sigma3))*pdf('normal',k13,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k33-mu1)/sigma1)*probnorm((k33-
mu2)/sigma2)*pdf('normal',k33,mu3,sigma3)*(UL3-LL3)/2;
f4=a2**2/((a1+a2)*(a2+a3))*probnorm((k24-mu1)/sigma1)*(1-probnorm((k24-
mu3)/sigma3))*pdf('normal',k24,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k14-mu2)/sigma2))*(1-probnorm((k14-
mu3)/sigma3))*pdf('normal',k14,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k34-mu1)/sigma1)*probnorm((k34-
mu2)/sigma2)*pdf('normal',k34,mu3,sigma3)*(UL3-LL3)/2;
f5=a2**2/((a1+a2)*(a2+a3))*probnorm((k25-mu1)/sigma1)*(1-probnorm((k25-
mu3)/sigma3))*pdf('normal',k25,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k15-mu2)/sigma2))*(1-probnorm((k15-
mu3)/sigma3))*pdf('normal',k15,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k35-mu1)/sigma1)*probnorm((k35-
mu2)/sigma2)*pdf('normal',k35,mu3,sigma3)*(UL3-LL3)/2;
VUS[i,1]=(f1+f2)*(322+13*sqrt(70))/900+(f3+f4)*(322-
13*sqrt(70))/900+f5*128/225+a1*a3/(4*(a1+a2)*(a2+a3))*probnorm((mu3-
mu1)/sqrt(sigma1**2+sigma3**2));/********** wi f(xi) *************/
end;
print vus;
```

```
create vus.vus_original from vus;
append from vus;
quit;

/********************** calculate VUS for validation data
************************/
proc iml;
use vus.mean_std_mh_v;
read all into mh;
close vus.mean_std_mh_v;

use vus.mean_std_mc_v;
read all into mc;
close vus.mean_std_mc_v;

use vus.mean_std_mm_v;
read all into mm;
close vus.mean_std_mm_v;

vus=j(100,1,0);

do i=1 to 100;
a1=0.333333;a2=0.333333;a3=1-a1-a2;/**probability**/
n1=21;n2=36;n3=42;
mu1=mm[i,1];mu2=mc[i,1];mu3=mh[i,1];/*mu1<mu2<mu3*/
sigma1=mm[i,2];sigma2=mc[i,2];sigma3=mh[i,2];
iteration=300;*simulation times;
success=0;
*find real VUS using Gaussian quadrature;
LL1=mu1-3*sigma1;UL1=mu1+3*sigma1;
LL2=mu2-3*sigma2;UL2=mu2+3*sigma2;
LL3=mu3-3*sigma3;UL3=mu3+3*sigma3;
w1=sqrt(5-2*sqrt(10/7))/3;w2=-sqrt(5-2*sqrt(10/7))/3;
w3=sqrt(5+2*sqrt(10/7))/3;w4=-sqrt(5+2*sqrt(10/7))/3;w5=0;
k11=(UL1-LL1)/2*w1+(LL1+UL1)/2;k12=(UL1-LL1)/2*w2+(LL1+UL1)/2;
k13=(UL1-LL1)/2*w3+(LL1+UL1)/2;k14=(UL1-LL1)/2*w4+(LL1+UL1)/2;k15=(UL1-
LL1)/2*w5+(LL1+UL1)/2;
k21=(UL2-LL2)/2*w1+(LL2+UL2)/2;k22=(UL2-LL2)/2*w2+(LL2+UL2)/2;
k23=(UL2-LL2)/2*w3+(LL2+UL2)/2;k24=(UL2-LL2)/2*w4+(LL2+UL2)/2;k25=(UL2-
LL2)/2*w5+(LL2+UL2)/2;
k31=(UL3-LL3)/2*w1+(LL3+UL3)/2;k32=(UL3-LL3)/2*w2+(LL3+UL3)/2;
k33=(UL3-LL3)/2*w3+(LL3+UL3)/2;k34=(UL3-LL3)/2*w4+(LL3+UL3)/2;k35=(UL3-
LL3)/2*w5+(LL3+UL3)/2;
f1=a2**2/((a1+a2)*(a2+a3))*probnorm((k21-mu1)/sigma1)*(1-probnorm((k21-
```

