SCIE2100 Practical 6, Week 9

Aim

In this practical you will develop methods for analysing gene expression data. Specifically, you will prepare standard microarray data (as extracted from raw image data), identify genes that (a) appear significantly up- or down-regulated in human disease tissue (breast cancer), and (b) find genes that are co-expressed during the cell cycle in yeast.

Files

For this week's practical you will need this guide, *genome.py*, *stats.py*, *webservice.py* and two microarray data files *GDS38.soft* and *GDS3716.soft*; these data files were obtained from the Gene Expression Omnibus (GEO; http://www.ncbi.nlm.nih.gov/geo/).

Gene expression profiling using microarrays

Microarrays are often used to answer specific hypotheses about the patterns of expression of the complete set of genes of an organism—usually by comparing the relative levels of expression of individual genes. A single gene expression microarray provides expression levels of many genes in a given sample; what we want to do is compare across different samples. Before expression levels can be compared, the data needs to be transformed to improve the accuracy of comparisons.

In our microarray analysis, we'll assume that the value for each arrayed gene represents its relative expression level (relative to all other genes on the same chip and the set of samples collected). We will use GEO "data set" files, which are collections of expression samples.

Here, we'll be looking at a published gene expression dataset that allows us to compare expression for diseased vs. normal tissue in breast cancer (Graham et al., 2010). This dataset is available from the GEO, with accession GDS3716 (http://www.ncbi.nlm.nih.gov/sites/GDSbrowser?acc=GDS3716), though you should use the file provided on Blackboard.

A microarray dataset is essentially one big matrix: Usually, and in all our examples, each *row* corresponds to a gene (actually a probe with a specific DNA fingerprint). Each *column* corresponds to a sample (the biological material experimented with).

Preparing microarray data

Before doing any analysis, we need to transform our microarray expression level data (measured by the intensity of the image associated with a probe) to a value that allows statistically robust comparisons. For a broader tutorial on the subject, see Quackenbush (2002).

Open up the *GDS3716.soft* file in TextWrangler and inspect the data. Graham and colleagues collected microarray data for 42 patients: 18 suffering from breast cancer (9 ER- and 9 ER+) and 24 healthy controls (18 subject to reduction mammoplasty and

6 subject to prophylactic mastectomy). The 18 breast cancer samples and the 18 mammoplasty samples were age-matched (as ordered in file).

The first two columns are identifiers: the probe identifier and the corresponding gene name. The remaining columns correspond to the samples (i.e., the expression 'counts') from each of the 42 patients. Every row is a single probe; there may be multiple probes for a single gene, so there isn't exactly one row per gene, but each row should uniquely map to a single gene. In addition, the beginning of *GDS3716.soft* file holds useful information about the data. For instance,

```
!dataset_channel_count = 1
```

tells you that the microarray uses a single channel (not two-channel), and

```
!dataset_value_type = count
```

tells you that the data is the raw intensity data from the experiment; it hasn't already been processed, which is why we now need to transform it.

1. In order to be meaningful, the level of expression in a specific state should be compared to the level in a reference state. When we have two conditions (often referred to as *R* and *G* for Red and Green) we can assess the *change* between conditions. We often use the ratio between our reference state and state of interest (i.e., $T = R / G$).

   Firstly import *genome.py*

   ```
   >>> from genome import*
   ```

   Load *GDS3716.soft* using `readGEOFile` in *genome.py*,

   ```
   >>> ge3716 = readGEOFile('GDS3716.soft')
   ```

   This should create an instance of the `class` `GeneExpression`, which is defined in the same module.

   How many genes are profiled in the data file?
   You can find what 'samples' are in your set by

   ```
   >>> print ge3716.getHeaders()
   ['GSM512539', 'GSM512540', 'GSM512541',
   ```

   `GeneExpression` records the identifiers from the first two columns separately, so the first sample appears at index zero. You can access columns (one for each sample) by number,

   ```
   >>> print ge3716[0]
   [1662.2 292.2 37.6 ..., 81.2 8.5 564.4]
   ```

   What are these numbers?

   These numbers are the expression levels of all the genes in the given sample.

   We can also find out which sample is at column zero with the following.

   ```
   >>> print ge3716.getHeaderIndex('GSM512539')
   0
   ```

   You can get a dictionary (with probe/gene name as key) too. Here's an example that extracts samples with index 33 and 0:

```
>>> print ge3716.getSamples([33,0])
{'211034_s_at': array([ 1858.5,  1662.2]), '212034_s_at': ...
```

211034_s_at is a probe on the microarray chip.

