BIOINFORMATICS APPLICATIONS NOTE Vol. 23 no. 14 2007, pages 1846–1847
doi:10.1093/bioinformatics/btm254

Gene expression

GEOquery: a bridge between the Gene Expression Omnibus (GEO) and BioConductor

Sean Davis* and Paul S. Meltzer

Genetics Branch, National Cancer Institute, National Institutes of Health, Bethesda, MD, USA

Received on April 3, 2007; revised on April 30, 2007; accepted on May 4, 2007

Advance Access publication May 12, 2007

Associate Editor: Joaquin Dopazo

ABSTRACT

Microarray technology has become a standard molecular biology tool. Experimental data have been generated on a huge number of organisms, tissue types, treatment conditions and disease states. The Gene Expression Omnibus (Barrett *et al*., 2005), developed by the National Center for Bioinformatics (NCBI) at the National Institutes of Health is a repository of nearly 140 000 gene expression experiments. The BioConductor project (Gentleman *et al*., 2004) is an open-source and open-development software project built in the R statistical programming environment (R Development core Team, 2005) for the analysis and comprehension of genomic data. The tools contained in the BioConductor project represent many state-of-the-art methods for the analysis of microarray and genomics data. We have developed a software tool that allows access to the wealth of information within GEO directly from BioConductor, eliminating many the formatting and parsing problems that have made such analyses labor-intensive in the past. The software, called GEOquery, effectively establishes a bridge between GEO and BioConductor. Easy access to GEO data from BioConductor will likely lead to new analyses of GEO data using novel and rigorous statistical and bioinformatic tools. Facilitating analyses and meta-analyses of microarray data will increase the efficiency with which biologically important conclusions can be drawn from published genomic data.

Availability: GEOquery is available as part of the BioConductor project.

Contact: sdavis2@mail.nih.gov

1 OVERVIEW OF GEO AND GEOQUERY

The NCBI Gene Expression Omnibus (GEO) serves as a public repository for a wide range of high-throughput experimental data. These data include single and dual channel microarray-based experiments measuring mRNA, genomic DNA and protein abundance as well as non-array techniques, such as serial analysis of gene expression (SAGE) and mass spectrometry proteomic data. Currently, nearly 140 000 samples and over 3000 different microarray platforms are represented in GEO. At the most basic level of organization of GEO, there are four basic entities. The first three (Sample, Platform and Series) are supplied by users; the fourth, the dataset, is compiled and curated by GEO staff from the user-submitted data.[1] GEO platform files (accessions like GPLxxxx where 'x' is a number) describe an array layout and content while GEO samples (accessions like GSMxxxx) describe the actual results of a single hybridization. An entire experiment, including information about all hybridizations and their associated platform descriptions can be accessed as a GEO series (GSExxxx). Finally, GEO staff have selectively curated GEO series into a more compact format, the GEO dataset (GDSxxxx) that includes a single spreadsheet of 'final' values and accompanying rich sample annotation. The data can be accessed from GEO in a proprietary format they term SOFT. Each of these entity types has a corresponding class representation defined by GEOquery. Methods for accessing the various pieces of the GEO entity are available from GEOquery, making access to the data easy and quick once the data from GEO has been parsed into the GEOquery data structures.

2 USING GEOQUERY

The primary goal of GEOquery is to download and parse the SOFT format files from GEO, maintaining all of the information contained in the GEO records. The design of GEOquery makes accessing data from GEO very simple. There is only one command that is needed, `getGEO`. As an example, GEOquery is used here to demonstrate how one would get the data associated with GDS1, a GEO dataset.

```
> library(GEOquery)
```

This command loads the GEOquery library into R.

```
> gds <- getGEO('GDS1')

File stored at:
/tmp/RtmpitNEHc/GDS1.soft.gz
parsing geodata
parsing subsets
ready to return
```

Now, the R variable `gds` contains the GEOquery data structure (of class *GDS*) that represents the GDS1 entry from GEO. Note that the filename used to store the download was output to the screen for later use to a call to

*To whom correspondence should be addressed.

