Connecting Nutrition, Health and Environment

Wiktor Jurkowski, Perla Troncoso Rey, Earlham Institute

25 - 26 January 2017

Contents

List of Figures

List of Tables

1   Introduction

In this practice, we will look at how to explore gene expression data (which could be extracted from microarray or RNA sequencing data). In the first part of this practical session we will see general techniques to explore the patterns or structure of the data using Principal Component Analysis, PCA, and Hierarchical Clustering. We will then look into compiling an interaction network using an online resource and how to visualise the network using Cytoscape [2] [3].

In the second part, we will look at ways to rank genes (and/or metabolites) using approaches based on logistic regression. We will then perform pathway analysis using those rankings.

We will use a publicly available data from a study on fatty liver disease of obese and lean human subjects [6].

Part I
Part I

1.1   Exploring the expression data

One could obtain gene expression from microarrays or RNA sequencing data. It is not within the scope of this session to look at the details of obtaining quantifying the expression of genes from microarrays or RNA sequencing data but instead we will start our analysis assuming that expression data has been quantified. The gene expression data is normally stored in tabular file, representing a matrix where the columns are the samples or experiments, and the rows represent the genes.

Example of expression data is shown in Figure 1. The table shows the expression of six genes in four different experiments or samples. This is, gene A has expression of 0.1 for sample 1, 0.8 for sample 2, 0.3 for sample 3, and so forth.

Figure 1: Example of expression data with samples across the columns and individual genes down the rows.

Expression data can be obtained using different algorithms. One of the most well know are TopHat and Cufflinks protocol for the analysis of RNA sequencing data, which includes quantification of gene expression. Table 1 shows an example of the output provided by cufflinks with the estimated gene-level expression values. Cufflinks uses the notation "XLOC_*numeric_sequence*" to identify a gene.

1.1.1 Principal Component Analysis

Principal Component Analysis, commonly known as PCA, is a mathematical technique that is used to explore data, specially high-dimensional data, to extract the most important trends in the data.

When thinking of gene expression data, high dimensionality comes from the large number of dimensions of the data. This is, the result of each experiment can be thought as a kind of space, where each each feature is a coordinate in the space. There are typically thousands of genes (dimensions) and the structure of pattern in the data extends to all the dimensions.

How PCA works

Table 1: Example of the output provided by cufflinks for the quantification of gene expression from RNA sequencing data.

tracking_id	sample1	sample2	sample3	sample4	sample5	sample6	sample7	sample8
XLOC_000001	35.1077	50.9662	78.7724	35.4736	69.6067	63.9241	57.7967	61.4227
XLOC_000002	49.7359	64.6178	46.8884	74.617	66.0371	42.9654	645.65	64.8351
XLOC_000003	0	0	0.937767	0	0	0	0	0
XLOC_000004	89.7196	85.5504	185.678	74.617	142.783	168.718	172.63	167.206
XLOC_000005	12.6778	39.1347	158.483	22.0181	28.5566	45.0613	15.9701	50.0481
XLOC_000006	10.7273	9.1011	10.3154	13.4555	7.13915	6.28762	7.60483	12.512
XLOC_000007	0	0	0.937767	0	0	0	0	0
XLOC_000008	55.5871	37.3145	86.2746	66.0544	66.9295	53.4448	54.7548	75.0722
XLOC_000009	37.0581	16.382	24.3819	24.4646	38.3729	15.7191	24.3355	50.0481
XLOC_000010	812.352	483.269	696.761	748.616	1094.97	521.873	675.309	741.622
XLOC_000011	0	0	0	0	0	1.04657	0.760483	1.13746

The mean represents the average of the values in the data:

$$\bar{\mathbf{X}} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{1}$$

The variance provides the the spread of the data:

$$\text{Var}(\mathbf{X}) = \sigma^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{\mathbf{X}})^2 \tag{2}$$

For example, figure 2 shows two distributions with the same mean but different variance. This means that the data points are at the same location but with a different strength. Thus, the third statistic we'll need is the covariance.

The covariance represents the degree of co-dependence of two variables, i.e., it measures the co-dependency of two variables, given by:

$$\text{Cov}(\mathbf{X}, \mathbf{Y}) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{\mathbf{X}})(y_i - \bar{\mathbf{Y}}) \tag{3}$$

Increases with increasing co-dependency and variance. Just as the variance measures the degree to which a set of data varies, the co-variance is a measure of the way two sets of data vary together.

