Project 1 - Microarray Analysis Replicating Marisa et. al.

Konrad Thorner, Aishwarya Deengar, Jia Liu, Morgan Rozman, Marzie (TA)

Introduction

Colon cancer has been hard to study in the past because no gene signature has been successful in estimating the prognosis of the disease. A major challenge in determining gene signature was the heterogeneity in the cancer cells due to presence of distinct molecular entities which develop via different pathways depending on the molecular features. The Marisa et. al. study uses microarray data to provide a molecular classification based on expression of mRNA [1]. The goal of this study is to provide a new method of classification of colon cancer based on transcriptome analysis. This improves disease stratification by allowing the use of common DNA markers and clinical variables not previously used. Genome-wide mRNA expression analysis was performed and analysed using a large number of multicentre and extensively characterised samples. The associations between the molecular subtypes and its resultant pathology, DNA alterations and the prognosis was studied. Finally, to confirm the results thus obtained a large independent set was used to validate the molecular classification derived.

Data

The original study included 750 French patients whose tumors were collected at seven different samples between 1987 and 2007 [1]. Patients who had previously received preoperative chemotherapy or radiation therapy, as well as patients with primary rectal cancer were excluded for homogeneity. There were 566 samples that met RNA quality requirements based on guidelines laid out in [2]. These requirements include a 28s/18s ratio above 1.8 for microarray analysis to rule out degradation. No sources of error or contamination were discussed in the original study.

Our work includes data from 134 of the samples that were determined to be of high quality. Gene expression profiles for each of the samples were determined on Affymetrix U133 Plus 2.0 chips. This microarray data provides expression information on specific genes for each colon cancer tumor sample. There are a total of 54,675 genes included in analysis. Though the original study classified their full sample set into six different subtypes, the samples in the set used in our analyses are exclusively from C3 and C4 tumor subtypes. The original data is available on the NCBI Gene Expression Omnibus (http://www.ncbi.nlm.nih.gov/geo/; accession number GSE39582). We downloaded the 135 samples of interest in our analyses via .CEL files from this public repository.

Methods

1.1 Data normalization

Accounting for technical variation between microarray, package affy(1.64.0) is used to convert probe level data to expression values. Typically, it can read in probe level data, and then correct background. Here, we use this package to read array and normalize data. First, it uses rma() to do normalization and probe specific background correction, for example, subtracting 'intensity value' read from the mis-matches (the probes having one base mismatch with the target sequence intended to account for non-specific binding). Besides that, the package can summarize the probe set values into one expression measure and, in some cases, a standard error for this summary.

ReadAffy() is used to read the array data. For this project, there are 1164 rows and 1164 columns in the dataset. The function rma() (Robust Multi-Array Average Expression Measure) meanwhile is used for background subtraction, quantile normalization and summarization (via median-polish) [3]. Before and after the normalization, the datasets are both S4 type, but the numbers of rows and columns change to 54675 features and 134 samples. So, the function rma() not only normalizes the data for different backgrounds, but also converts AffyBatch object into ExpressionSet object and normalizes them together. It implements probe specific correction of the PM probes by using a model based on observed intensity being the sum of signal and noise. It also uses quantile normalization to do normalization of corrected PM probes. Then, it calculates expression measure by using median polish.

2.1 Compute RLE and NUSE

Package affyPLM(1.62.0) is used to compute Relative Log Expression (RLE) and Normalized Unscaled Standard Errors (NUSE) scores of the microarray samples [4]. It extends and improves the functionality of the base affy package. The focus of affyPLM is on implementation of methods for fitting probe-level models and tools.

The function fitPLM() in the affyPLM package converts AffyBatch into PLMset by fitting a specified robust linear model to the probe level data. The usage of fitPLM() is very flexible, and it has many parameters. In this project, the normalized data is the object with parameters ‘normalize’ and ‘background’ equal to true. In other words, keep models and other parameters as default parameters. This is done because we want to normalize data using quantile normalization, and correct background with RMA background correction. To get a general idea of the output, a boxplot of the results is plotted (Fig1). The median of each sample appears to lie on a straight line, indicating the data has been normalized and the background has been corrected.

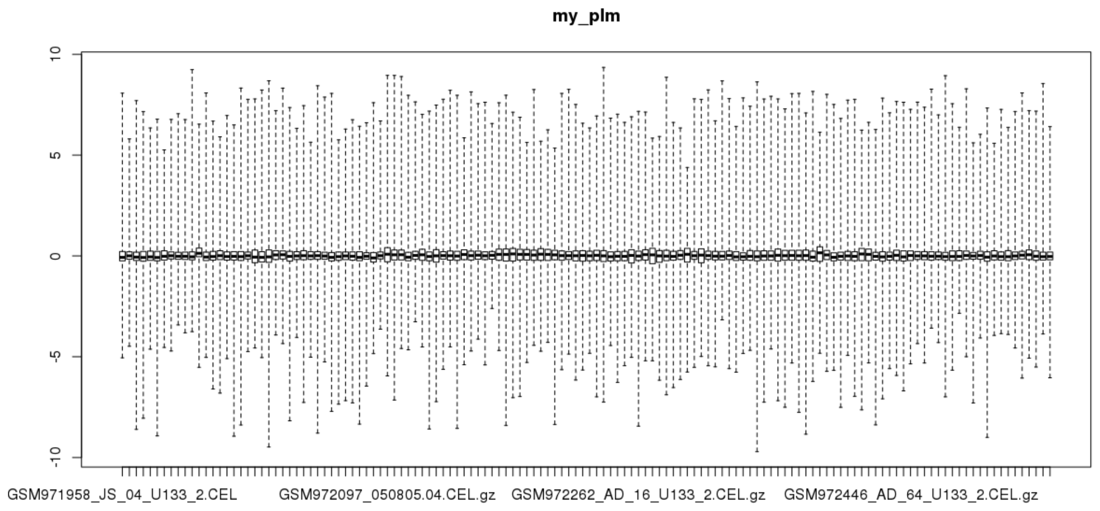


Fig1. boxplot of PLMset

Next, the function RLE (Relative Log Expression) serves as a quality assessment tool. Using this function, RLE values can be computed for each probeset by comparing the expression value on each array against the median expression value for that probeset across all arrays. The goal is to compute the median RLE for each sample, then examine the distribution by plotting them in a histogram. RLE() outputs a matrix where columns are the samples and the first row contains the medians. The medians are selected and passed to hist() to produce the final histogram (Fig3). The same steps are followed for plotting the median of every Normalized Unscaled Standard Errors (NUSE) value, another mathematical method for quality assessment. (Fig4).

