compendiumdb: a database and R package for storing and analyzing expression data

Umesh K. Nandal and Perry D. Moerland

January 20, 2014

Contents

1 Introduction **1**

2 Installation of compendiumdb package **2**

3 Commonly used functions **4**
3.1 Connecting to and creating the database 4
3.2 Loading data into the compendium database 4
3.3 Creating ExpressionSets 6

4 Sample annotation **7**
4.1 Using *createESET* 7
4.2 Using the inSilicoDb package 8
4.3 Using GEO Datasets 9

5 Querying the compendium database **10**

6 Use case: building a breast cancer compendium **10**
6.1 Functional enrichment analysis 11

1 Introduction

Public repositories such as the Gene Expression Omnibus (GEO) (Barrett et al. 2013) and ArrayExpress (Rustici et al. 2013) provide a large amount of expression data from a wide range of studies performed in different organisms and on different (microarray) platforms. However, integrating datasets for a specific domain of study to extract meaningful biological information from these repositories is often challenging. Both the use of different platforms and the maintenance of a large number of flat files can hamper an integrative analysis of these datasets. Several programs and web-based resources have been developed (Bareke et al. 2010; Cheng et al. 2010; Kilpinen et al. 2008; Lacson et al. 2010; Liu et al. 2011; Taminau et al. 2011; Xia et al. 2009) to facilitate the aggregation of data from gene expression data repositories. However, a uniform, flexible, and portable framework for analysis and integration of these datasets is still lacking. We developed the R package compendiumdb that enables

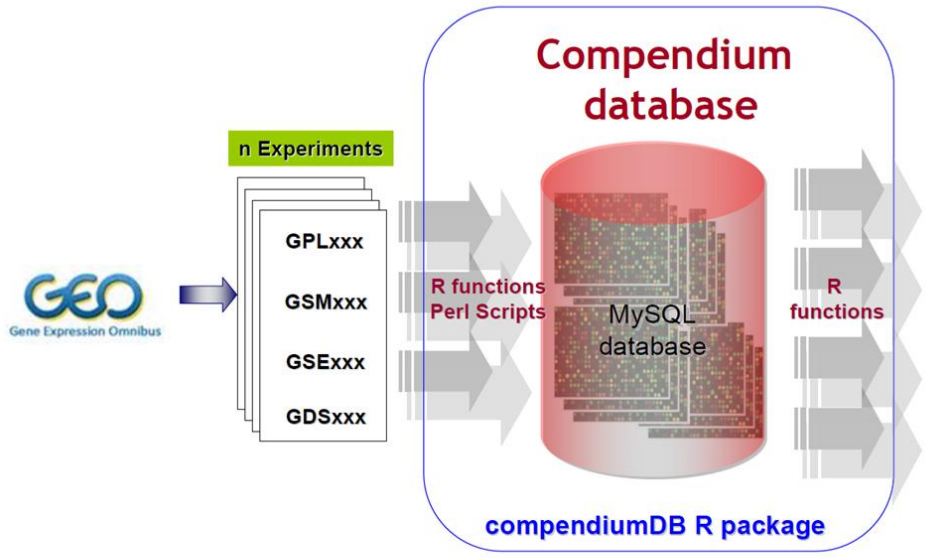

Figure 1: Typical workflow when loading data from GEO into the database using the compendiumdb package

building a domain-specific compendium of expression datasets via a flexible and homogeneous framework in the form of a MySQL database. The compendiumdb package consists of a number of R functions developed to access the database either locally or remotely. The database schema has been designed to be rich enough to store most of the information provided by MIAME-compliant expression databases such as GEO. Furthermore, an empty MySQL database in the form of a MySQL dump file is bundled with the package.

The objective of the compendiumdb package is to provide a homogeneous framework to store and analyze a large number of expression datasets from different studies and expression profiling platforms (Figure 1). The package provides R functions to (*i*) download data from GEO given the identifier of the experiment, (*ii*) load the expression data and probe annotation to a relational database, and (*iii*) save the expression data as an *ExpressionSet* (in binary file format). The resulting *ExpressionSet* and other data stored in the compendium database can then be queried using R functions.

2 Installation of compendiumdb package

Installing compendiumdb in a Linux or MacOSX environment is straightforward. It just requires recent distributions of MySQL and Perl to be present. The package can then be installed from source the default way (using *install.packages*).