$$\text{Cov}(\mathbf{X}, \mathbf{X}) = \text{Var}(\mathbf{X}) \tag{4}$$

The covariance also increases in magnitude as the variance of each of the two datasets increases. Correlation values can be negative or positive, indicating whether the values of two variables increase or decrease together. Figure 3 shows some examples of correlation.
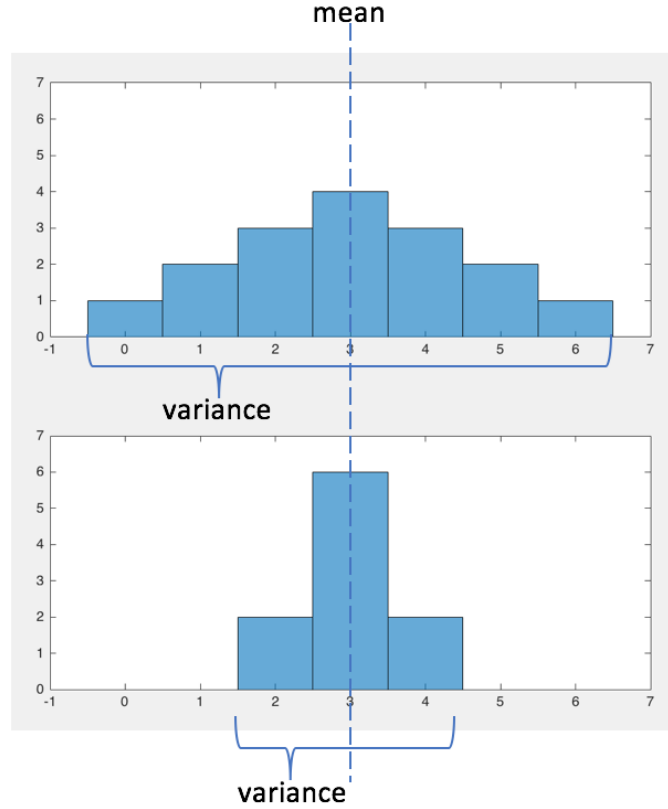
Figure 2: Two distributions with the same mean but different variance.

Coordinate transformations

In a two dimensional space described by coordinates, a point in space is described by X and Y such that $\mathbf{v} = [x_1, y_1]$. For example, the vector $v_1 = [1\ 2]^T$ represents a point in the 2 dimensional space as shown in Figure 4 (a). An alternative coordinate system described by the coordinates X' and Y', has a different column vector describing the same point $\mathbf{v}' = [x_1', y_1']$, shown in Figure 4 (b).

The two coordinate systems are $T\mathbf{v} = \mathbf{v}'$, related to the orthogonal transform matrix T. An orthogonal matrix is the kind of matrix which performs rotated-axis coordinate transforms. Thus, we can make a new coordinate system by using a transformation matrix T, which relates the two coordinates vectors by matrix multiplication. There are many types of transformations but we are particularly interested in transformations which rotate the coordinate axis. These are performed by matrices which have the property called orthogonality.

Eigenvalues and Eigenvectors
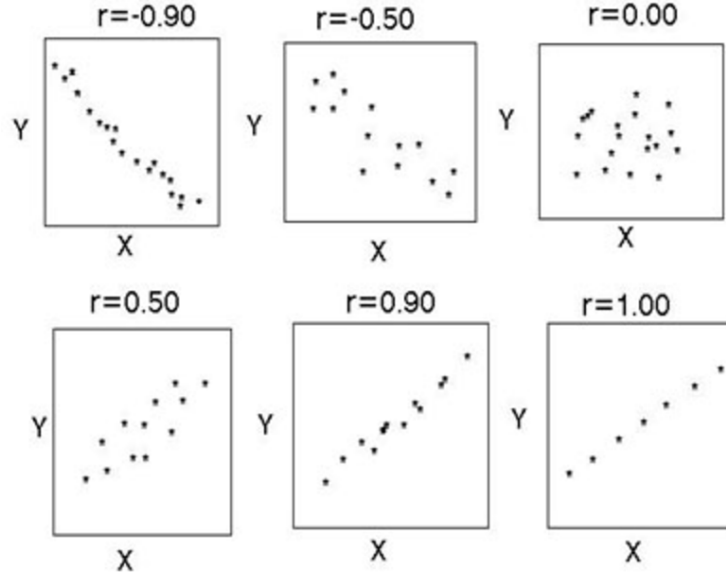
Figure 3: Examples of correlation

a)

b)
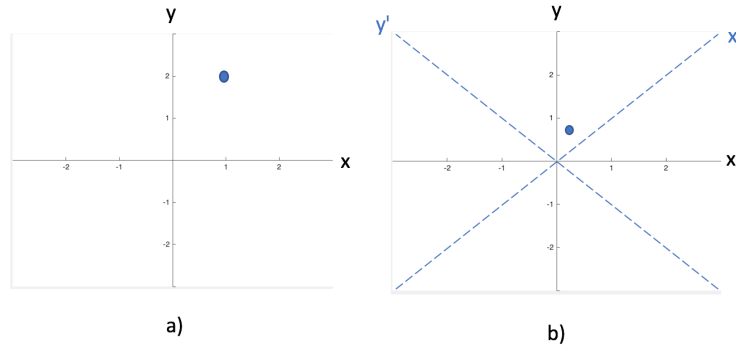
Figure 4: Example of a coordinate transform

When a transformation matrix maps a vector to a multiple of itself, then the vector is called an Eigenvector. The amount by which the vector is multiplied (stretched) is the associated Eigenvalue:

$$Tx = \lambda x \tag{5}$$

where λ are the Eigenvalues and x are the Eigenvectors.