```
mu3)/sigma3))*pdf('normal',k21,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k11-mu2)/sigma2))*(1-probnorm((k11-
mu3)/sigma3))*pdf('normal',k11,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k31-mu1)/sigma1)*probnorm((k31-
mu2)/sigma2)*pdf('normal',k31,mu3,sigma3)*(UL3-LL3)/2;
f2=a2**2/((a1+a2)*(a2+a3))*probnorm((k22-mu1)/sigma1)*(1-probnorm((k22-
mu3)/sigma3))*pdf('normal',k22,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k12-mu2)/sigma2))*(1-probnorm((k12-
mu3)/sigma3))*pdf('normal',k12,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k32-mu1)/sigma1)*probnorm((k32-
mu2)/sigma2)*pdf('normal',k32,mu3,sigma3)*(UL3-LL3)/2;
f3=a2**2/((a1+a2)*(a2+a3))*probnorm((k23-mu1)/sigma1)*(1-probnorm((k23-
mu3)/sigma3))*pdf('normal',k23,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k13-mu2)/sigma2))*(1-probnorm((k13-
mu3)/sigma3))*pdf('normal',k13,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k33-mu1)/sigma1)*probnorm((k33-
mu2)/sigma2)*pdf('normal',k33,mu3,sigma3)*(UL3-LL3)/2;
f4=a2**2/((a1+a2)*(a2+a3))*probnorm((k24-mu1)/sigma1)*(1-probnorm((k24-
mu3)/sigma3))*pdf('normal',k24,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k14-mu2)/sigma2))*(1-probnorm((k14-
mu3)/sigma3))*pdf('normal',k14,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k34-mu1)/sigma1)*probnorm((k34-
mu2)/sigma2)*pdf('normal',k34,mu3,sigma3)*(UL3-LL3)/2;
f5=a2**2/((a1+a2)*(a2+a3))*probnorm((k25-mu1)/sigma1)*(1-probnorm((k25-
mu3)/sigma3))*pdf('normal',k25,mu2,sigma2)*(UL2-LL2)/2
+a2*a3/(2*(a1+a2)*(a2+a3))*(1-probnorm((k15-mu2)/sigma2))*(1-probnorm((k15-
mu3)/sigma3))*pdf('normal',k15,mu1,sigma1)*(UL1-LL1)/2
+a1*a2/(2*(a1+a2)*(a2+a3))*probnorm((k35-mu1)/sigma1)*probnorm((k35-
mu2)/sigma2)*pdf('normal',k35,mu3,sigma3)*(UL3-LL3)/2;
VUS[i,1]=(f1+f2)*(322+13*sqrt(70))/900+(f3+f4)*(322-
13*sqrt(70))/900+f5*128/225+a1*a3/(4*(a1+a2)*(a2+a3))*probnorm((mu3-
mu1)/sqrt(sigma1**2+sigma3**2));/********** wi f(xi) ************/
end;
print vus;
create vus.vus_validation from vus;
append from vus;
quit;

/*************** linear and quadratic modify ************************/
data vus.vus_orig_modify;
     set vus.vus_original;
     linear=(0.5*(col1-5/32))/(3/4-5/32)+0.5;
     quadratic=(0.5*(col1**0.5-(5/32)**0.5))/((3/4)**0.5-(5/32)**0.5)+0.5;
run;
```

```
data vus.vus_vali_modify;
    set vus.vus_validation;
    linear_v=(0.5*(col1-5/32))/(3/4-5/32)+0.5;
    quadratic_v=(0.5*(col1**0.5-(5/32)**0.5))/((3/4)**0.5-(5/32)**0.5)+0.5;
run;
/************** shrinkage and mean of shrinkage ********************/
data vus.shrinkage;
    merge vus.vus_orig_modify vus.vus_vali_modify(rename=col1=col_v);
    ori_shrinkage=col1-col_v;
    linear_shrinkage=linear-linear_v;
    quadratic_shinkage=quadratic-quadratic_v;
    keep ori_shrinkage linear_shrinkage quadratic_shinkage;
run;