2.2 Correct for Batch Effects

Package sva(3.34.0) is used to correct the dataset returned from the rma() function for batch effects. The sva package can be used to remove artifacts in multiple ways. It can identify and estimate surrogate variables for unknown sources of variation in high-throughput experiments. But it can also directly remove known batch effects using ComBat. In this project, we use the ComBat() function to correct the dataset.

The ComBat function adjusts for known batches using an empirical Bayesian framework [5]. In order to use the function, the batch variables in the dataset are needed. This is imported by using the package readr. This package provides an easy way to read rectangular data, in this case the dataset ‘proj_metadata.csv’ containing ‘normalizationcombatbatch’, with ‘read_csv()’. In addition, the normalized data should be a matrix, which is done with the function ‘exprs()’. ‘exprs()’ can access the expression and error measurements of assay data stored in an object derived from the eSet-class. Finally, the parameter mod is set with ‘model.matrix(~normalizationcombatmod, data = proj_metadata)’. Function ‘model.matrix()’ creates a model matrix for the adjustment variables, including the variable of interest.

After getting parameters dat, batch and mod, the function ‘ComBat()’ is used to produce the final normalized data. The expression data is written to a CSV file, where probesets are rows and samples are columns.

2.3 Principal Component Analysis

Principal Component Analysis (PCA) is performed on the normalized data using the prcomp() function. PCA is a type of linear transformation on a given data set that has values for a certain number of variables (coordinates) for a certain amount of spaces. This linear transformation fits this dataset to a new coordinate system in such a way that the most significant variance is found on the first coordinate, and each subsequent coordinate is orthogonal to the last and has a lesser variance. In this way, one can transform a set of x correlated variables over y samples to a set of p uncorrelated principal components over the same samples.

It is important to first center and scale the data when performing PCA. To scale within each gene rather than within each sample, the first step is to transpose the expression matrix using t(), and then use the function ‘scale()’ to actually center and scale the dataset. In the ‘scale()’ function, there are two parameters, center and scale, both of which are set to true. For the centering, this means the column means (omitting NAs) are subtracted from their corresponding columns. And for scaling, this means the now centered columns of data are divided by their standard deviations. After scaling, the dataset should be re-transposed as before. The ‘prcomp()’ function is then used to perform PCA, with parameters center and scale set to false, returning an object of class prcomp [6]. The attribute ‘rotation’ contains values for each of the principal components, which can be selected and used to plot the PCA, specifically PC1 VS. PC2.

Besides affy, affyPLM and sva packages, some other packages are also used in this project. First, the AnnotationDbi package(1.48.0). The main target of this package is to imply a user-friendly interface for querying SQLite-based annotation data. And the package hgu133plus2.db(3.2.3) which stands for Affymetrix Human Genome U133 Plus 2.0 Array annotation data.

2.4 Noise filtering and dimensionality reduction

The next step in the analysis begins with reading in the RMA normalized, Combat adjusted gene expression data into R. It takes the form of a data table with probeset IDs and sample names corresponding to the rows and columns.

Three filters are employed to remove noise and reduce the total amount of genes to be analyzed. These are:

1. At least 20% of the samples for each gene have an expression value > log2(15)
2. The variance for each gene is significantly different from the median variance ($p < 0.01$)
3. The coefficient of variance for the gene is > 0.186

In each case, apply() is used to pass each row of the expression matrix into a function and return an array of boolean values. These values correspond to whether a given gene passes the filter, and the expression matrix can then be subset to keep only the genes of interest. With each new filter, it is applied to the previously filtered matrix until only genes that pass all three filters remain.

Filter 1 loops through each row and column, tracking how many samples are above log2(15). It returns the count divided by the total number of samples to indicate genes where 20% of samples pass the threshold.

Filter 2 begins by looping through each row of the gene expression matrix, calculating the standard deviation, and returning the median. The chi-square test is then employed to find genes with significantly different variance from this median. This involves calculating the degrees of freedom, the test statistic, and the upper critical value given an alpha of 0.01. It returns whether the test statistic is greater than the critical value.

Finally, filter 3 calculates the coefficient of variation (the ratio of standard deviation to mean) and checks if it is greater than 0.186.

2.5 Hierarchical clustering and subtype discovery

Using the filtered gene expression matrix, hierarchical clustering was performed. The objective is to create groupings of our samples based on similarities in gene expression. This is a three-step process that can be handled with R's default packages.

The function dist() takes the matrix and returns an object described as a "dissimilarity matrix". It's important that the transverse of the matrix is used so the clustering applies to the samples. The return value is used as the argument for the hclust() function, which will produce a dendrogram or tree that represents the clustering. The default "complete" method is chosen to cluster based on similarity. Lastly, cutree() takes the output of hclust() and a number of groups, in this case k=2, and returns a vector assigning each of the samples to one of the groups. This is used to create two separate matrices for each cluster.

Heatmap construction is done entirely through the heatmap.2(), a more fully featured function within the gplots package. The provided annotation matrix was read in to access the C3/C4 information in the cit-coloncancermolecularsubtype field. The subtype is determined for each sample and a vector of color values corresponding to them is constructed. This serves as the value of the optional argument “ColSideColors”, which aids the interpretation.

A Welch’s t-test is then run between our two data sets from clusters 1 and 2 using t.test(). This serves to compare each gene’s mean expression and determine where there are significant differences. The t-statistics and p-values are selected from the results and aggregated into a data table, which includes a column for adjusted p-values. This is found using the function p.adjust(), which takes the unadjusted p values and a specified method, in this case “fdr”. The purpose of adjusting them is to reduce the false discovery rate.

To find the genes of greatest significance, the data table of t-test results is filtered in a similar manner to before. The genes with an adjusted p-value below 0.05 are kept and are organized in ascending order of adjusted p-value.

2.6 In-depth analysis

To read the upregulated and downregulated genes, the fread() function was used to read the ttest.csv file containing the genes, their symbols, p value, adjusted p value and t statistic.To extract top 1000 genes, sort() function was used and finally to get the top 10 upregulated and downregulated genes , head() and tail() functions were used respectively.