In general terms, PCA uses covariance to encode the structure in the data and then eigenvectors to devise a new set of coordinates that best reveals the structure by finding the appropriate set of directions. One result from linear algebra is that if the eigenvectors are placed next to each other to construct an orthogonal matrix that performs a

coordinate transformation. It is important to mention that Eigenvectors are placed in descending order of their corresponding eigenvalue to ensure that the first components encode most of the variance in the data. The transpose of this matrix of Eigenvectos is an orthogonal matrix which performs a rotated-axis coordinate transformation. We can transform our data matrix, D, to the new coordinates, D_{PCA}:

$$D_{PCA} = W^T D$$

For example:

The matrix: $\begin{pmatrix} 1 & 3 \\ 2 & 2 \end{pmatrix}$ has eigenvalues 4 and -1 and the eigenvectors $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} 3 \\ -2 \end{pmatrix}$

such that

$$\begin{pmatrix} 1 & 3 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 4 \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 3 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 3 \\ -2 \end{pmatrix} = -1 \begin{pmatrix} 3 \\ -2 \end{pmatrix}$$

The orthogonal matrix using these Eigenvector is:

$$W^T = \begin{bmatrix} 1 & 1 \\ 3 & -2 \end{bmatrix}$$

1.1.2 Exercise: PCA for the expression of two genes

We will use R and the package stats to perform PCA. We will use an example data which represents several measurements of the expression of two genes, x and y, with the following values:

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2.0	1.6
1.0	1.1
1.5	1.6
1.1	0.9

We start by create a matrix of points in 2-d space (gene expression data) by using the following syntax:

```
#n number of samples
name_gene_1 <- c(values_in_sample1, ..., _value_in_sampleN)
#m number of genes
name_gene_2 <- c(gene_1, gene_2, ..., gene_m)
samples <- c(sample1, ..., sampleN)
# expression matrix
Exp <- data.frame(gene1 = name_gene1, ..., geneM = name_geneM)
```

Then, we plot these two genes (see figure 5a) using the command:

```
plot(x, y)
```

Trends are already apparent because data is simple but this is not usually the case. We then perform an statistical analysis using Principal Component Analysis. Before starting with PCA, it is best to first have centered the data with mean zero. This is, calculate the mean of each of the two variables and substracted to obtain centered data (shown in figure 5b).

Now, let us calculate the covariance matrix. Covariance matrix for two variables:

$$\left[\begin{array}{cc} \text{Cov(x,x)} & \text{Cov(x,y)} \\ \text{Cov(y,x)} & \text{Cov(y,y)} \end{array} \right]$$

and Covariance matrix for our data:

$$\left[\begin{array}{cc} 0.016 & 0.615 \\ 0.615 & 0.716 \end{array} \right]$$

The Eigenvalues of this matrix are: 1.284 and 0.0490. Eigenvalues gives the relative variance of the data in the direction defined by the Eigenvectors. From the values we can inferred that most variation is in one direction. To calculate the Eigenvalues in R use type:

```
eigen(covariance_matrix)
```

The corresponding eigenvector are then placed in a matrix in descending order of eigenvalue:

(a)

(b)

(c)
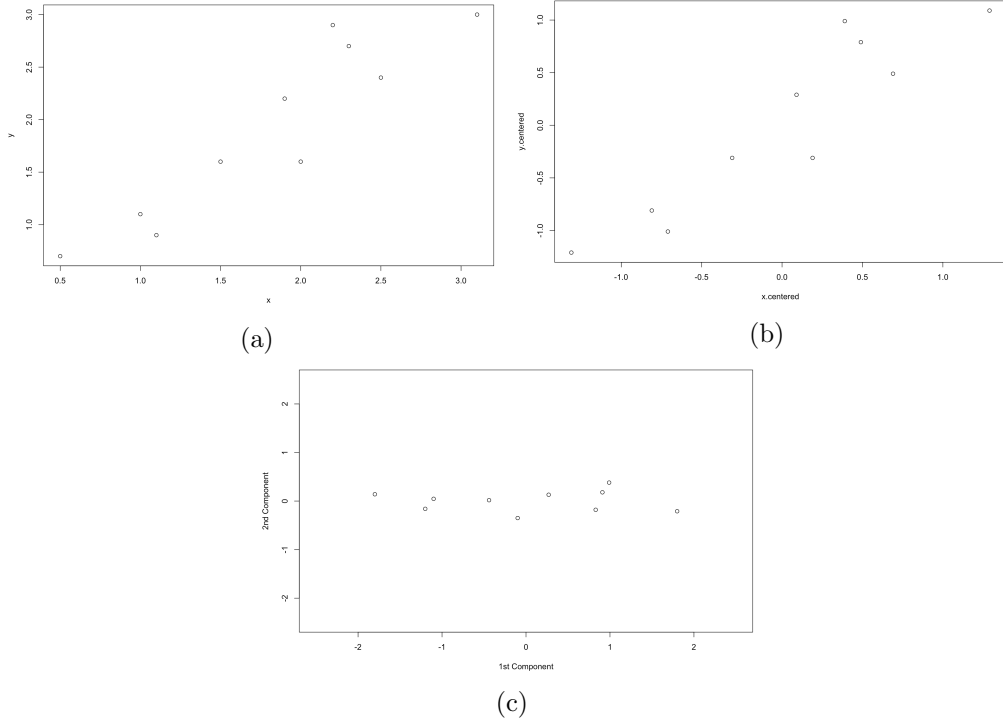
Figure 5: Plot: in (a) shows the expression of two genes, (b) the expression of the same two genes after centering the data (expression has zero mean), (c) gene expression is plot in the new coordiantes (PCA)