Under a Windows operating system some more effort is required, since no Windows binary

has been made available for the package RMySQL that compendiumdb depends upon. The steps to take are:

1. Install the most recent version of MySQL:
   (a) Download the MySQL Installer from `http://dev.mysql.com/downloads/installer/`.
   (b) Open the MySQL Installer by clicking on the MSI (MicroSoft Installer) file you just downloaded.
   (c) Use the default settings; for a minimal installaton choose 'Server only' under Setup Type.
   (d) Create a root account by entering a password under Configuration.
2. Add the path name to your MySQL bin directory (e.g., `C:\Program Files\MySQL\MySQL Server 5.6\bin` to the PATH environment variable (see `http://dev.mysql.com/doc/mysql-windows-excerpt/5.6/en/mysql-installation-windows-path.html`).
3. Open a Command Prompt window and log in to your MySQL account by typing `mysql -u root -p` and entering your password for the root account.
4. On the mysql prompt create a database named `compendium` using `CREATE DATABASE compendium;`.
5. Install a recent version of Perl:
   (a) Go to ActiveState's ActivePerl home page `http://www.activestate.com/activeperl`.
   (b) Click on 'Download Now' and to download the installer for ActivePerl for Windows. There is no need to fill out any of the contact information on the next page in order to download ActivePerl.
   (c) Install ActivePerl by clicking on the MSI file you just downloaded and accepting the default options.
   (d) Check if the following Perl modules are already installed. If not go to the Command Prompt and type `ppm`. This will open the Perl Package Manager. Install the following modules: `DateTime-Format-DateManip, DBD-mysql`.
6. Install the latest version of Rtools by downloading the executable from `http://cran.r-project.org/bin/windows/Rtools/` and running it. In the setup check the option to edit the PATH environment variable.
7. Install the RMySQL package (`http://cran.r-project.org/web/packages/RMySQL/index.html`).
   (a) Create (if it does not exist yet) or edit the file `C:\Program Files\R\R-3.0.2\etc\Renviron.site` and add a line containing the path to your MySQL installation: `MYSQL_HOME="C:\Program Files\MySQL\MySQL Server 5.6"`. If necesary, change these to the settings appropriate for your computer.

(b) Copy `libmysql.lib` from `MYSQL_HOME\lib` to `MYSQL_HOME\lib\opt` to meet dependencies (create the directory `opt` if it does not exist yet).

(c) Start R and run *install.packages('RMySQL',type='source')*.

(d) Load RMySQL using *library(RMySQL)*.

8. Install compendiumdb:

```
> install.packages("path_to_tar_file/compendiumdb_0.1.0.tar.gz",
+     type = "source")
```

3 Commonly used functions

3.1 Connecting to and creating the database

We start by loading compendiumdb in the current R session:

```
> library(compendiumDB)
```

To create a compendium database one first has to connect to the MySQL database using the function *connectDatabase*. Before calling this function, a MySQL server should be running on the host machine and an (empty) database `compendium` has to be created on the MySQL server (see Section 2).

```
> conn = connectDatabase(user = "root", password = "root", host = "localhost",
+     dbname = "compendium")
```

Here we connected to a database running on a local machine, but the *host* argument can also be used to connect to a database on a remote server. Once the connection to the database has been established, load the database schema of the MySQL compendium database using the function *loadDatabaseSchema* (default value *updateSchema=FALSE*):

```
> loadDatabaseSchema(conn, updateSchema = TRUE)
```

Note that in general one should set *updateSchema=TRUE* only once, i.e., before filling the database with expression data, or if one explicitly wants to delete all the records of the database and reload the schema.

3.2 Loading data into the compendium database

First one should download the expression datasets of interest from GEO (`http://www.ncbi.nlm.nih.gov/geo/`). For this purpose, the package provides the function *downloadGEOdata*. GEO contains the following types of records (see also `http://www.ncbi.nlm.nih.gov/geo/info/overview.html`):

- Platform record (GPL): describes properties of the microarray, e.g., cDNA or oligonucleotide probesets. Each platform has a unique identifier (GPLxxx).