[1]See http://www.ncbi.nih.gov/geo for more information.

getGEO(filename = ...). The same command will obtain any other GEO accession, such as GSM3, a GEO sample.

```
> gsm <- getGEO('GSM3')
File stored at:
/tmp/RtmpitNEHc/GSM3.soft
```

2.1 GEOquery data structures

Two of the most powerful features of GEOquery are the custom data structures and associated methods that organize the data from a GEO download. The GEOquery data structures can be described in two broad groups that are analogous to the structure used by GEO to deliver the data. The first group, comprising *GDS*, *GPL* and *GSM*, all have similar GEO SOFT format structure and the associated GEOquery methods have similar effects on each. Each of these classes is comprised of a metadata header, taken nearly verbatim from the SOFT format header and including information pertaining to the entire GEO object, and a GEODataTable. The GEODataTable has two simple parts, a Columns part which describes the column headers and a table part which typically contains a 2D table of information. Here is the truncated output of some of the accessors applied to the GEO sample that was stored in the previous section.

```
> Meta(gsm)[1:3]
$channel_count
[1] '1'
$contact_address
[1] '6 Center Drive'
$contact_city
[1] 'Bethesda'
> Table(gsm)[1:3, c(2, 6, 7)]
Signal_Raw Norm_Value Const
1  138392.6  395070.1 39542
2  100973.5  395070.1 39542
3  118994.0  395070.1 39542
> Columns(gsm)[c(2, 6, 7), ]
      Column         Description
2 SIGNAL_RAW          raw signal
6 NORM_VALUE normalization value
7      CONST      constant value
```

The *GPL* behaves exactly as the *GSM* class, although the `GSM` class contains gene expression data in its GEODataTable, while the GPL class contains microarray feature information. However, the GDS class has a bit more information associated with the `Columns` method, as it contains the annotation associated with each array:

```
> Columns(gds)[1:3, 1:3]
 sample gender              tissue
1  GSM3   male              testis
2  GSM4   male              testis
3  GSM5   male gonadectomized male
```

The *GSE* is a composite class. A GSE entry from GEO can represent an arbitrary number of samples hybridized on an arbitrary number of platforms. The *GSE* has a metadata section, just like the other classes. However, it does not have a GEODataTable. Instead, it contains two lists, accessible using `GPLList` and `GSMList`, that are lists of *GPL* and *GSM* objects, respectively. The use of class accessors, combined with R's powerful data manipulation capabilities make it possible to quickly and easily access and reshape the GSE data into any form appropriate for further analysis.

2.2 Converting to BioConductor data structures

GEO datasets are quite similar to the *limma* package (Smyth, 2005) data structure *MAList* and to the *Biobase* package (Gentleman *et al*., 2004) data structure *ExpressionSet*. Therefore, there are two functions, `GDS2MA` and `GDS2eSet` that convert from the *GDS* to either of the two data structures.

As a simple example, converting `gds` object from above into an *ExpressionSet* is as simple as:

```
> eset <- GDS2eSet(gds, do.log2 = TRUE)
File stored at:
/tmp/RtmpitNEHc/GPL5.soft
```

Now, `eset` is an *ExpressionSet* that contains the same information as in the GEO dataset, including the sample information, log-transformed data and probe annotation from the associated GPL. Functions to convert to other BioConductor data structures are also available.

3 CONCLUSION

The GEOquery package provides a bridge between the BioConductor analysis tools and the vast public data resources contained in the NCBI GEO repositories. By maintaining the full richness of the GEO data rather than focusing on getting only the 'numbers', it is possible to integrate GEO data into current Bioconductor data structures and to perform analyses on that data quite quickly and easily or to export the data into any number of formats for use by other tools or for local storage and data mining. GEOquery will hopefully open GEO data more fully to the bioinformatics community at large.

ACKNOWLEDGEMENTS

We would like to thank the staff at NCBI GEO for valuable input and support during the development process.

Conflict of Interest: none declared.

REFERENCES

Barrett,T. *et al*. (2005) Ncbi geo: mining millions of expression profiles–database and tools. *Nucleic Acids Res*., **33**, 562–566.

Gentleman,R.C. *et al*. (2004) Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol*., **5**.

R Development Core Team (2005) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. ISBN 3-900051-07-0.

Smyth,G.K. (2005) Limma: linear models for microarray data. In: Gentleman,R. *et al*. (eds) *Bioinformatics and Computational Biology Solutions using R and Bioconductor*. Springer, New York, pp. 397–420.