$$\begin{bmatrix} 0.6778734 & -0.7351787 \\ 0.7351787 & 0.6778734 \end{bmatrix}$$

The transpose of this Eigenvectos will perform the coordinate transformation:

$$W^T = \begin{bmatrix} 0.6778734 & 0.7351787 \\ -0.7351787 & 0.6778734 \end{bmatrix}$$

This is an orthogonal matrix which performs a rotated-axis coordinate transformation. We can transform our data matrix so that the data is represented in the new coordinates:

$$D_{PCA} = W^T D$$

which in our example is:

$$D_{PCA} = \begin{bmatrix} 0.6778734 & 0.7351787 \\ -0.7351787 & 0.6778734 \end{bmatrix} \begin{bmatrix} 0.69 & -1.31 & 0.39 & 0.09 & 1.29 & 0.49 & 0.19 & -0.81 & -0.31 & -0.71 \\ 0.49 & -1.21 & 0.99 & 0.29 & 1.09 & 0.79 & -0.31 & -0.81 & -0.31 & -1.01 \end{bmatrix}$$

9

$$= \begin{bmatrix} 0.83 & -1.8 & 0.99 & 0.27 & 1.8 & 0.91 & -0.099 & -1.1 & -0.44 & -1.2 \\ -0.18 & 0.14 & 0.38 & 0.13 & -0.21 & 0.18 & -0.35 & 0.046 & 0.018 & -0.16 \end{bmatrix}$$

The we can plot our data in the new coordinates, as shown in figure 6, where each coordinate is called principal component. The first coordinate aligns with the direction in the expression space where has the most variation. Subsequent coordinates would align with directions with descending degrees of variation. This is why we are careful to order according to the size of the eigenvalues. Thus, PCA is capturing as much variation in the first component as possible, then the same for the second coordinate, and so on. In the case of our data, all the meaningful variation sees to have been captured with the first coordinate, or the first principal component. Specially compared to the second component which would seem to be random scatter. So we have reduced the dimensionality of our data from two to one. In cases when dealing with thousands of genes, PCA might be able to capture most of the variation of the data in with only two or three principal components. Thus making it easier to visualise it, which is one of the main motivations of performing PCA.
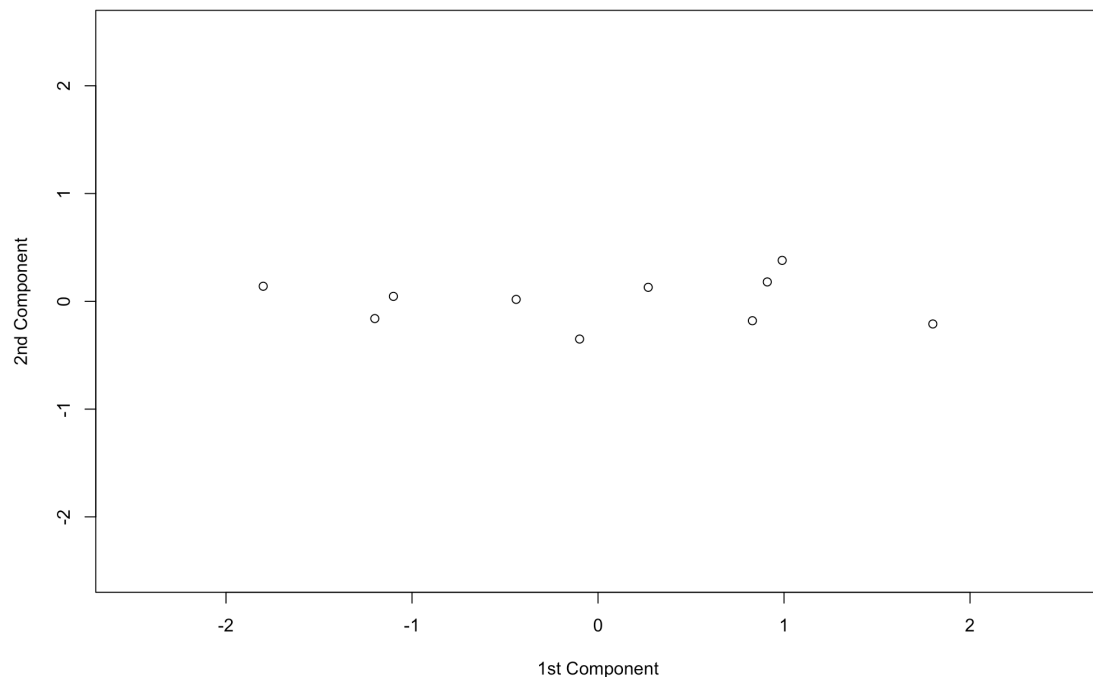

