GEO

Mikhail Dozmorov

2016-10-03

Contents

Reading the NCBI's GEO microarray SOFT files in R/BioConductor

GEO expression omnibus, https://www.ncbi.nlm.nih.gov/geo/, is the largest repository of gene expression data. It may have the data you're looking for, saving you time and money to do the experiment yourself.

We will discuss how to load GEO SOFT format microarray data from the Gene Expression Omnibus database (GEO) (hosted by the NCBI) into R/BioConductor. SOFT stands for Simple Omnibus Format in Text. There are actually four types of GEO SOFT file available:

- `GEO Platform (GPL)` - These files describe a particular type of microarray. They are annotation files.
- `GEO Sample (GSM)` - Files that contain all the data from the use of a single chip. For each gene there will be multiple scores including the main one, held in the VALUE column.
- `GEO Series (GSE)` - Lists of GSM files that together form a single experiment.
- `GEO Dataset (GDS)` - These are curated files that hold a summarized combination of a GSE file and its GSM files. They contain normalized expression levels for each gene from each sample (i.e. just the VALUE field from the GSM file).

As long as you just need the expression level then a GDS file will suffice. If you need to dig deeper into how the expression levels were calculated, you'll need to get all the GSM files instead (which are listed in the GDS or GSE file)

Installing GEOquery. Assuming you are running a recent version of BioConductor (1.8 or later) you should be able to install it from within R as follows:

```
source("http://www.bioconductor.org/biocLite.R")
biocLite("GEOquery")
```

Loading a GDS file with GEOquery. Here is a quick introduction to how to load a GDS file, and turn it into an expression set object:

```
library(Biobase)
library(GEOquery)
# Download GDS file, put it in the current directory, and load it:
gds858 <- getGEO("GDS858", destdir = "../data/")
```

```
## Using locally cached version of GDS858 found here:
## ../data//GDS858.soft.gz
```

```
# Or, open an existing GDS file (even if its compressed):
gds858 <- getGEO(filename = "../data/GDS858.soft.gz", destdir = "../data/")
```

The SOFT file is available in compressed form here GDS858.soft.gz, but GEOquery takes care of finding this file for you and unzipping it automatically.

There are two main things the GDS object gives us, meta data (from the file header) and a table of expression data. These are extracted using the `Meta` and `Table` functions. First lets have a look at the metadata:

```
Meta(gds858)$channel_count
```

```
## [1] "1"
```

```
Meta(gds858)$description
```

```
##  [1] "Comparison of lung epithelial Calu-3 cells infected with the mucoid alginate-producing FRD1 st
##  [2] "FRD1"
##  [3] "FRD440"
##  [4] "FRD875"
##  [5] "FRD1234"
##  [6] "uninfected"
##  [7] "control"
##  [8] "mucoid"
##  [9] "motile"
## [10] "non-mucoid, non-motile"
```

```
Meta(gds858)$feature_count
```

```
## [1] "22283"
```

```
Meta(gds858)$platform
```

```
## [1] "GPL96"
```

```
Meta(gds858)$sample_count
```

```
## [1] "19"
```

```
Meta(gds858)$sample_organism
```

```
## [1] "Homo sapiens"
```

```
Meta(gds858)$sample_type
```

```
## [1] "RNA"
```

```
Meta(gds858)$title
```

```
## [1] "Mucoid and motile Pseudomonas aeruginosa infected lung epithelial cell comparison"
```

```
Meta(gds858)$type
```

```
##  [1] "Expression profiling by array" "infection"                    "infection"
##  [7] "genotype/variation"            "genotype/variation"           "genotype/variation"
```

