Using the geneRecommender Package

Greg Hather

September 12, 2024

1 What is geneRecommender?

geneRecommender is a package used to identify coexpressed genes in microarray data. In particular, the package ranks genes according to how strongly they correlate with a set of query genes over the experiments for which the query genes behave in a similar fashion. The query genes, which are chosen by the user, are intended to be genes known to be involved in a process of interest. Since genes which are coexpressed are more likely to be functionally related, the ranked list produced by *geneRecommender* will suggest other genes which are involved in or related to the process of interest. The package is an implementation of the Gene Recommender algorithm developed by Owen et al [1]. *geneRecommender* is designed to work even with a significant amount of missing data.

2 An Example

Suppose we are analyzing the `geneData` dataset in the *Biobase* package. The data consists of gene expression levels for 500 probesets across 26 experiments. As a fictional example, suppose that the probe sets "31613_at", "31712_at", and "31497_at" correspond to genes known to have closely related functions. Suppose that we wish to use the set for a *geneRecommender* query. The first step is to normalize the dataset with the `gr.normalize` function. The second step is to apply the `gr.main` function to the normalized data.

```
> library(geneRecommender)
> data(geneData)
> my.query <- c("31613_at", "31712_at", "31497_at")
> normalized.data <- gr.normalize(geneData)
> gr.main(normalized.data, my.query, ngenes = 10)

$main.result
      [,1]              [,2]
 [1,] "31712_at"        "Within query"
 [2,] "31736_at"        "Not within query"
 [3,] "31613_at"        "Within query"
 [4,] "31668_f_at"      "Not within query"
 [5,] "31497_at"        "Within query"
 [6,] "31558_at"        "Not within query"
 [7,] "31485_at"        "Not within query"
 [8,] "31679_at"        "Not within query"
 [9,] "31455_r_at"      "Not within query"
[10,] "AFFX-LysX-3_at"  "Not within query"
```

The result is a list containing a single item named `main.result`. `main.result` is a matrix with `ngenes` rows and two columns. The first column tabulates the top `ngenes` identified by the algorithm, listed in the order of decreasing score. The second column identifies the genes in the result that were in the query.

3 Cross Validation

In some cases, the user will be interested in judging the performance of *geneRecommender* for a given query. To this end, the function `gr.cv` performs leave-one-out cross validation. The input for `gr.cv` is the normalized dataset and the query. The output for `gr.cv` is a vector containing the rank of each element in the query produced by applying `gr.main` to the query with that element removed.

```
> gr.cv(normalized.data, my.query)
```

```
[1]  6 10  9
```

In addition to measuring performance, the results of the cross validation can be used to determine if some element(s) in the query might not belong. If one of the elements in the above vector had been very large, one would suspect that the associated gene was regulated differently than the other genes in the query.

4 Additional Options

For users interested in modifying the workings of the algorithm or seeing additional output, the *geneRecommender* package is fairly accommodating. This is explained in the following subsections, some of which use notation from the Owen et al paper [1].

4.1 Normalization

`gr.normalize` normalizes the dataset so that for each gene, the normalized expression values of the experiments are distributed uniformly between -1 and 1. However, alternative normalizations are certainly possible, either by writing one's own normalization function, or by further processing the output of `gr.normalize`. For example, the data could be normalized so that for each gene, the normalized expression values of the experiments have a standard normal distribution. This can be easily done by applying qnorm to the result of `gr.normalize`.

```
> normal.normalized.data <- qnorm((normalized.data + 1)/2)
> gr.main(normal.normalized.data, my.query, ngenes = 10)
```

This type of normalization may be better, but no one knows with certainty at the time of this writing.

4.2 Scoring function

The scoring function allows the algorithm to choose how many experiments to include in the calculation. It is a function applied to the ranks in $S_G(i)$ of the query genes. The algorithm selects the number of experiments which minimizes the result of the scoring function, breaking ties in favor of the largest experiment set. The default scoring function is `median`, which is the same choice used in the paper. However, other choices of the scoring function are possible.

```
> my.fun.1 <- function(input.vector){
+    sum(input.vector^(1/2), na.rm = T)
+ }
> gr.main(normalized.data, my.query, ngenes = 10, fun = my.fun.1)
```

This type of scoring function may be better, but no one knows with certainty at the time of this writing.