Figure 6: Plot of two genes in the new coordinates

1.1.3 Exercise: PCA using example data

In this example we will use a publicly available dataset to explore expression data. We will use the stats package in R for computing PCA and will show how to visualise it

using the function ggbiplot, which was implemented by Vince Q Vu [5].

Gene expression data is usually stored in a tab delimited text file. The extension of such files could be .csv, .soft, .xls(x), etc. Use Excel, R or MATLAB to open and preview the file. It is important to mention that gene expression values must be normalised before PCA plotting.

The dataset used in this example is from a study on non-alcoholic fatty liver disease, published in 2015 by Wruck et al. [6]. The transcriptomics data was extracted from nine from patients that were recruited in the Multidisciplinary Obesity Research project at the Medical University of Graz, Austria, or at the Interdisciplinary Adipositas Center at the Kantonsspital St Gallen, Switzerland. Each sample is described in table 2) in terms of gender, age, BMI, percent of steatosis and steatosis grouping.

Table 2: Details of the samples for microarray data for a study in fatty liver [6]

ID	gender	age	BMI	% steatosis	steatosis grouping
H0004	f	54	47	10	obese, low steatosis
H0007	f	33	51	40	obese, high steatosis
H0008	m	61	46	40	obese, high steatosis
H0009	f	48	49	5 - 10	obese, low steatosis
H0011	f	58	45	70	obese, high steatosis
H0012	f	50	35	0	obese, low steatosis
H0018	f	35	41	30 - 40	obese, high steatosis
H0021	m	49	41	0	no steatosis
H0022	m	45	49	40	obese, high steatosis

The gene expression matrix, gene annotation and sample annotation can be found in the following files:

- **gene-expression-table.txt**: gene expression table. This table is available as reference but for the simplicity we will use the following files which contain the expression data separately from the gene annotation, the names of samples, and the steatosis groups per sample.

- **gene-expression.txt**: numerical matrix for the gene expression values, where the columns represent genes and the rows represent the samples. This is the transpose of the expression matrix because the function requires the rows of the input matrix to be observations and the columns features, which means rows to be the gene expression profiles (samples) and columns to be the genes.

- **gene-annotation.txt**: string vector containing the gene names for the expression data

- **samples.txt**: name of each sample

- **groups.txt**: name of the steatosis group for each sample

11

Now, we can load the data into R to begin the analysis. We can either load the gene expression table by typing:

```
#Expression data is saved in a tabular txt file
#The data is in a numerical matrix with no headers
ExpData <- read.delim(FileName, header = FALSE, sep = '\t',
stringsAsFactors = FALSE)
Annotation <- read.delim(FileName, header = TRUE, sep = '\t',
stringsAsFactors = FALSE)
samples <- read.delim(FileName, header = TRUE, sep = '\t',
stringsAsFactors = FALSE)
genes.class <- read.delim(FileName, header = TRUE, sep = '\t',
stringsAsFactors = FALSE)
```

Next, calculate the principal components using *prcomp* and plot the first two components. Then plot the first two components using *ggbiplot* [5]:

```
## Computing the principal components using prcomp
        genes.pca <- prcomp(gene_expression_data)

# Plot the components using ggbiplot
        library(ggbiplot)

        g <- ggbiplot(genes.pca, obs.scale = 1, var.scale = 1,
                    groups = genes.class$groups, ellipse = TRUE,
                    circle = FALSE, var.axes = FALSE) +
          scale_color_discrete(name = '') +
          theme(legend.direction = 'horizontal', legend.position = 'top')

        print(g)
```

ggbiplot produces a plot which is shown in figure 7. Each dot is a gene expression from a sample in each category (group) from a patient, and is coloured by its type. The two axis are the first two principal components and the numbers represent the percentage of variance that is captured by each component. Typically, the first three component captures the most variance, whereas following components capture only a small percentage of variance. Dots of the same type tend to cluster together which means that samples of the same type have similar expression profiles. The distance

on the dots on each axis should not be treated equally (as each component captures a different percentage of variance). Thus, the difference on the first component should be taken into more consideration. Furthermore, the ellipse in the figure represents the normal data ellipse for each group for the details of 68%.

Figure 7: The first two principal components for gene expression data in a study on Fatty Liver. The plot was generated using the R package ggbiplot [5]

Exercise 3

Plot the three principal components for the expression data. Figure 8 shows an example of such plot.