- Sample record (GSM): describes the conditions under which an individual Sample in the experiment was handled, the manipulations it underwent, and the abundance measurement of each element derived from it. It refers to only one sample and can be part of multiple series. Its unique identifier is GSMxxx.
- Series record (GSE): links a number of individual related samples together and provides a description of the whole study, the data obtained, analysis and conclusions. Its unique identifier is GSExxx.
- Dataset record (GDS):

The function *downloadGEOdata* downloads SOFT (Simple Omnibus Format in Text) files from GEO to the user's local machine for GSEs, GPLs, GSMs, and GDSs corresponding to the GSE identifiers provided by the user. For example, here we download the very first Series record and its associated GPL and GSMs from GEO:

```
> downloadGEOdata(GSEid = "GSE18290", destdir = getwd())
```

The function *downloadGEOdata* creates a data directory called `BigMac` (BIoinformatics Group MicroArray Compendium) in a directory *destdir* specified by the user. The `BigMac` directory contains several subdirectories: `annotation`, `COMPENDIUM`, `data` and `log`. The `data` directory contains further subdirectories to store the downloaded SOFT files corresponding to GSEs, GSMs, GPLs, and GDSs downloaded from GEO. More information about the structure of the `BigMac` directory can be found at `http://www.bioinformaticslaboratory.nl/twiki/bin/view/BioLab/compendiumdb`.

The data corresponding to GSE18290, for example, can be loaded to the compendium database using the function *loadDataToCompendium*:

```
> loadDataToCompendium(conn, "GSE18290")
```

The current contents of the compendium database can be inspected using the function *GSEinDB*:

```
> GSEinDB(conn)
```

```
   id_Compendium Experiment experimentDesign    Chip Samples           Tag
1              1   GSE18290               SC GPL2112      16          <NA>
2              1   GSE18290               SC  GPL339      18          <NA>
3              1   GSE18290               SC  GPL570      18          <NA>
4              2   GSE35547               SC GPL6885       8          <NA>
5              3   GSE18931               SC  GPL570       6          <NA>
6              4   GSE11121               SC   GPL96     200 breast cancer
7              5    GSE2990               SC   GPL96     189 breast cancer
8              6    GSE7390               SC   GPL96     198 breast cancer
9              8    GSE1456               SC   GPL96     159 breast cancer
10             8    GSE1456               SC   GPL97     159 breast cancer
   OrganismNCBIid OrganismName     GDS         date_loaded
1            9913   Bos taurus GDS3960 2014-01-20 10:22:08
2           10090 Mus musculus GDS3958 2014-01-20 10:22:08
```

```
3            9606 Homo sapiens GDS3959 2014-01-20 10:22:08
4           10090 Mus musculus    <NA> 2014-01-20 10:27:08
5            9606 Homo sapiens    <NA> 2014-01-20 10:28:08
6            9606 Homo sapiens    <NA> 2014-01-20 10:39:14
7            9606 Homo sapiens    <NA> 2014-01-20 10:46:30
8            9606 Homo sapiens    <NA> 2014-01-20 10:52:55
9            9606 Homo sapiens    <NA> 2014-01-20 11:29:06
10           9606 Homo sapiens    <NA> 2014-01-20 11:29:06
```

GSE18290 contains time course expression data from early bovine, human, and mouse embryos (Xie et al. 2010). Since a different platform was used for each species the table contains three entries, one for each species.

3.3 Creating ExpressionSets

Once a dataset has been loaded into the database, one would often like to further analyze the dataset using other packages provided in R/Bioconductor. For this purpose the package provides the function *createESET* that creates an *ExpressionSet* given a GSE identifier:

```
> createESET(conn, "GSE18290")

ExpressionSet esetGSE18290_GPL570 created
ExpressionSet esetGSE18290_GPL339 created
ExpressionSet esetGSE18290_GPL2112 created

> esetGSE18290_GPL2112

ExpressionSet (storageMode: lockedEnvironment)
assayData: 24128 features, 16 samples
  element names: exprs
protocolData: none
phenoData
  rowNames: GSM456627 GSM456628 ... GSM456642 (16 total)
  varLabels: development stage GPL
  varMetadata: labelDescription
featureData
  featureNames: AFFX-BioB-5_at AFFX-BioB-M_at ... Bt.19900.1.A1_at
    (24128 total)
  fvarLabels: ID Gene title ... GenBank Accession (10 total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation: GPL2112
```

Again, since a different platform was used for each species, three different *ExpressionSets* were created. The numerical data contained in the *assayData* slot is identical to the normalized expression data provided by GEO. The *featureData* slot is based upon the most recent (probe) annotation provided by GEO.

4 Sample annotation

It is a well-known problem that the annotation of individual samples in public expression data repositories is often inconsistent or even non-existent (Pitzer et al. 2009). The compendiumdb package offers various ways to obtain a better sample annotation.