proc means data=vus.shrinkage noprint;
    output out=vus.shrinkage_mean;
run;
data vus.shrinkage_mean;
    set vus.shrinkage_mean;
    if _n_=4;
    drop _freq_ _stat_ _type_;
run;
/************** export shrinkage and its mean as excel data
********************/
PROC EXPORT DATA= Vus.Shrinkage_mean
            OUTFILE= "C:\Dongmei Wang\Research\vus bootstrap\shrinkage_m
ean.xls"
            DBMS=EXCEL2000 REPLACE;
     SHEET="shrinkage_mean_boots";
RUN;

PROC EXPORT DATA= Vus.Shrinkage
            OUTFILE= "C:\Dongmei Wang\Research\vus bootstrap\shrinkage.xls"
            DBMS=EXCEL2000 REPLACE;
     SHEET="shrinkage_boots";
RUN;
```

APPENDIX 3.2 2-fold cross validation for the shrinkage of VUS

```
/***** 2-fold cross validation to calculate the shrinkage of VUS******/
libname class 'C:\Dongmei Wang\Research\vus bootstrap'; run;
```

```
libname cross 'C:\Dongmei Wang\Research\VUS cross';run;
/****************** bootstrap sample ***********************/
data class.mh_pop;/********* pop for mock-HSV1 **********/
      merge class.mh1(rename=(mh1-mh182=m1-m182))
class.mh2(rename=(mh1-mh182=h1-h182));
      drop _name_;
      id=_n_;
run;

data class.mc_pop;/********* pop for mock-HSV1 **********/
      merge class.mc1(rename=(mc1-mc182=m1-m182))
class.mc2(rename=(mc1-mc182=c1-c182));
      drop _name_;
      id=_n_;
run;

/***************** generate bootstrap sample *********************/
%macro sampling(data1,data2,data3,a,n);/** data1=population data2=sample**/

%let j=%sysevalf(&a*666+666);
      data one;
            set &data1;
            index=ranuni(&j);
      run;
proc sort data=one out=one;by index;run;
data &data2;
      set one;
      by index;
      if _n_<=&n/2;
      drop index;
run;
data &data3;
      set one;
      by index;
      if _n_>&n/2;
      drop index;
run;
%mend;

%macro calculate(data);/********** sign rank test ***************/
*ods trace on;
ods output    TestsForLocation=p;
proc univariate data=&data;run;
data p;
```

```
    set p;
    if Testlab="S" then output;
run;
%mend;

/************ find the common range ***************/
%macro range(data);
proc transpose data=&data out=&data prefix=v;run;
data one;
    merge &data class.p;
run;
data one;
    set one;
    if p1<=0.005;
    if p2<=0.005;
    drop p1 p2;
run;
proc transpose data=one out=one prefix=v;run;
%mend;

%macro mean_std(n);
data two;
    set one(keep=p inf);
    if inf=&n;
    drop inf;
run;
proc means data=two noprint;
    output out=mean;
run;
data mean;
    set mean;
    if _n_>=4;
    drop _stat_ _type_ _freq_;
run;
proc transpose data=mean out=mean prefix=v;run;
%mend;

data cross.percent_number;
    if 1=1 then delete;
run;
data cross.var_number;
    if 1=1 then delete;
run;
data cross.mean_std_mh;
```

```
    if 1=1 then delete;
run;
data cross.mean_std_mc;
    if 1=1 then delete;
run;
data cross.mean_std_mm;
    if 1=1 then delete;
run;
data cross.mean_std_mh_v;
    if 1=1 then delete;
run;
data cross.mean_std_mc_v;
    if 1=1 then delete;
run;
data cross.mean_std_mm_v;
    if 1=1 then delete;
run;


%macro repeat;
%do a=0 %to 99;