Now, let’s get to work. Create a new data set containing the expression ratios for disease vs. healthy samples, using the reduction mammoplasty samples as healthy.

```
>>> ratio = GeneExpression('GDS3716_ratio')
>>> ratio.addSamples('S1_ER+/Healthy', ge3716.getRatio(33,0))
```

Note that you need to work out which columns that correspond to the correct states (e.g. that sample index 33 is ER positive, and that index 0 is the age-matched healthy tissue). You will have to do this for all the other column pairs too. In the process you are creating a new `GeneExpression` data set where each sample is a ratio between two in the original data set.

Hint: The `getHeaderIndex` method will be useful to find the indexes of the breast cancer and normal samples.

In order to get useful expression ratios, it is typical to take the $\log_2$ transform of the ratio so that up- and down-regulation are easily comparable, i.e., $\log_2(R / G) = -\log_2(G / R)$. Pair each breast cancer sample (ER+1-9 and ER-1-9) with a healthy sample (RM1-18) and determine the difference in expression.

Submit (Q1-a): Explain why it is “useful” with log-transformed log ratios. In your response, pick three probes, and show the (i) the raw values, (ii) the ratio, and (iii) the log-ratio for the first ER positive patient sample and its age-matched healthy sample. Include in you response the three probes that you picked.

Submit (Q1-b): Print and plot a histogram of the all the resulting expression ratios. You should use `matplotlib` for plotting. The easiest way to generate a histogram of a list of numbers (called `values`) is:

```
>>> import matplotlib.pyplot as plt
>>> values = ratio.matrix[:] # this will give all values
>>> plt.hist(values.ravel(), bins=50)
>>> plt.show()
```

Submit (Q1-c): Determine for each gene, the log-ratio between all sample pairs (18 in total). Plot the log-transformed ratios and compare to the ratios. The ratios and the log transformed follow quite different distributions. The log-transformed values should be close to normal—comment on whether you see this in the plot.

Identify significantly up- or down-regulated genes

Now that we have a transformed set of expression data giving us ratios, we can analyse the data to identify genes that are differentially expressed in healthy and diseased states. Specifically, we want to understand which genes have different expression levels in healthy and breast cancer samples; this information would ultimately help to find what genes might be causing disease, whether by increased or by decreased expression.

2. In order to obtain genes with consistently large positive or negative changes, you need to first establish the mean log-transformed ratio and the standard deviation of each pair. If the values follow a normal distribution, the Z-score is suited to identify changes for each pair.

   The `GeneExpression` class in this week's *genome.py* file has an implementation of Z-scoring: `getZScore`. This method takes a column identifier and returns a dictionary mapping from probe identifiers to their Z-score.

   Submit (Q2): List all genes that are significantly up- or down-regulated in breast cancer compared to the healthy samples. Specifically, identify and list all genes that are two or more standard deviations away (up or down) from their respective means (i.e., have Z-scores of greater than 2 or less than -2) for at least 11 sample pairs. Pick one example of an up- or down-regulated gene (specify probe identifier) and explain how its Z-score was determined (including the necessary transformations).
   Include in your submission the code you used to obtain the Z-score.

Find co-expressed genes during the cell cycle

In the previous section we looked at expression differences across conditions. Another way of looking at gene expression data is as correlation between genes. For example, we can ask what other genes are commonly co-expressed with a given gene.

The cell cycle is an interesting system for our purpose, as it necessitates careful regulation and coordination of multiple genes at different times in the cycle. Many genes have specific roles to play during the cell cycle. Their level of expression may thus be periodic and cyclic. In particular many kinases (enzymes that phosphorylate other proteins) tend to work in concert with the cell cycle (and may even control the progression of the cell cycle).

In order to investigate gene co-expression, we'll be looking at yeast cell cycle data (Spellman et al., 1998). This dataset is available from the GEO, with accession GDS38 (http://www.ncbi.nlm.nih.gov/sites/GDSbrowser?acc=GDS38), though you should use the file provided on Blackboard. Open the GDS38 file in TextWrangler.

This data set is different from the cancer data set above, in that

```
!dataset_channel_count = 2
!dataset_value_type = log ratio
```

which means that the data has already been prepared and that you can start analysing immediately. The data set represents the level of expression of all genes in yeast collected over the cell cycle at a number of time points (see Spellman et al., 1998). All cells that were being analysed were "synchronised" to make sure that time points were in agreement.