Next, Bioconductor package “GSEABase” was installed to load and study the genesets downloaded. getGmt() function was used to read each of the three gene sets, i.e., KEGG, GO and Hallmark. Next, the number of genesets were determined using the length() function. To conduct the Fisher test a function was written which computed the hypergeometric statistics and p-values to compare overlap for each gene set and each gene list. The Fisher test was then performed using the given function fisher.test(). Then, a table of statistics for each of the comparisons was created (csv format), including gene set name, statistic estimate and p-value. The p-values for multiple hypotheses were adjusted using the Benjamini-Hochberg (FDR) procedure and then appended to the data frame. The data frames were sorted by the nominal p-value and the top three results of each are reported in the result section, as are the number of significantly enriched gene sets for KEGG, Hallmark and GO when $p < 0.05$.

3. Evaluation

The analysis runs smoothly and requires a few minutes to run. Some functions like ‘rma()’ and ‘t()’ are written in C, so they have high speed and efficiency. The package affyPLM extends and improves the functionality of the base affy package. Routines that make heavy use of compiled code for speed. In this project, Bioconductor, RStudio and SCC have been used to install R packages and run the script.

Results

RLE and NUSE are key steps for quality assessment to identify aberrant samples, and the results are visualized in Figs 2-4. The boxplot in Fig2 shows the quality of the dataset is variable as some medians of RLE values stray from the line, while in Fig3, RLE medians of about 90 samples are concentrated between -0.05 to 0.05. Assuming that most genes are not changing in expression across arrays, RLE values will be near 0, so this signifies good quality. But many samples remain outside this range with lesser quality. In the case of NUSE, the standard error estimates obtained for each gene on each array from fitPLM are taken and standardized across arrays so that the median standard error for the gene is 1 across all arrays. This process accounts for differences in variability between genes, such that medians close to 1 indicate high quality. In Fig4, some samples are not very close to 1, for example, the long tail after 1.04. But, the majority of the data is within acceptable limits.