%sampling(class.mh_pop,mh1,mh2,&a,21);

data sample1; set mh1(keep=m1-m182 rename=(m1-m182=mh1-mh182));run;
data sample2; set mh1(keep=h1-h182 rename=(h1-h182=mh1-mh182));run;
data mh_sample1;/********** model ***********/
    set sample1 sample2;
run;

data sample1; set mh2(keep=m1-m182 rename=(m1-m182=mh1-mh182));run;
data sample2; set mh2(keep=h1-h182 rename=(h1-h182=mh1-mh182));run;
data mh_sample2;/************ validation **************/
    set sample1 sample2;
run;

/************ sample for mock-cox 18+18 ***************/
%sampling(class.mc_pop,mc1,mc2,&a,18);

data sample1; set mc1(keep=m1-m182 rename=(m1-m182=mc1-mc182));run;
data sample2; set mc1(keep=c1-c182 rename=(c1-c182=mc1-mc182));run;

data mc_sample1;/******** model ********/
    set sample1 sample2;
```

```
run;

data sample1; set mc2(keep=m1-m182 rename=(m1-m182=mc1-mc182));run;
data sample2; set mc2(keep=c1-c182 rename=(c1-c182=mc1-mc182));run;

data mc_sample2;/******** validation ********/
    set sample1 sample2;
run;

/************* sample for mock-mock 21 ***************/
%sampling(class.inner_m_m,mm_sample1,mm_sample2,&a,21);

%calculate(mh_sample1);/************* sign rank sum test ************/
data p1;set p(keep=pvalue rename=(pvalue=p1));run;
%calculate(mc_sample1);
data p2;set p(keep=pvalue rename=(pvalue=p2));run;

data class.p;
    merge p1 p2;
run;

%range(mh_sample1);/*************** find the common range
*********************/
data mh1;
    set one;
    inf=2;
run;
%range(mh_sample2);
data mh2;
    set one;
run;

%range(mc_sample1);
data mc1;
    set one;
    inf=1;
run;
%range(mc_sample2);
data mc2;
    set one;
run;

%range(mm_sample1);
data mm1;
```

```
    set one;
    inf=0;
run;
%range(mm_sample2);
data mm2;
    set one;
run;

data depend;/************** count the number of variables used *************/
    set class.p;
    retain p 0;
    if p1<=0.005 & p2<=0.005 then p=p+1;
    call symput("counterxzero",p);
run;
%let counter=%sysevalf(&counterxzero+0);

data a; a=&counter; run;
data cross.var_number;
    set cross.var_number a;
run;

data pls;
    set mh1 mc1 mm1 mh2 mc2 mm2;
run;

ods output    PercentVariation=percent;/************** count the percentage
**************/
proc pls data =pls;
    model inf=v1-v&counter/solution;
output out=no PREDICTED=no;
run;

data percent;
    set percent(keep=TotalYVariation rename=(TotalYVariation=y));
    retain p 0;
    if y<=94.5 then p=p+1;
    call symput("percent",p);
run;
%let percentnumber=%sysevalf(&percent+0);
/***************** record number of components ***********************/
data p; p=&percentnumber;run;

data cross.percent_number;
    set cross.percent_number p;
```

```
run;
/*****************************************************************
******/
proc pls data =pls nfac=&percentnumber;/*pls is selected intra*/
      model inf=v1-v&counter;
output out=one PREDICTED=p;
run;

%mean_std(2); /************* mean std for VUS of bootstrap data
***************/
data cross.mean_std_mh;
      set cross.mean_std_mh mean;
run;
%mean_std(1);
data cross.mean_std_mc;
      set cross.mean_std_mc mean;
run;
%mean_std(0);
data cross.mean_std_mm;
      set cross.mean_std_mm mean;
run;

data one; set one; if _n_>48;run;
data one;
      set one;
      if _n_<=22 then inf=2;
      if _n_<=40 & _n_>22 then inf=1;
      if _n_>40 then inf=0;
run;

%mean_std(2); /************* mean std for VUS ***************/
data cross.mean_std_mh_v;
      set cross.mean_std_mh_v mean;
run;
%mean_std(1);
data cross.mean_std_mc_v;
      set cross.mean_std_mc_v mean;
run;
%mean_std(0);
data cross.mean_std_mm_v;
      set cross.mean_std_mm_v mean;
run;

%end;
```

```
%mend;
%repeat;
```