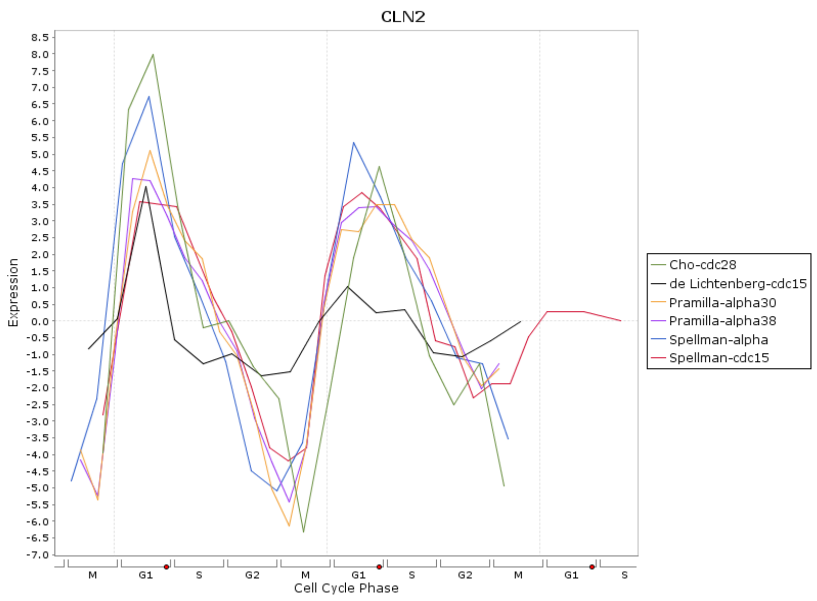

Figure 1: According to several microarray studies, the expression of the kinase CLN2 is highly periodic during the cell cycle (graph from cyclebase.org).

3. The kinase CLN2 has been experimentally shown to exhibit a highly periodic expression pattern (see Figure 1). Identify additional genes that follow the same pattern. Analyse the expression levels of CLN2 with all other genes using Pearson's correlation coefficient and list the five most correlated genes.

   You'll need to search through the *GDS38.soft* file to find the probe identifier corresponding to CLN2. There is an implementation of Pearson's correlation coefficient in the **`GeneExpression`** class in *genome.py*, called **`getPearson`**, which takes a probe identifier and returns a dictionary mapping probe identifiers to correlation coefficients.

   First load *GDS38.soft* using **`readGEOFile`** in *genome.py* with the alternative approach shown below.
   Note that by default each row is keyed by the probe identifier, but you can associate samples with the gene identifier instead:

```
>>> ge38 = readGEOFile('GDS38.soft', id_column=1).
```

   Submit (Q3-a): Sort your correlation coefficients, and list the top five genes whose expression patterns most strongly correlate with that of CLN2. Include the code you used to obtain the top 5 genes with comments.

   Submit (Q3-b): Now that we have the top 5 genes. Using the Gene Ontology (GO) to determine if these top 5 genes are known to be involved in cell cycle activity. The code below should get you started. (Also see practical 1, which first introduced GO.)

```
>>> # replace CLN2 with the top 5 genes
>>> rows=search('taxonomy:4932+AND+gene:CLN2','uniprot',format='list')
>>> terms=getGOTerms(rows[0]) # take the first search term
>>> print terms
set(['GO:0005737', 'GO:0000321', 'GO:0000079', 'GO:0016538',
'GO:0051301', 'GO:0000307', 'GO:0005634', 'GO:0007049'])
```

Assessment

Answer the questions/complete the tasks highlighted above with Submit in a **PDF or Word document** and submit it to Blackboard as an **attached file** to the assessment by the due date.

References

Quackenbush J. Microarray data normalization and transformation. *Nat Genet* 2002; 32(4) Suppl:496-501. doi:10.1038/ng1032. PMID: 12454644

Graham, K. et al. Gene expression in histologically normal epithelium from breast cancer patients and from cancer-free prophylactic mastectomy patients shares a similar profile. *Br J Cancer*. 2010 Apr 13;102(8):1284-93. PMID: 20197764

Spellman PT, Sherlock G, Zhang MQ, Iyer VR et al. Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell* 1998 Dec;9(12):3273-97. PMID: 9843569