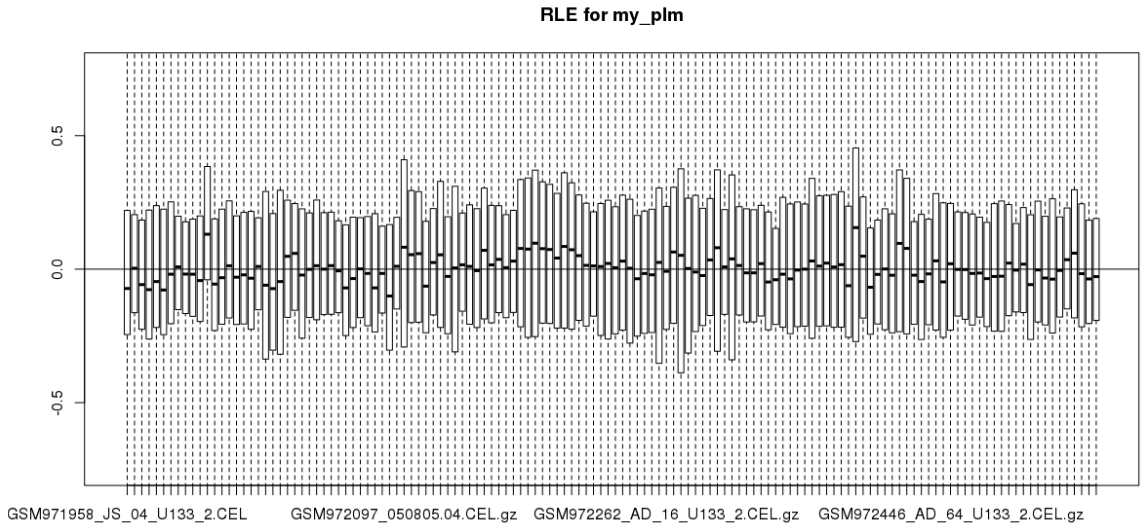


Fig2. Boxplot of RLE values

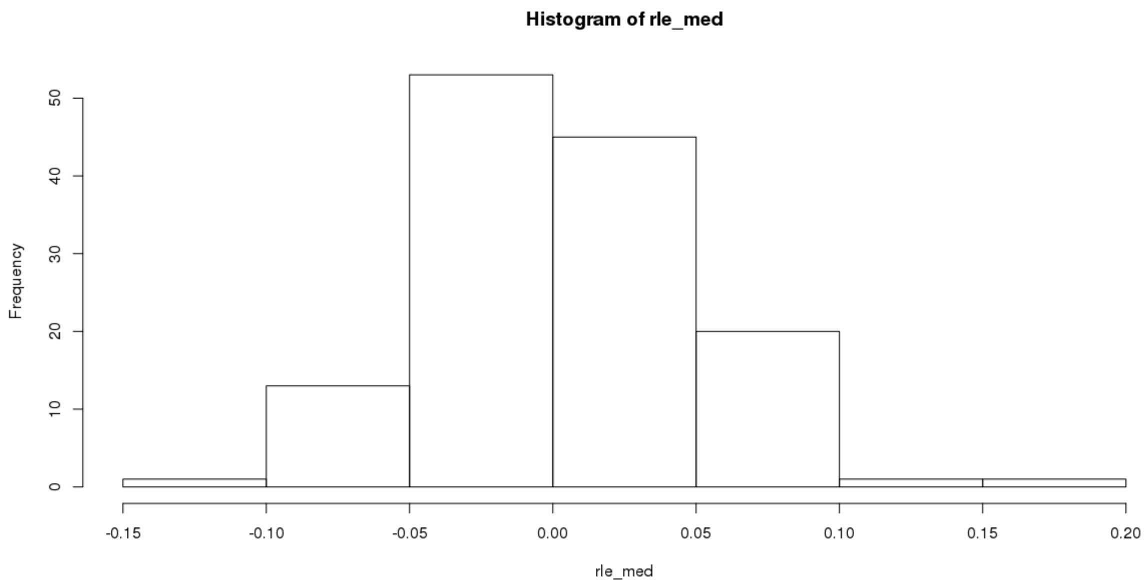


Fig3. Histogram of RLE medians

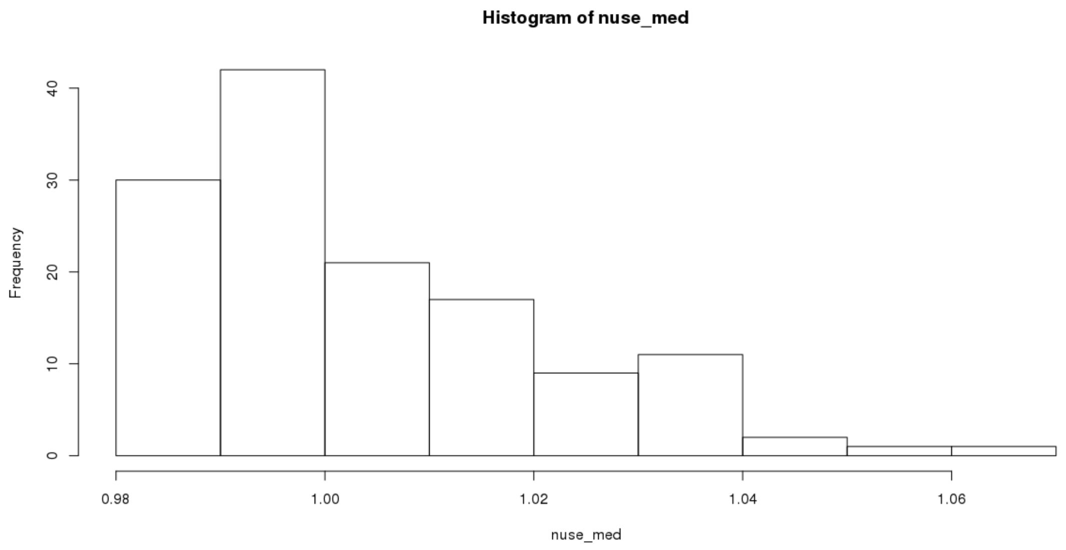


Fig4. Histogram of NUSE medians

The gene expression data was then transformed with PCA to visualize the variance and find possible outliers. Fig5 shows PC1 VS. PC2. However, the sample distribution in Fig5 is not concentrated enough to form clusters. So, other parameters are tried in PCA plots, namely PC2 VS. PC3 (Fig6), and PC1 VS. PC3 (Fig7).