Useful stuff, and now the expression data table:

```
colnames(Table(gds858))
```

```
##  [1] "ID_REF"     "IDENTIFIER" "GSM14498"   "GSM14499"   "GSM14500"   "GSM14501"   "GSM14513"   "GSM
## [17] "GSM14505"   "GSM14509"   "GSM14510"   "GSM14511"   "GSM14512"
```

```
Table(gds858)[1:10, 1:6]
```

```
##       ID_REF IDENTIFIER GSM14498 GSM14499 GSM14500 GSM14501
## 1  1007_s_at       DDR1   3736.9   3811.0   3699.6   3897.6
## 2    1053_at       RFC2    343.0    500.3    288.3    341.3
## 3     117_at      HSPA6    120.9     34.3    145.8    110.5
```

```
## 4     121_at       PAX8   1523.8   1281.1   1281.9   1493.4
## 5  1255_g_at     GUCA1A     51.6     15.9     45.9      8.1
## 6    1294_at       UBA7    253.2    164.8    200.0    205.2
## 7    1316_at       THRA    199.6    250.7    290.3    218.6
## 8    1320_at     PTPN21     81.7     13.4     13.9     88.7
## 9  1405_i_at       CCL5     18.9      5.6     11.0      9.5
## 10   1431_at     CYP2E1     99.7     74.5     72.6    114.8
```

Now, lets turn this GDS object into an expression set object (using base 2 logarithms) and have a look at it:

```
eset <- GDS2eSet(gds858, do.log2 = TRUE)
```

```
## File stored at:
## /var/folders/tq/q7zhthbj7l574j7qwf1sm60m0000gq/T//RtmpEYyAQI/GPL96.annot.gz
```

```
eset
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 22283 features, 19 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: GSM14498 GSM14499 ... GSM14512 (19 total)
##   varLabels: sample infection genotype/variation description
##   varMetadata: labelDescription
## featureData
##   featureNames: 1007_s_at 1053_at ... AFFX-TrpnX-M_at (22283 total)
##   fvarLabels: ID Gene title ... GO:Component ID (21 total)
##   fvarMetadata: Column labelDescription
## experimentData: use 'experimentData(object)'
##   pubMedIds: 15494517
## Annotation:
```

```
featureNames(eset)[1:10]
```

```
##  [1] "1007_s_at" "1053_at"   "117_at"    "121_at"    "1255_g_at" "1294_at"   "1316_at"   "1320_at"
```

```
sampleNames(eset)[1:10]
```

```
##  [1] "GSM14498" "GSM14499" "GSM14500" "GSM14501" "GSM14513" "GSM14514" "GSM14515" "GSM14516" "GSM145
```

GEOquery does an excellent job of extracting the phenotype data, as you can see:

```
pData(eset)
```

```
##            sample  infection     genotype/variation
## GSM14498 GSM14498 uninfected                control Value for GSM14498: CONTROL 1; src: Calu-3 human
## GSM14499 GSM14499 uninfected                control Value for GSM14499: CONTROL 2; src: Calu-3 human
## GSM14500 GSM14500 uninfected                control Value for GSM14500: CONTROL 3; src: Calu-3 human
## GSM14501 GSM14501 uninfected                control Value for GSM14501: CONTROL 4; src: Calu-3 human
## GSM14513 GSM14513     FRD875 non-mucoid, non-motile  Value for GSM14513: FRD875 1; src: Calu-3 human
## GSM14514 GSM14514     FRD875 non-mucoid, non-motile  Value for GSM14514: FRD875 2; src: Calu-3 human
## GSM14515 GSM14515     FRD875 non-mucoid, non-motile  Value for GSM14515: FRD875 3; src: Calu-3 human
## GSM14516 GSM14516     FRD875 non-mucoid, non-motile  Value for GSM14516: FRD875 4; src: Calu-3 human
## GSM14506 GSM14506    FRD1234 non-mucoid, non-motile Value for GSM14506: FRD1234 1; src: Calu-3 human
## GSM14507 GSM14507    FRD1234 non-mucoid, non-motile Value for GSM14507: FRD1234 2; src: Calu-3 human
## GSM14508 GSM14508    FRD1234 non-mucoid, non-motile Value for GSM14508: FRD1234 3; src: Calu-3 human
## GSM14502 GSM14502       FRD1                 mucoid    Value for GSM14502: FRD1 1; src: Calu-3 human
```

```
## GSM14503 GSM14503       FRD1                mucoid    Value for GSM14503: FRD1 2; src: Calu-3 human
## GSM14504 GSM14504       FRD1                mucoid    Value for GSM14504: FRD1 3; src: Calu-3 human
## GSM14505 GSM14505       FRD1                mucoid    Value for GSM14505: FRD1 4; src: Calu-3 human
## GSM14509 GSM14509     FRD440                motile  Value for GSM14509: FRD440 1; src: Calu-3 human
## GSM14510 GSM14510     FRD440                motile  Value for GSM14510: FRD440 2; src: Calu-3 human
## GSM14511 GSM14511     FRD440                motile  Value for GSM14511: FRD440 3; src: Calu-3 human
## GSM14512 GSM14512     FRD440                motile  Value for GSM14512: FRD440 4; src: Calu-3 human
```

```
pData(eset)$infection
```

```
##  [1] uninfected uninfected uninfected uninfected FRD875     FRD875     FRD875     FRD875     FRD1234
## Levels: FRD1 FRD1234 FRD440 FRD875 uninfected
```

```
pData(eset)$"genotype/variation"
```

```
##  [1] control                control                control                control                non-
## [10] non-mucoid, non-motile non-mucoid, non-motile mucoid                 mucoid                 muco
## [19] motile
## Levels: control motile mucoid non-mucoid, non-motile
```