Summary

In summary, PCA is a method of revealing underling trends in large amounts of data. PCA reduces high dimensional data to just a few principal components which hopefully capture most of the variation of the data and allows inferring meaningful structure.
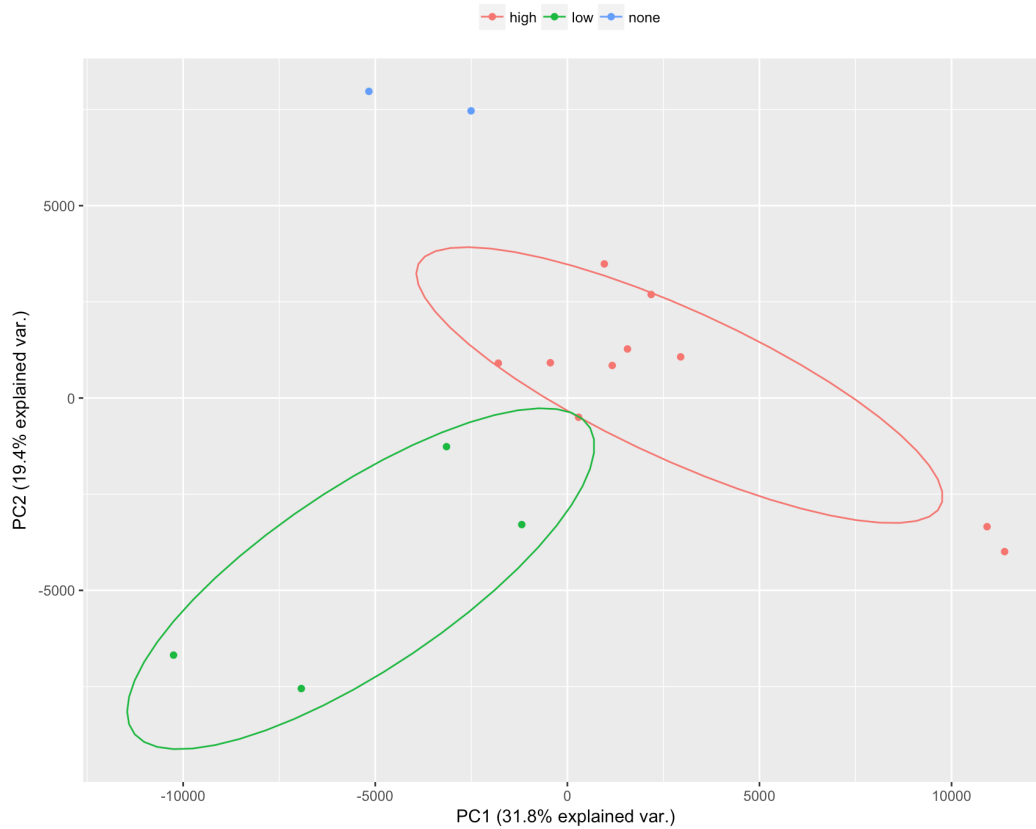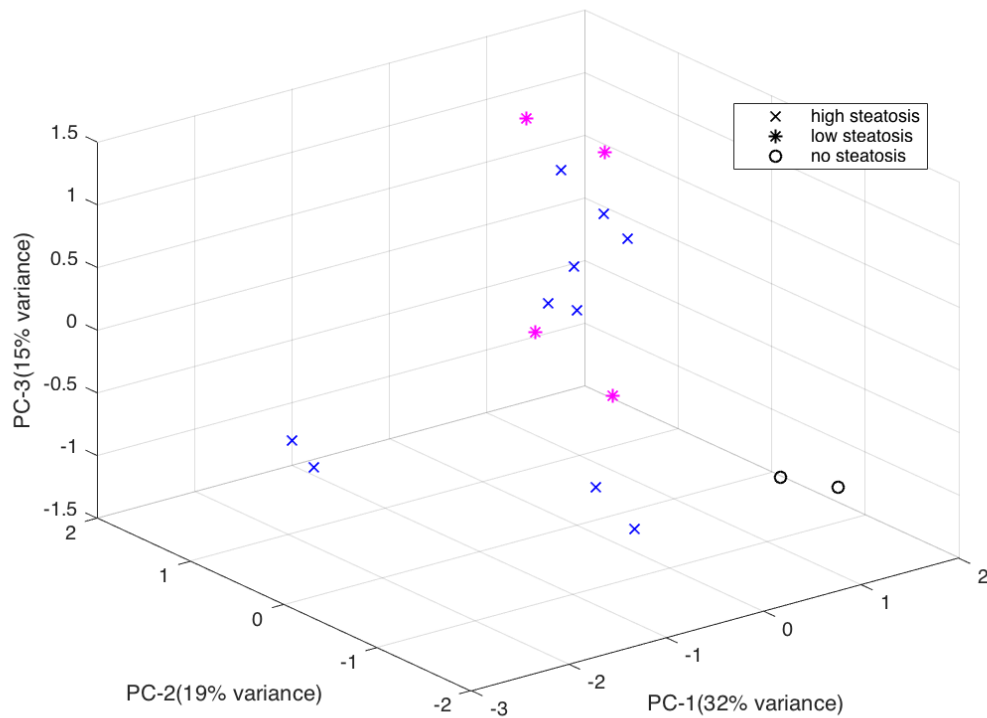
Figure 8: The first three principal components for gene expression data in a study on Fatty Liver [6]. Note: the plot was generated using MATLAB, can you plot the three components in R

A new coordinate system is constructed by rotating the axes (each representing a gene). The first new coordinate, or first principal component, is the direction in which the data varies most, then the second component, and so on. PCA allows to select a few new variables which contain most of the variation of the data which can also be visualised.