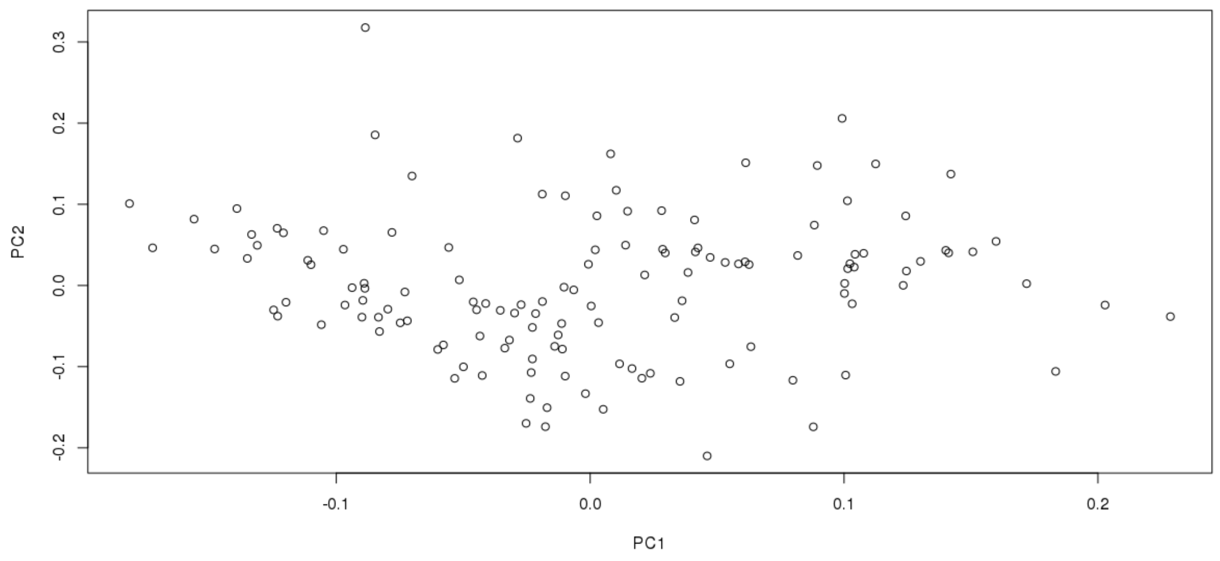


Fig5. PC1 VS. PC2

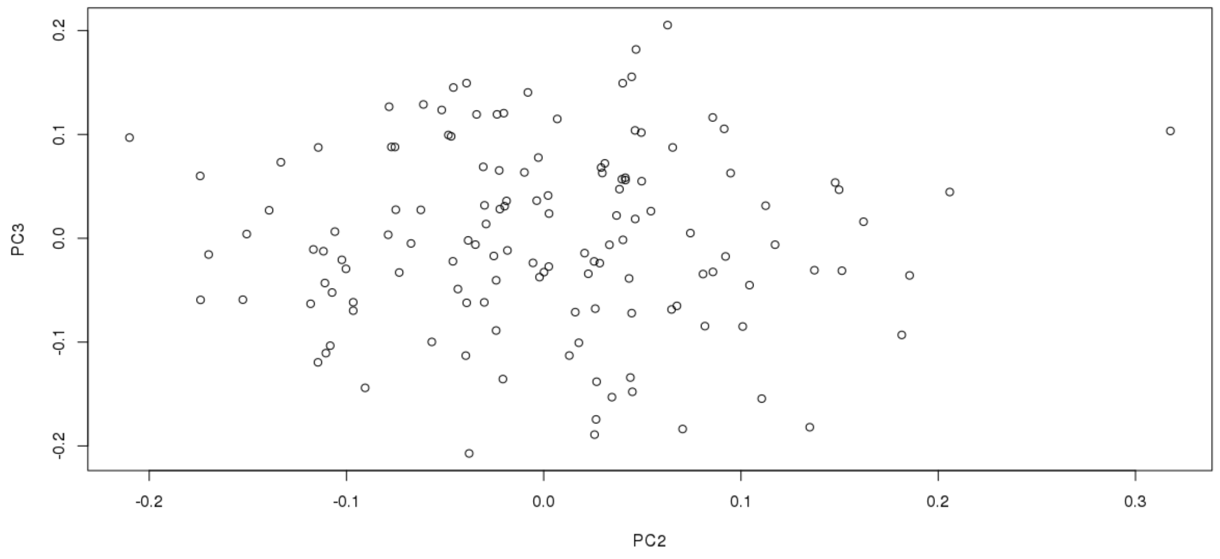


Fig6. PC2 VS. PC3

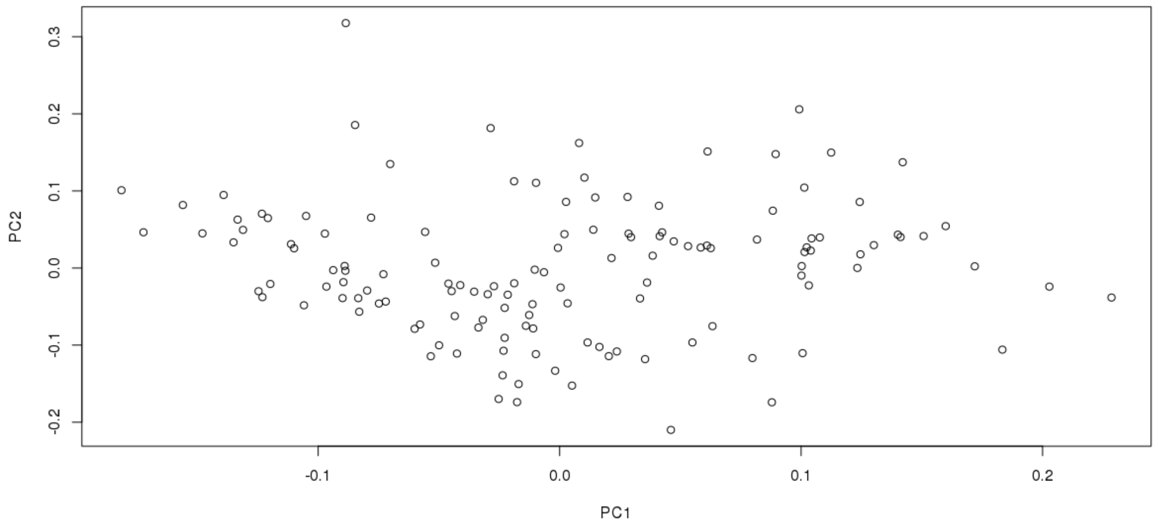


Fig7. PC1 VS. PC3

Despite this, the results are not sufficient enough to describe the dataset. So, a threshold is set to perform removal. Here, two thresholds were tried on the scaled data that results from the ‘prcomp()’ function. Rows are selected which do not have values less than either 2 or 1, and 2 is found to return a larger set of data. ‘prcomp()’ is again used with the same parameters as before and PC1 VS. PC2 is plotted as seen in Fig8. The result seems better than without using the threshold, and the samples are divided into two clusters. PC1 VS. PC3 is also plotted in Fig9. The distribution in Fig8 is still more concentrated, so we conclude that PC1 VS. PC2 contains more readable information.