4.1 Using *createESET*

As an example, consider GSE35547 containing gene expression data on the role of Notch in CD4+ T cell response (Helbig et al. 2012):

```
> downloadGEOdata("GSE35547")
> loadDataToCompendium(conn, "GSE35547")
```

The function *GSMdescriptions* provides a convenient overview of the sample title, sample characteristics, and sample source fields for each sample.

```
> head(GSMdescriptions(conn, "GSE35547"), n = 4)
```

```
          sampletitle
GSM870390 "C-Ig day1_mouse1"
GSM870391 "C-Ig_day3_mouse1"
GSM870392 "DLL4_day1_mouse1"
GSM870393 "C-Ig_day3_mouse1 (technical replicate)"
          samplesource
GSM870390 "naive CD4+ T cells, control, day 1"
GSM870391 "naive CD4+ T cells, control, day 3"
GSM870392 "naive CD4+ T cells, Delta4-Ig, day 1"
GSM870393 "naive CD4+ T cells, control, day 3"
          samplechar
GSM870390 "strain: C57BL6/NCrl;tissue: inguinal, axillary and brachial lymph nodes and spleen
GSM870391 "strain: C57BL6/NCrl;tissue: inguinal, axillary and brachial lymph nodes and spleen
GSM870392 "strain: C57BL6/NCrl;tissue: inguinal, axillary and brachial lymph nodes and spleen
GSM870393 "strain: C57BL6/NCrl;tissue: inguinal, axillary and brachial lymph nodes and spleen
          GPL
GSM870390 "GPL6885"
GSM870391 "GPL6885"
GSM870392 "GPL6885"
GSM870393 "GPL6885"
```

According to GEO guidelines (see `http://www.ncbi.nlm.nih.gov/geo/info/spreadsheet.html#samples_tab`) the sample characteristics field should contain detailed sample annotation. For GSE35547 this is indeed the case, and for each sample the variables strain, tissue, celltype, stimulus, timepoint, and mouse are defined. In GSMdescriptions these variables are separated by ';'. The function *createESET* with its argument *parsing=TRUE* enables splitting the sample characteristics into separate columns:

```
> createESET(conn, "GSE35547", parsing = TRUE)
```

```
Parsing phenoData.................Done.
ExpressionSet esetGSE35547_GPL6885 created
```

```
> head(pData(esetGSE35547_GPL6885), n = 4)
```

```
               strain                                                    tissue
GSM870390  C57BL6/NCrl  inguinal, axillary and brachial lymph nodes and spleen
GSM870391  C57BL6/NCrl  inguinal, axillary and brachial lymph nodes and spleen
GSM870392  C57BL6/NCrl  inguinal, axillary and brachial lymph nodes and spleen
GSM870393  C57BL6/NCrl  inguinal, axillary and brachial lymph nodes and spleen
                  cell_type    stimulus timepoint mouse
GSM870390  naive CD4+ T cells  control Ig     day 1     1
GSM870391  naive CD4+ T cells  control Ig     day 3     1
GSM870392  naive CD4+ T cells   Delta4-Ig     day 1     1
GSM870393  naive CD4+ T cells  control Ig     day 3     1
                                    sampletitle
GSM870390                         C-Ig day1_mouse1
GSM870391                         C-Ig_day3_mouse1
GSM870392                         DLL4_day1_mouse1
GSM870393 C-Ig_day3_mouse1 (technical replicate)
                                  samplesource     GPL
GSM870390   naive CD4+ T cells, control, day 1 GPL6885
GSM870391   naive CD4+ T cells, control, day 3 GPL6885
GSM870392 naive CD4+ T cells, Delta4-Ig, day 1 GPL6885
GSM870393   naive CD4+ T cells, control, day 3 GPL6885
```

With the resulting *ExpressionSet* it is straightforward to perform follow-up analyses, for example, testing for differential expression using limma.