Some of the benefits of using PCA benefits are:

- A powerful tool to visualise high dimensional data

- Shows quantified difference among observations

- Used to assess data quality and discover relationships between data points

- Some software to compute PCA is available in MATLAB and R (using package stats)

14

1.1.4 Hierarchical Clustering

Hierarchical Clustering is another way to visualise high dimensional data. It clusters observations by distance and builds a hierarchical structure. It gives more detailed information of the differences among clusters, for example, what genes contributes the most to the differences between two clusters.

Hierarchical clustering uses a distance metric (typically Euclidean but could be correlation, Hamming distance, etc.) between each pair of genes to create a hierarchical tree-like structure of the data. Then it uses a linkage function to calculate the distance between clusters. For more details please see [1].

Figure 9 shows an example of clustergram from gene expression data. The clustergram is made of a heat map in the middle and dendograms in the left and top, with row and column labels on the right and bottom (depending on the number of genes and samples) and a scale bar. Each column is a sample expression profile, and each row represents a gene. The colours suggests relative expression values, where red indicates high expression values and blue indicates low expression values. Ideally, samples of the same type will cluster together, e.g., all control samples will cluster together and all cases as well.

1.1.5 Exercise: Hierarchical clustering for expression data

Compute hierarchical clustering on the Fatty Liver [6]. Since MATLAB is a commercial software, we will use a free to use wrapper for MATLAB's clustergram function. This wrapper is available from MultiPEN (`https://github.com/TGAC/MultiPEN/`, [4]). The wrapper provided in MultiPEN reads the expression data provided as a tabular file and plots the hierarchical clustering image on screen, which is also saved as a png image. To use the wrapper just use MultiPEN in a terminal using the following syntax:

Syntax

```
MultiPEN HierarchicalClustering Output Expression Threshold Title
```

Description

   MultiPEN: This is the path to the binary executable of MultiPEN
   Output: Specify directory to save the output image, default is: *output/MultiPEN/stats/*
   Expression: The expression data is in tabular format where the rows represent the features (e.g. genes) and the colums are the samples.
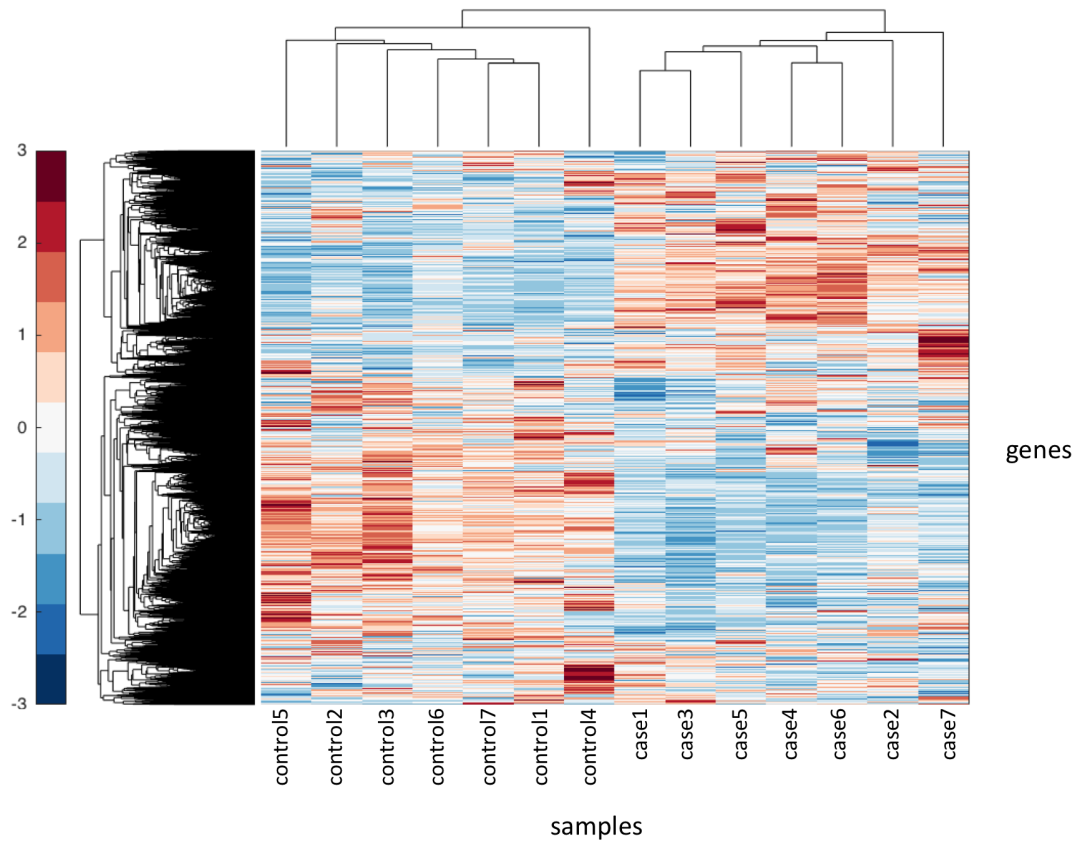
Figure 9: Example of hierarchical clustering using the MATLAB function clustergram