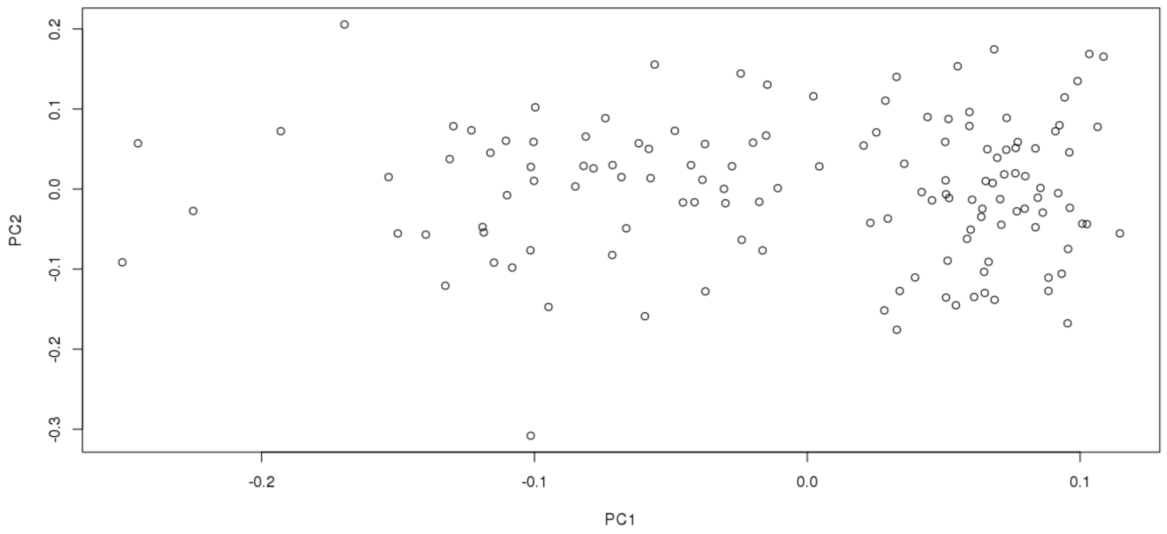


Fig8. PC1 VS. PC2 after remove

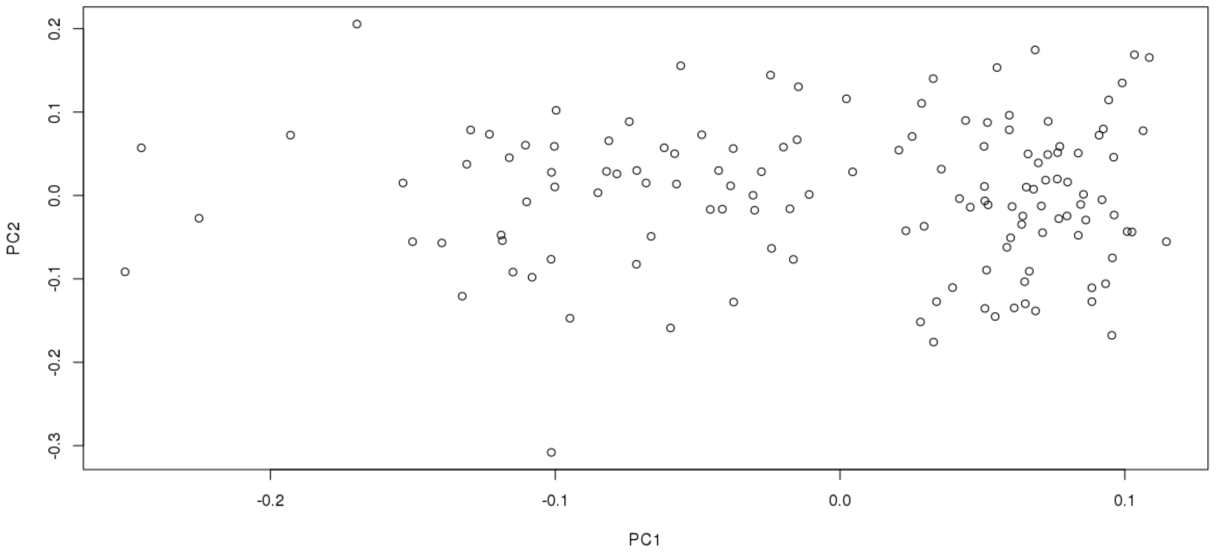


Fig9. PC1 VS. PC3 after remove

Proceeding from the initial processing, the gene expression data still has “noise” that makes further analysis difficult. Using the previously described filtering, 54,675 probesets are reduced to 1531. From that matrix, hierarchical clustering identifies similarities between the 134 samples and divides them into two clusters of 57 and 77. Welch’s t-test is subsequently used to find differential gene expression for all 1531 genes between clusters 1 and 2. The number of genes that pass the adjusted p-value threshold of 0.05 is 1236.

According to Marisa *et al.*, subtypes were defined using the top five upregulated and downregulated genes. Following similar logic, the genes were sorted and the five with the lowest adjusted p-values (the most significant) were selected (Fig10). These all corresponded to comparisons with a positive t-statistic, indicating gene expression was higher in cluster 1. Alternatively, those with the lowest adjusted p-values but negative t-statistics should have higher gene expression in cluster 2.

Cluster 1		Cluster 2	
Probeset ID	Adjusted p-value	Probeset ID	Adjusted p-value
204457_s	6.784208e-46	203240	3.989731e-28
209868_s	2.942653e-44	220622	8.997583e-25

223122_s	5.440165e-44	210107	1.702844e-24
225242_s	1.187888e-43	238750	6.113043e-24
202291_s	2.784627e-43	204673	9.795522e-24

Fig10. Table of most differentially expressed genes for each cluster, in order of significance.

Next, a heatmap for all probesets and samples was constructed to provide a broad measure of differential gene expression between subtypes (Fig11). The columns represent the samples, where red corresponds to the C3 subtype and blue corresponds the C4 subtype. Each row represents a different gene or probeset, and each axis also has an associated dendrogram.

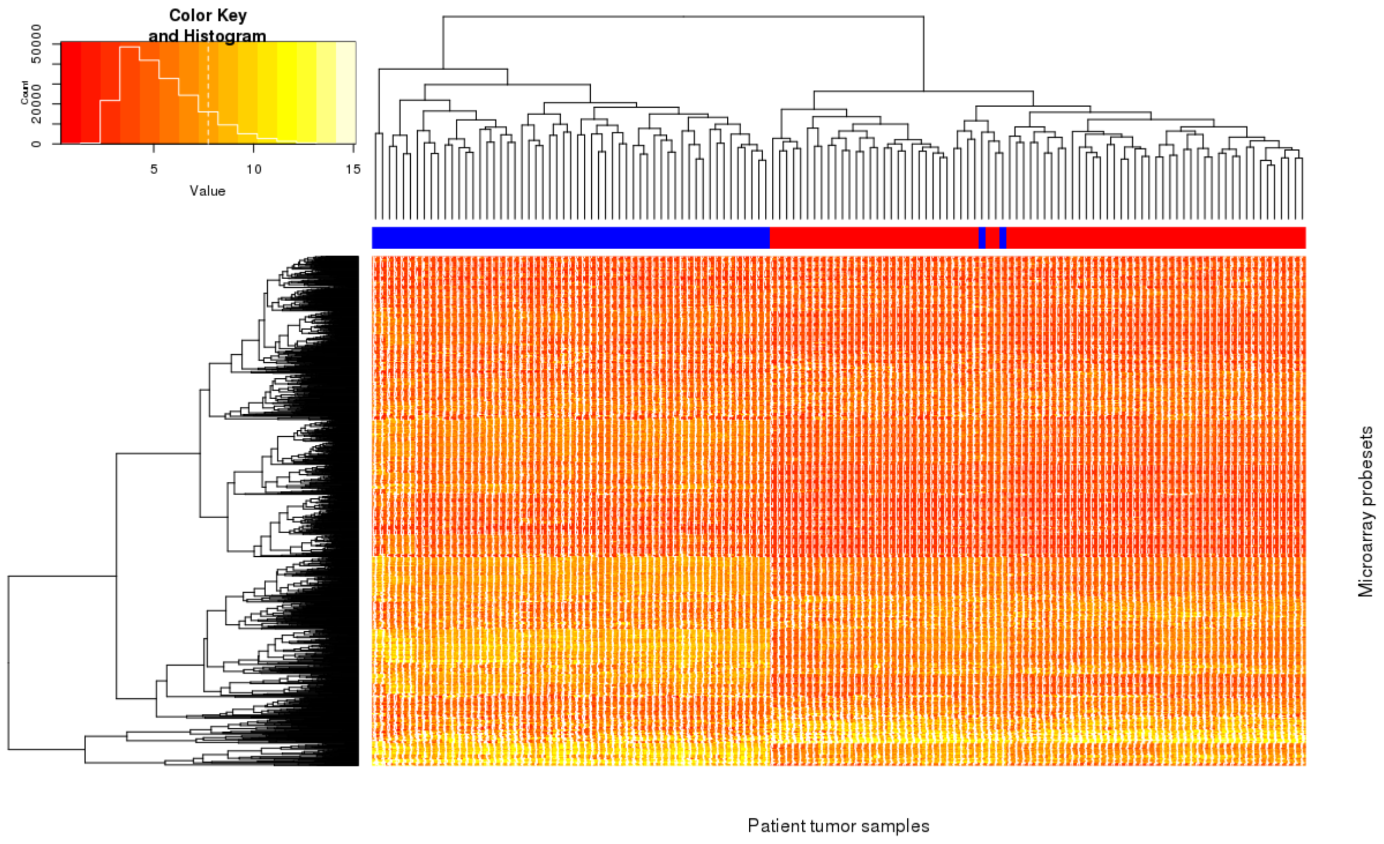
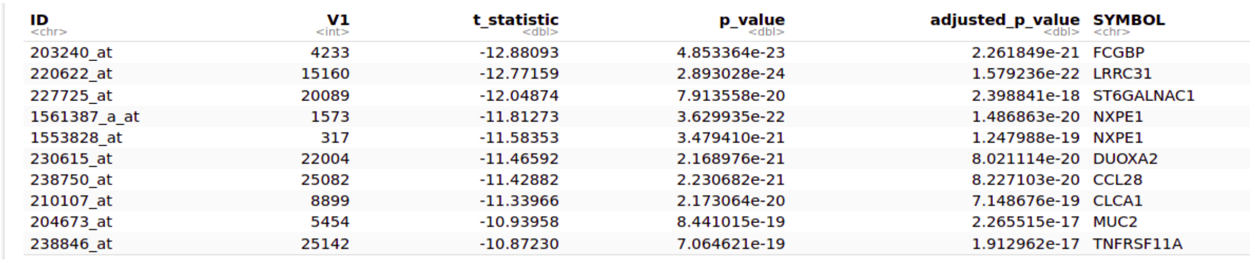


