Creating reference datasets:
The Broad Connectivity Map (v1)

Thomas Sandmann

October 13, 2014

Contents

1 Introduction **2**

2 Analyzing the Broad Connectivity map (v.1) data **2**

1 Introduction

Public repositories, such as ArrayExpress or GEO provide access to many published expression profiling datasets, featuring perturbations in many different organisms, model systems and conditions.

The Atlas search engine offers a simple way to identify perturbation experiments of interest in the ArrayExpress repository.

This vignette shows how to obtain and process raw microarray data from a large-scale drug perturbation study performed in human cells, the `Connectivity Map` dataset (version 1), released by Lamb and co-workers in 2006. Similar workflows can be used to download an process many other publically available datasets.

In this study, the researchers treated multiple human cell lines with 164 distinct small molecules or matched controls. In total, 564 samples were generated, RNA extracted, labeled and hybridized either to A-AFFY-113 Affymetrix (HT_HG-U133A) or A-AFFY-33 Affymetrix (HG-U133A) microarrays.

The raw data for this study is available from ArrayExpress under accession `E-GEOD-5258` . The raw .cel files and the array annotations can be downloaded and compiled into a suitable `eSet` objects using the ArrayExpress Bioconductor package. Alternatively, the final RData object can be downloaded directly from ArrayExpress.

Please note that this is a large dataset and executing the following code will download more than 700 MB of data.

2 Analyzing the Broad Connectivity map (v.1) data

Data download and normalization

A call to the `ArrayExpress` function will retrieve the raw data for study E-GEOD-6907 from ArrayExpress. (As this is a large dataset, this might take while...)

```
> library(ArrayExpress)
> GEOD5258.batch <- ArrayExpress("E-GEOD-6907")
```

As this experiment was performed on two different array platforms, a list with two affyBatch objects is returned, one for each array platform.

We normalize each object separately using the `rma` function from the affy package.

```
> library( affy )
> length( GEOD5258.batch )
> GEOD5258.eSets <- lapply( GEOD5258.batch, rma )
```

The `mapNmerge` function from the gCMAP package averages the expression values different probes for the same gene by mapping them to Entrez ids. Alternatively, the `nsFilter` function from the genefilter package could be used.

```
> GEOD5258.eSets <- lapply( GEOD5258.eSets, mapNmerge)
```

Now that we have mapped the expression values to Entrez Ids, we can combine the two ExpressionSets into one

```
> GEOD5258.eSet <- mergeCMAPs( GEOD5258.eSets[[1]], GEOD5258.eSets[[2]] )
```

Defining perturbation experiments and performing differential expression analysis

The ArrayExpress dataset is associated with extensive sample annotation information, available in the `phenoData` slot of the `ExpressionSet`. Experimental factors are marked with the `Factor` prefix in the column name.

```
> head( pData(GEOD5258.eSet ))
> conditions <- grep("^Factor", varLabels( GEOD5258.eSet ), value=TRUE)
> conditions
```

In this case, we are interested in studying the effect of the different compounds, which are specified in the column of the `phenoData` slot. Controls are annotated with the Compound level `none` .

```
> unique( pData(  GEOD5258.eSet )$Factor.Value..Compound.)
```

To associate drug perturbation with their matched controls, we require that control experiments must have been performed in the same `CellLine` and with the same `Vehicle` . With this information, the `splitPerturbations` function from the gCMAP package can group treatment and perurbation samples into individual experiments of interest. Each of these experimental instances is returned in a separate `ExpressionSet`, grouped in the GEOD5258.list list.

```
> GEOD5258.list <- splitPerturbations( GEOD5258.eSet,
                       factor.of.interest="Compound",
                       control="none",
                       controlled.factors=c("CellLine", "Vehicle", "Time")
)
```

To track the experimental conditions assayed in each perturbation experiment, the first line (containing the perturbation) is extracted from each `phenoData` slot and deposited in a data.frame with one row for each perturbation / `ExpressionSet` in GEOD5258.list.

```
> anno <- t(sapply( GEOD5258.list, function(x) pData(x)[1,conditions]))
> anno <- apply( anno, 2, unlist)
> anno <- data.frame( anno )
> colnames( anno ) <- c("CellLine", "Vehicle", "Compound", "Time", "Dose")
```

The `generate_gCMAP_NChannelSet` function performs differential expression analysis (using limma) separately for each `ExpressionSet` in the list. It returns an `NChannelSet` object containing the log2 fold change, raw p-values and z-scores for all experiments.

```
> GEOD5258.ref <- generate_gCMAP_NChannelSet( GEOD5258.list,
                       uids=1:length( GEOD5258.list ),
                       sample.annotation=anno)
> pData(  GEOD5258.ref)[10:15,]
```

This object, containing the differential expression results for 12701 genes from 214 different perturbation experiments and sample-level annotations in its `phenoData` slot, is now ready to be used as a reference dataset by gCMAPWeb.

Inducing gene sets

If required, we can apply a threshold to one channel of the `NChannelSet` and define sets of differentially up- and down-regulated genes. For example, the following command applies a z-score cutoff of >3 or <-3 to each experiment and stores the results in a sparse-matrix within a `CMAPCollection`.

```
> GEOD5258.sets <- induceCMAPCollection( GEOD5258.ref, element="z", higher=3, lower=-3 )
> head( setSizes( GEOD5258.sets ) )
```

```
> sessionInfo()

R version 3.1.1 Patched (2014-09-24 r66678)
Platform: i386-w64-mingw32/i386 (32-bit)

locale:
[1] LC_COLLATE=C
[2] LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
 [1] tools     stats4    parallel  stats     graphics  grDevices utils
 [8] datasets  methods   base

other attached packages:
 [1] gCMAPWeb_1.6.0       Rook_1.0-9           yaml_2.1.13
 [4] gCMAP_1.10.0         limma_3.22.0         GSEABase_1.28.0
 [7] graph_1.44.0         annotate_1.44.0      XML_3.98-1.1
[10] AnnotationDbi_1.28.0 GenomeInfoDb_1.2.0   IRanges_2.0.0
[13] S4Vectors_0.4.0      Biobase_2.26.0       BiocGenerics_0.12.0
[16] brew_1.0-6

loaded via a namespace (and not attached):
 [1] DBI_0.3.1         GSEAlm_1.26.0     Matrix_1.1-4      RSQLite_0.11.4
 [5] genefilter_1.48.0 grid_3.1.1        hwriter_1.3.2     lattice_0.20-29
 [9] splines_3.1.1     survival_2.37-7   xtable_1.7-4
```