Threshold: To filter expression values. For example, for gene expression, it is common practice to discard genes with counts smaller than 100. This is an optional input argument.

Title: Specify the title to be displayed in the plot. This is an optional input argument.

Exercise

Run the the wrapper in MultiPEN to perform hierarchical clustering in the expression data for the Fatty Liver data [6] used in previous exercise. For more information on running MultiPEN visit `https://github.com/TGAC/MultiPEN/` [4].

1.2   Interaction Network

1.2.1   Compile the network with Cytoscape

Part II
Part II

2   Statistical Approaches with Sparsity

2.1   Exercise: Features selection from expression data using MultiPEN

Perform feature selection for the gene expression data on the Fatty Liver study [6]. Using the feature selection function provided by MultiPEN (`https://github.com/TGAC/MultiPEN/`, [4]) which uses a penalised logistic regression approach.

Syntax

```
MultiPEN FeatureSelection Output Expression Interactions Class lambda
DecisionThreshold NumIterations
```

Description

   MultiPEN: This is the path to the binary executable of MultiPEN
   Output: Specify directory to save the output image, default is: *output/MultiPEN/stats/*
   Expression: The expression data is in tabular format where the rows represent the features (e.g. genes) and the colums are the samples.
   Interactions: The interaction matrix where the ith interaction (row) is represented as: [source target score] where *source* and *target* are names (symbolID for genes and CHEBI IDs for metabolites) of the connected nodes and *score* is a number in the range [0,1] representing the interaction confidence (where 1 corresponds to the maximum level of confidence).
   Class: For each sample (one sample per row) specify if control (0) or case (1).
   lambda: This is the lambda parameter that optimises the logistic regression problem for your specific data. Different lambdas can be tested using cross validation, then selecting the value that provides better results (in terms of the size of the largest connected component, accuracy or area under the curve).

DecisionThreshold: The decision threshold is set to 0.5 by default. However, if want to test another value specify it here.

NumIterations: Maximum number of iterations for the optimisation solver. Default value is 100.

Feature Selection Output Files

Feature selection produces seven output files:

MultiPEN-Rankings_lambdaX.txt: Ranking of features for the corresponding lambda X.

MultiPEN-Rankings_lambdaX_genes-higher-in-cases.txt: Ranking of features which includes only features with higher expression in cases samples.

MultiPEN-Rankings_lambdaX_genes-higher-in-control.txt: Ranking of features which includes only features with higher expression in control samples.

MultiPEN-vts_lambdaX.txt: Intercept term (logistic regression model).

MultiPEN-performance_feature-selection_lambdaX.txt: file that contains statistics on the performance of feature selection: largest connected component (LCC), area under the curve (auc), accuracy, true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN).

MultiPEN-feature-selection_config.txt: Contains the information of the parameters used: lambda, number of iterations and decision threshold.

3 Pathway Analysis

Objective

Steps

Input data A list of ranked genes. For this session we will use the rankings from feature selection in ExampleOutputsMultiPENRankings_lambda0.0001.txt.

To run the R script type in the terminal:

Rscript enrichmentGO.R ../ExampleOutputs/MultiPENRankings_lambda0.0001.txt output/

References

[1] MATLAB and MathWorks. Compute hierarchical clustering.

[2] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S. Baliga, Jonathan T. Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11):2498–2504, 2003.

[3] Michael E Smoot, Keiichiro Ono, Johannes Ruscheinski, Peng-Liang Wang, and Trey Ideker. Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, 27(3):431–432, Feb 2011.

[4] Perla Troncoso-Rey and Wiktor Jurkowski. Multipen, 2017.

[5] Vinve Q. Vu. Ggbiplot.

[6] Wasco Wruck, Karl Kashofer, Samrina Rehman, Andriani Daskalaki, Daniela Berg, Ewa Gralka, Justyna Jozefczuk, Katharina Drews, Vikash Pandey, Christian Regenbrecht, Christoph Wierling, Paola Turano, Ulrike Korf, Kurt Zatloukal, Hans Lehrach, Hans V. Westerhoff, and James Adjaye. Multi-omic profiles of human non-alcoholic fatty liver disease tissue highlight heterogenic phenotypes. *Scientific Data*, 2:150068 EP –, 12 2015.