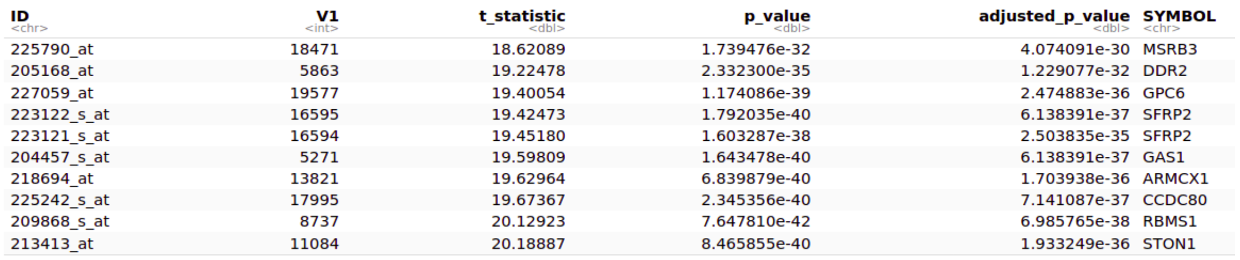
Fig11. Heatmap of gene expression across C3 (red) and C4 (blue) subtypes for colon cancer samples

Finally, to connect the microarray results to actual biological processes, the probesets should be associated with actual genes using gene set enrichment analysis. The top 10 up- and down-regulated probesets on the basis of the t-statistics were found as shown in Fig11 and Fig12.



ID	V1	t_statistic	p_value	adjusted_p_value	SYMBOL
<chr>	<int>	<dbl>	<dbl>	<dbl>	<chr>
203240_at	4233	-12.88093	4.853364e-23	2.261849e-21	FCGBP
220622_at	15160	-12.77159	2.893028e-24	1.579236e-22	LRRC31
227725_at	20089	-12.04874	7.913558e-20	2.398841e-18	ST6GALNAC1
1561387_a_at	1573	-11.81273	3.629935e-22	1.486863e-20	NXPE1
1553828_at	317	-11.58353	3.479410e-21	1.247988e-19	NXPE1
230615_at	22004	-11.46592	2.168976e-21	8.021114e-20	DUOXA2
238750_at	25082	-11.42882	2.230682e-21	8.227103e-20	CCL28
210107_at	8899	-11.33966	2.173064e-20	7.148676e-19	CLCA1
204673_at	5454	-10.93958	8.441015e-19	2.265515e-17	MUC2
238846_at	25142	-10.87230	7.064621e-19	1.912962e-17	TNFRSF11A

Fig12. Table showing the top 10 downregulated genes with gene symbol, t-statistic, nominal p-value, and adjusted p-value columns



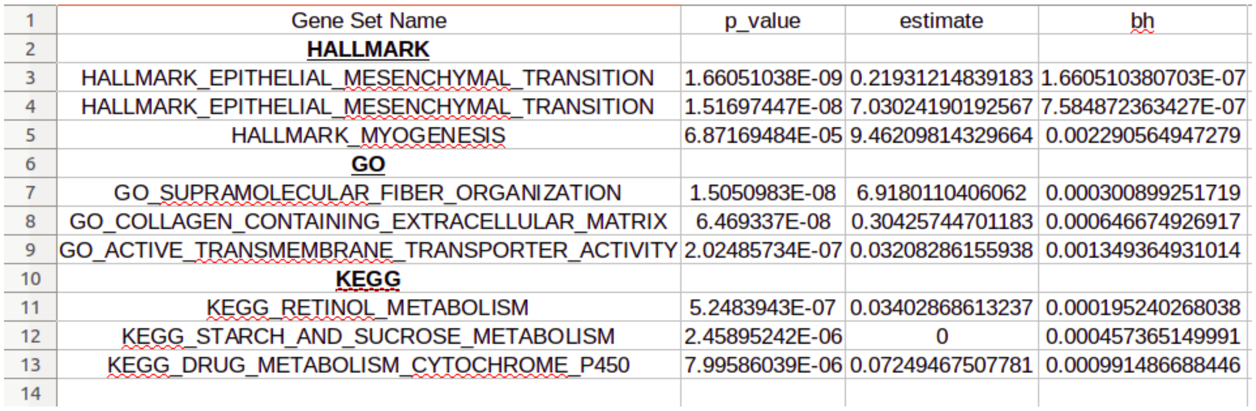
ID	V1	t_statistic	p_value	adjusted_p_value	SYMBOL
<chr>	<int>	<dbl>	<dbl>	<dbl>	<chr>
225790_at	18471	18.62089	1.739476e-32	4.074091e-30	MSRB3
205168_at	5863	19.22478	2.332300e-35	1.229077e-32	DDR2
227059_at	19577	19.40054	1.174086e-39	2.474883e-36	GPC6
223122_s_at	16595	19.42473	1.792035e-40	6.138391e-37	SFRP2
223121_s_at	16594	19.45180	1.603287e-38	2.503835e-35	SFRP2
204457_s_at	5271	19.59809	1.643478e-40	6.138391e-37	GAS1
218694_at	13821	19.62964	6.839879e-40	1.703938e-36	ARMCX1
225242_s_at	17995	19.67367	2.345356e-40	7.141087e-37	CCDC80
209868_s_at	8737	20.12923	7.647810e-42	6.985765e-38	RBMS1
213413_at	11084	20.18887	8.465855e-40	1.933249e-36	STON1

Fig13. Table showing the top 10 upregulated genes with gene symbol , t-statistic, nominal p-value, and adjusted p-value columns

Next, the following three gene set collections were used for the analysis:

1) names: KEGG gene set (186 total)
   unique identifiers: ACSS2, GCK, ..., EIF4G1 (5241 total)
   Length of gene set: 186
2) names: GO gene set (9996 total)
   unique identifiers: CDK9, CHD1, ..., ANXA2R (18974 total)
   Length of gene set: 9996
3) names: HALLMARK gene set (50 total)
   unique identifiers: JUNB, CXCL2, ..., SRP14 (4384 total)
   Length of gene set: 50

The data frames were sorted by the nominal p-value and the top three results of each are reported below:



Gene Set Name	p_value	estimate	bh
HALLMARK			
HALLMARK_EPITHELIAL_MESENCHYMAL_TRANSITION	1.66051038E-09	0.21931214839183	1.660510380703E-07
HALLMARK_EPITHELIAL_MESENCHYMAL_TRANSITION	1.51697447E-08	7.03024190192567	7.584872363427E-07
HALLMARK_MYOGENESIS	6.87169484E-05	9.46209814329664	0.002290564947279
GO			
GO_SUPRAMOLECULAR_FIBER_ORGANIZATION	1.5050983E-08	6.9180110406062	0.000300899251719
GO_COLLAGEN_CONTAINING_EXTRACELLULAR_MATRIX	6.469337E-08	0.30425744701183	0.000646674926917
GO_ACTIVE_TRANSMEMBRANE_TRANSPORTER_ACTIVITY	2.02485734E-07	0.03208286155938	0.001349364931014
KEGG			
KEGG_RETINOL_METABOLISM	5.2483943E-07	0.03402868613237	0.000195240268038
KEGG_STARCH_AND_SUCROSE_METABOLISM	2.45895242E-06	0	0.000457365149991
KEGG_DRUG_METABOLISM_CYTOCHROME_P450	7.99586039E-06	0.07249467507781	0.000991486688446

Fig 14: Table showing the sorted list of top three nominal p values of each geneset- Hallmark, KEGG and GO.

The number of significantly enriched genesets for KEGG, Hallmark and GO when $p < 0.05$ are:

1) KEGG- 32 genesets
2) HALLMARK -22 genesets
3) GO - 645 genesets

Discussion

The study concluded 6 molecular subtypes of prognostic value which could be segregated by clinico-biological characteristics and validated by 9 datasets. Using anatomo-clinical factors and common DNA alterations reliable information for highlighting subtype characteristics could be determined. Further analysis enabled the authors to determine the relevant deregulated pathways and their biological relevance. The subtypes concluded from the study are:

1) Serrated Precursor Neoplasia
    a) C2 (dMMR)
    b) C3 (KRASm)
    c) C4 (CSC)
    d) C6 ($\text{CIN}_{\text{NORML}}$)
2) Conventional Precursor Neoplasia
    a) C1 ($\text{CIN}_{\text{ImmuneDown}}$)
    b) C5 ($\text{CIN}_{\text{WntUp}}$)

For our attempt at replicating this analysis, we worked with one part of the study involving the C3 and C4 subtypes. Beginning with the data preprocessing, the challenge was concentrated in the principle component analysis. It is difficult to find a suitable threshold or cutoff to cluster all the data. If we perform PCA on unfiltered data, the result is not readable enough, in other words, no clear clusters can be found in the plot. If we set threshold or filter low value data, we are not certain whether these processes could influence the result.

Looking back at how the gene expression matrix was filtered in Methods 2.4, there are some aspects to consider. Genes filtered on the basis of high variance compared to all other genes might be indicative of genes related to the cancer phenotype of interest rather than housekeeping genes for example. The coefficient of variation filter on the other hand reflects the variation across samples for a single gene. Greater variation may suggest a gene that would be relevant when finding differences between samples during hierarchical clustering.

One observation from the table of probe sets in each cluster (Fig10) is that adjusted p-values were much lower for cluster 1 than cluster 2. This suggests that the most highly differentially expressed genes occurred in this cluster, but this would be subject to change with the addition of more clusters or subtypes.

As for the heatmap seen in Fig11, the density of data means at best, general patterns in gene expression between the two clusters can be commented on. It appears that for a majority of the genes assessed, the red C3 samples had higher expression levels than their blue C4 counterparts. Because this heatmap was constructed separately from the clustering and t-tests, we cannot determine if the red and blue subtypes correspond to the two clusters, although a relationship is likely.

When the geneset of our data and that of the paper being studied was compared, KEGG_Starch_and_Sucrose_metabolism was the only one that matched while none of the genesets of GO overlapped. KEGG Starch and Sucrose Metabolism represents a crucial pathway regulating carbohydrate metabolism.

Conclusion

In our efforts to reproduce the results from Marisa et. al, we began with the raw Affymetrix data and applied normalization and correction for batch effects. The probesets were filtered to those of greatest relevance, allowing the samples to be classified into one of two subtypes. There were clear patterns of differential gene expression between them, genes that were then identified through gene set enrichment analysis. There was minimal overlap with those of the paper, but there was agreement on the role of KEGG starch and sucrose metabolism.

With this data, the overarching goal is to help develop prognostic gene signatures of the second and third stage of colorectal cancer, and help identify specific targets for future drug development. The study conducted was retrospective, and our group only used a small subset of the total samples, so the significance and robustness of the subtype classification requires further confirmation using a larger prospective patient cohort.

References

1. Marisa et al. Gene Expression Classification of Colon Cancer into Molecular Subtypes: Characterization, Validation, and Prognostic Value. PLoS Medicine, May 2013.

2. de Reynie`s A, Assie ́ G, Rickman DS, Tissier F, Groussin L, et al. Gene expression profiling reveals a new classification of adrenocortical tumors and identifies molecular predictors of malignancy and survival. J Clin Oncol 27: 1108–1115, 2009.
3. Rafael. A. Irizarry, Benjamin M. Bolstad, Francois Collin, Leslie M. Cope, Bridget Hobbs and Terence P. Speed. Summaries of Affymetrix GeneChip probe level data Nucleic Acids Research 31(4):e15, 2003.
4. Bolstad BM. Low Level Analysis of High-density Oligonucleotide Array Data: Background, Normalization and Summarization. Ph.D. thesis, University of California, Berkeley, 2004.
5. W.E. Johnson, C. Li, and A. Rabinovic. Adjusting batch effects in microarray data using empirical bayes methods. Biostatistics, 8(1):118–127, 2007.
6. Becker, R. A., Chambers, J. M. and Wilks, A. R. The New S Language. Wadsworth & Brooks/Cole, 1988.