4.2 Using the inSilicoDb package

Often data uploaded to GEO does not conform with the guidelines. As an example, consider GSE18931 containing gene expression data in human normal mammary stem cells (Pece et al. 2010):

```
> downloadGEOdata("GSE18931")
> loadDataToCompendium(conn, "GSE18931")
> GSMdescriptions(conn, "GSE18931")
```

Here the essential information of a sample being either PKH positive or PKH negative is not provided in the sample characteristics field but in the sample title field. In this case, one could use curated sample annotation accessible via the inSilicoDB (Taminau et al. 2011). This package is a command-line front-end to the InSilico DB (`http://insilico.ulb.ac.be`), a web-based database currently containing close to 160,000 expert-curated Affymetrix and Illumina expression profiles compiled from almost 3,700 GEO repository series in human, mouse, and rat (Coletta et al. 2012).

```
> require(inSilicoDb)
> sample.annot <- getAnnotations(gse = "GSE18931", gpl = "GPL570")
> pdata <- pData(sample.annot)
> pdata <- data.frame(rownames(pdata), pdata)
> head(pdata, n = 4)
```

```
          rownames.pdata.                Cell.Type  PKH26.Label
GSM468802       GSM468802 mammary epithelial cells PKH-negative
GSM468803       GSM468803 mammary epithelial cells PKH-positive
GSM468804       GSM468804 mammary epithelial cells PKH-negative
GSM468805       GSM468805 mammary epithelial cells PKH-positive
```

The current annotation of a sample can be easily updated using the function *updatePhenoData*:

```
> updatePhenoData(conn, pdata)
> head(GSMdescriptions(conn, "GSE18931"), n = 4)
```

```
          rownames.pdata. Cell.Type                   PKH26.Label    GPL
GSM468802 "GSM468802"     "mammary epithelial cells" "PKH-negative" "GPL570"
GSM468803 "GSM468803"     "mammary epithelial cells" "PKH-positive" "GPL570"
GSM468804 "GSM468804"     "mammary epithelial cells" "PKH-negative" "GPL570"
GSM468805 "GSM468805"     "mammary epithelial cells" "PKH-positive" "GPL570"
```

Here the sample annotation was imported from InSilico DB. Of course, a user can also create a dataframe with curated sample annotation, for example with *fix*, and use *updatePhenoData* to store the updated sample annotation in the database.

4.3 Using GEO Datasets

When downloading a GSE, the function *downloadGEOData* checks if the GSE has been curated by GEO staff and been made available as a GDS. If so, the corresponding GDS is also downloaded and when loading the data to the database the curated sample annotation provided by the GDS is stored. For GSE18290 loaded above this is the case:

```
> GDSforGSE(conn, "GSE18290")
```

```
  id_Compendium Experiment experimentDesign    Chip Samples  Tag OrganismNCBIid
1             1   GSE18290               SC GPL2112      16 <NA>           9913
2             1   GSE18290               SC  GPL339      18 <NA>          10090
3             1   GSE18290               SC  GPL570      18 <NA>           9606
  OrganismName     GDS         date_loaded
1   Bos taurus GDS3960 2014-01-20 10:22:08
2 Mus musculus GDS3958 2014-01-20 10:22:08
3 Homo sapiens GDS3959 2014-01-20 10:22:08
```

GSE18290 has actually been split into three different GDSes, one for each species. If a GDS is available, the individual samples are in general well annotated:

```
> head(GSMdescriptions(conn, "GSE18290"))
```

5 Querying the compendium database

6 Use case: building a breast cancer compendium

The compendiumdb package provides a convenient framework to store and analyze a large number of expression datasets from a specific domain of study. Here we create a breast cancer compendium of gene expression datasets that have been generated over the past ten years. The selected GSEs were all measured using Affymetrix Human Genome U133A and/or U133B arrays, i.e., GPL96 and GPL97 platforms, respectively. First download the selected breast cancer datasets from GEO using their GSE identifiers:

```
> gseids <- c("GSE11121", "GSE2990", "GSE7390", "GSE1456")
> for (i in gseids) {
+     downloadGEOdata(i)
+ }
```

Then load the data to the relational database using *loadDataToCompendium*:

```
> for (i in gseids) {
+     loadDataToCompendium(con = conn, GSEid = i)
+ }
```

The datasets loaded to the compendium can be tagged with a specific label such as "breast cancer".

```
> for (i in gseids) {
+     tagExperiment(con = conn, GSEid = i, tag = "breast cancer")
+ }
```

Such a tag can, for example, be used to retrieve datasets of the user's interest from the compendium datase.