As with any expression set object, its easy to pull out a subset of the data:

```
eset["1320_at", "GSM14504"]
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 1 features, 1 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: GSM14504
##   varLabels: sample infection genotype/variation description
##   varMetadata: labelDescription
## featureData
##   featureNames: 1320_at
##   fvarLabels: ID Gene title ... GO:Component ID (21 total)
##   fvarMetadata: Column labelDescription
## experimentData: use 'experimentData(object)'
##   pubMedIds: 15494517
## Annotation:
```

In addition to loading a GDS file to get the expression levels, you can also load the associated platform annotation file. You can find this out from the GDS858 meta information:

```
Meta(gds858)$platform  # Which platform?
```

```
## [1] "GPL96"
```

Now let's load up the GPL file and have a look at it (its a big file, about 12 MB, so this takes a while!):

```
# Download GPL file, put it in the current directory, and load it:
gpl96 <- getGEO("GPL96", destdir = "../data/")
```

```
## Using locally cached version of GPL96 found here:
## ../data//GPL96.soft
```

```
# Or, open an existing GPL file:
gpl96 <- getGEO(filename = "../data/GPL96.soft", destdir = "../data/")
```

As with the GDS object, we can use the Meta and Table functions to extract information:

```
Meta(gpl96)$title
```

```
## [1] "[HG-U133A] Affymetrix Human Genome U133A Array"
```

```
colnames(Table(gpl96))
```

```
##  [1] "ID"                                "GB_ACC"                            "SPOT_ID"
##  [7] "Sequence Source"                   "Target Description"                "Representative Public ID
## [13] "RefSeq Transcript ID"              "Gene Ontology Biological Process" "Gene Ontology Cellular C
```

Lets look at the first four columns, for the first ten genes:

```
Table(gpl96)[1:10, 1:4]
```

```
##           ID GB_ACC SPOT_ID Species Scientific Name
## 1  1007_s_at U48705    <NA>            Homo sapiens
## 2    1053_at M87338    <NA>            Homo sapiens
## 3     117_at X51757    <NA>            Homo sapiens
## 4     121_at X69699    <NA>            Homo sapiens
## 5  1255_g_at L36861    <NA>            Homo sapiens
## 6    1294_at L13852    <NA>            Homo sapiens
## 7    1316_at X55005    <NA>            Homo sapiens
## 8    1320_at X79510    <NA>            Homo sapiens
## 9  1405_i_at M21121    <NA>            Homo sapiens
## 10   1431_at J02843    <NA>            Homo sapiens
```

You can also use Bioconductor annotation file for GPL96/HG-U133A by using `library(hgu133a)`.