To keep the dataset homogeneous, below we analyse *ExpressionSets* of breast cancer datasets from the platform GPL96 (Affymetrix Human Genome U133A Array). The *ExpressionSet* can be created using *createESET* function. Depending on the number of size of the expression data and type of machine, it might take some time in creating *ExpressionSet* from *createESET* function.

```
> tab <- GSEinDB(conn, )
> compendiumDatasets <- tab[which(tab$Tag == "breast cancer"),
+     ]
> for (i in compendiumDatasets$Experiment) {
+     createESET(con = conn, GSEid = i, GPLid = "GPL96")
+ }
> esets = list(esetGSE11121_GPL96 = esetGSE11121_GPL96, esetGSE2990_GPL96 = esetGSE2990_GPL96
+     esetGSE1456_GPL96 = esetGSE1456_GPL96, esetGSE7390_GPL96 = esetGSE7390_GPL96)
> save(esets, file = "breastCancerData_v2.Rdata")
```

The *ExpressionSets* generated by *createESET* contain expression data at the probeset level. In order to perform a functional enrichment analysis, one has to select a single probe if multiple probes map to the same to one gene expression data. This can be achieved by using the *nsFilter* from the genefilter package:

```
> require(genefilter)
> for (i in 1:length(esets)) {
+     annotation(esets[[i]]) <- "hgu133a"
+     esets[[i]] <- nsFilter(esets[[i]], remove.dupEntrez = TRUE,
+         var.func = sd, var.filter = FALSE)$eset
+ }
```

The annotation of samples, i.e., phenodata of the datasets is improper and can be modified manually. The annotated, pre-processed expression datasets can be loaded as follows:

```
> exprs(esets[[1]]) <- log2(exprs(esets[[1]]))
> esets$esetGSE1456_GPL96$grade <- esets$esetGSE1456_GPL96$ELSTON
> for (i in 1:length(esets)) {
+     esets[[i]]$grade <- sub(" ", "", esets[[i]]$grade)
+     esets[[i]] <- esets[[i]][, esets[[i]]$grade %in% c("1", "2",
+         "3")]
+ }
```

6.1 Functional enrichment analysis

To infer biological relevance from a compendium of related expression studies, it is important to understand how datasets in the compendium are functionally related to each other. In this section of the vignette, we illustrate a functional enrichment analysis that reveals the consistency and variation on a geneset level among datasets of the compendium. For this purpose, we selected a number of large breast cancer microarray datasets. Breast cancer is a well-known disease that comprises a diverse and heterogeneous set of subtypes. Identification of variations at the molecular level within one cancer type such as breast cancer has always been challenging. Analysis of a compendium of breast cancer datasets will provide insight in the consistency and variation among cancer types and will help in improving microarray breast cancer event predictions.

In this section we use the predefined gene sets (see c2BroadSets of GSVA package) and identify variation or consistency of the gene sets among breast cancer datasets. We use GSVA package to identify the enrichment of gene sets by comparing grade 1 (g1) and grade 2 (g2) phenotypes in each expression dataset. The package requires gene expression data and collection of gene set as the two main input arguments. Gene expression data of each breast cancer dataset is available in the form of ExpressionSet object as we obtained in the previous section. Once we have both ExpressionSet and GeneSets, we can perform enrichment analysis using the *gsva* function of GSVA package:

```
> require(GSVA)
> require(GSVAdata)
> require(limma)
```

```
> data(c2BroadSets)
> gseids = c("GSE11121", "GSE2990", "GSE7390", "GSE1456")
> fit.eb <- list()
> DEgeneSets <- list()
> adjPvalueCutoff <- 10^-12
> load("fit.eb.Rdata")
> load("DEgeneSets.Rdata")
> tstats <- c()
> for (i in 1:length(fit.eb)) {
+     tstat <- fit.eb[[i]]$t
+     tstats <- cbind(tstats, tstat[, "g3vsg1"])
+ }
> colnames(tstats) = gseids
> tstats <- c()
> for (i in 1:length(fit.eb)) {
+     tstat <- fit.eb[[i]]$t
+     tstats <- cbind(tstats, tstat[, "g3vsg1"])
+ }
> colnames(tstats) <- gseids
> heatmap(tstats[DEgeneSets[[1]][DEgeneSets[[1]]$adj.P.Val < 10^-11,
+     "ID"], ], cexCol = 1, cexRow = 0.6, scale = "none", margins = c(2,
+     10))
```

The behaviour of gene sets in each breast cancer dataset can be analysed by plotting the heatmap of log odd ratio of gene sets as shown in Figure 2.

Acknowledgements

References

Bareke, E., M. Pierre, A. Gaigneaux, B. Meulder, S. Depiereux, N. Habra, and E. Depiereux (2010). PathEx: a novel multi factors based datasets selector web tool. *BMC Bioinformatics 11*(1), 528.

Barrett, T., S. E. Wilhite, P. Ledoux, C. Evangelista, I. F. Kim, M. Tomashevsky, K. A. Marshall, K. H. Phillippy, P. M. Sherman, M. Holko, et al. (2013). Ncbi geo: archive for functional genomics data sets – update. *Nucleic acids research 41*(D1), D991–D995.

Cheng, W., M. Tsai, C. Chang, C. Huang, C. Chen, W. Shu, Y. Lee, T. Wang, J. Hong, C. Li, et al. (2010). Microarray meta-analysis database (M2DB): a uniformly pre-processed, quality controlled, and manually curated human clinical microarray database. *BMC Bioinformatics 11*(1), 421.

Coletta, A., C. Molter, R. Duque, D. Steenhoff, J. Taminau, V. De Schaetzen, S. Meganck, C. Lazar, D. Venet, V. Detours, et al. (2012). InSilico DB genomic datasets hub: an efficient starting point for analyzing genome-wide studies in GenePattern, Integrative Genomics Viewer, and R/Bioconductor. *Genome Biology 13*(11), R104.

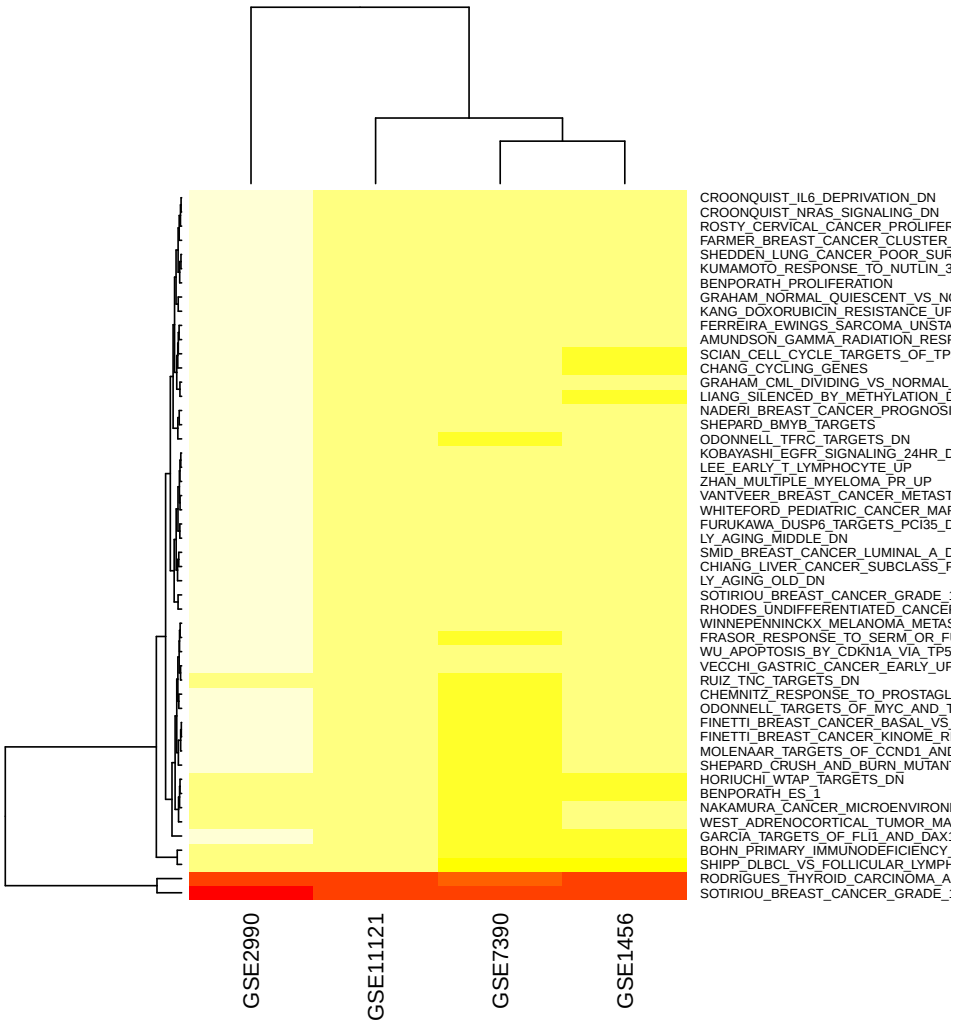

Figure 2: Heatmap of enriched pathways in breast cancer datasets at 0.1 % FDR where x-axis represents GSE ID and y-axis represents gene sets

Helbig, C., R. Gentek, R. A. Backer, Y. de Souza, I. A. Derks, E. Eldering, K. Wagner, D. Jankovic, T. Gridley, P. D. Moerland, et al. (2012). Notch controls the magnitude of T helper cell responses by promoting cellular longevity. *Proceedings of the National Academy of Sciences 109*(23), 9041–9046.

Kilpinen, S., R. Autio, K. Ojala, K. Iljin, E. Bucher, H. Sara, T. Pisto, M. Saarela, R. Skotheim, M. Björkman, et al. (2008). Systematic bioinformatic analysis of expression levels of 17,330 human genes across 9,783 samples from 175 types of healthy and pathological tissues. *Genome Biology 9*(9), R139.

Lacson, R., E. Pitzer, J. Kim, P. Galante, C. Hinske, and L. Ohno-Machado (2010). DSGeo: Software tools for cross-platform analysis of gene expression data in geo. *Journal of Biomedical Informatics 43*(5), 709–715.

Liu, F., J. White, C. Antonescu, D. Gusenleitner, and J. Quackenbush (2011). GCOD-GeneChip oncology database. *BMC Bioinformatics 12*(1), 46.

Pece, S., D. Tosoni, S. Confalonieri, G. Mazzarol, M. Vecchi, S. Ronzoni, L. Bernard, G. Viale, P. G. Pelicci, and P. P. Di Fiore (2010). Biological and molecular heterogeneity of breast cancers correlates with their cancer stem cell content. *Cell 140*(1), 62–73.

Pitzer, E., R. Lacson, C. Hinske, J. Kim, P. A. Galante, and L. Ohno-Machado (2009). Towards large-scale sample annotation in gene expression repositories. *BMC Bioinformatics 10*(Suppl 9), S9.

Rustici, G., N. Kolesnikov, M. Brandizi, T. Burdett, M. Dylag, I. Emam, A. Farne, E. Hastings, J. Ison, M. Keays, et al. (2013). Arrayexpress update – trends in database growth and links to data analysis tools. *Nucleic acids research 41*(D1), D987–D990.

Taminau, J., D. Steenhoff, A. Coletta, S. Meganck, C. Lazar, V. de Schaetzen, R. Duque, C. Molter, H. Bersini, A. Nowé, et al. (2011). inSilicoDb: an R/Bioconductor package for accessing human Affymetrix expert-curated datasets from GEO. *Bioinformatics 27*(22), 3204–3205.

Xia, X., M. McClelland, S. Porwollik, W. Song, X. Cong, and Y. Wang (2009). WebArrayDB: cross-platform microarray data analysis and public data repository. *Bioinformatics 25*(18), 2425–2429.

Xie, D., C.-C. Chen, L. M. Ptaszek, S. Xiao, X. Cao, F. Fang, H. H. Ng, H. A. Lewin, C. Cowan, and S. Zhong (2010). Rewirable gene regulatory networks in the preimplantation embryonic development of three mammalian species. *Genome Research 20*(6), 804–815.

```
> sessionInfo()

R version 3.0.2 (2013-09-25)
Platform: i386-w64-mingw32/i386 (32-bit)

locale:
[1] LC_COLLATE=English_United Kingdom.1252
[2] LC_CTYPE=English_United Kingdom.1252
[3] LC_MONETARY=English_United Kingdom.1252
```

```
[4] LC_NUMERIC=C
[5] LC_TIME=English_United Kingdom.1252

attached base packages:
[1] parallel  stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
 [1] GSVAdata_0.99.11     hgu95a.db_2.10.1     GSVA_1.10.2
 [4] GSEABase_1.24.0      graph_1.40.1         annotate_1.40.0
 [7] hgu133a.db_2.10.1    org.Hs.eg.db_2.10.1  AnnotationDbi_1.24.0
[10] genefilter_1.44.0    inSilicoDb_1.10.0    rjson_0.2.13
[13] compendiumDB_0.86    GEOmetadb_1.22.0     RSQLite_0.11.4
[16] GEOquery_2.28.0      limma_3.18.9         Biobase_2.22.0
[19] BiocGenerics_0.8.0   RMySQL_0.9-3         DBI_0.2-7

loaded via a namespace (and not attached):
[1] IRanges_1.20.6  RCurl_1.95-4.1  splines_3.0.2   stats4_3.0.2
[5] survival_2.37-4 tools_3.0.2     XML_3.98-1.1    xtable_1.7-1
```