4.3   Including all experiments

Suppose that out of curiosity, the user wishes to include ALL experiments in the calculation. Based on the discussion in the previous subsection, it should be apparent that setting `fun` to be a constant function will achieve this goal.

```
> my.fun.2 <- function(input.vector){
+    1
+ }
> gr.main(normalized.data, my.query, ngenes = 10, fun = my.fun.2)
```

4.4   Extra output

When the `gr.main` parameter `extra` is set to TRUE, the function will calculate several additional items and include these items in the result. Thus, the function will take slightly longer to execute.

```
> options(digits = 2)
> gr.main(normalized.data, my.query, ngenes = 10, extra = T)

$main.result
        [,1]                [,2]
 [1,] "31712_at"          "Within query"
 [2,] "31736_at"          "Not within query"
 [3,] "31613_at"          "Within query"
 [4,] "31668_f_at"        "Not within query"
 [5,] "31497_at"          "Within query"
 [6,] "31558_at"          "Not within query"
 [7,] "31485_at"          "Not within query"
 [8,] "31679_at"          "Not within query"
 [9,] "31455_r_at"        "Not within query"
[10,] "AFFX-LysX-3_at"    "Not within query"

$fifty.percent.recall
[1] 3

$experiments.included
 [1] "A" "B" "D" "E" "G" "H" "M" "N" "P" "R" "S" "T" "U" "V" "W"

$experiments.excluded
 [1] "C" "F" "I" "J" "K" "L" "O" "Q" "X" "Y" "Z"

$s.g.i
     31712_at        31736_at        31613_at        31668_f_at        31497_at
         0.49            0.48            0.47            0.47            0.47
```

```
         31558_at         31485_at         31679_at        31455_r_at AFFX-LysX-3_at
             0.46             0.46             0.44             0.44             0.43


$z.g.i
         31712_at         31736_at         31613_at         31668_f_at         31497_at
              4.7              4.6              4.5              4.5              4.5
         31558_at         31485_at         31679_at        31455_r_at AFFX-LysX-3_at
              4.4              4.4              4.2              4.2              4.1


$contribution
                  A     B     D     E     G     H     M     N     P     R     S     T
31712_at       0.53 0.590 0.28 0.200 0.78 0.78 0.33 0.33 0.377 0.65 0.20 0.92
31736_at       0.59 0.422 0.15 0.173 0.58 0.85 0.38 0.47 0.377 0.71 0.42 0.85
31613_at       0.53 0.365 0.18 0.120 0.78 0.65 0.43 0.38 0.244 0.65 0.33 0.92
31668_f_at     0.59 0.478 0.21 0.146 0.85 0.78 0.43 0.16 0.200 0.71 0.42 0.92
31497_at       0.53 0.534 0.15 0.093 0.85 0.78 0.58 0.16 0.333 0.47 0.47 0.78
31558_at       0.65 0.703 0.24 0.040 0.44 0.37 0.63 0.38 0.466 0.59 0.38 0.85
31485_at       0.59 0.422 0.21 0.013 0.78 0.85 0.43 0.16 0.422 0.47 0.55 0.78
31679_at       0.59 0.365 0.11 0.013 0.78 0.85 0.28 0.20 0.466 0.78 0.42 0.85
31455_r_at     0.48 0.646 0.21 0.120 0.71 0.58 0.38 0.29 0.022 0.71 0.42 0.92
AFFX-LysX-3_at 0.59 0.028 0.21 0.120 0.85 0.65 0.38 0.33 0.244 0.53 0.51 0.85
                  U     V     W
31712_at       0.24 0.92 0.179
31736_at       0.24 0.92 0.081
31613_at       0.33 0.92 0.212
31668_f_at     0.47 0.55 0.114
31497_at       0.55 0.63 0.081
31558_at       0.29 0.78 0.081
31485_at       0.29 0.70 0.179
31679_at       0.24 0.63 0.081
31455_r_at     0.16 0.92 0.081
AFFX-LysX-3_at 0.38 0.70 0.049
```

fifty.percent.recall is the number of genes found at 50 percent recall.

experiments.included is a vector containing the names of the experiments included in the analysis.

experiments.excluded is a vector containing the names of the experiments excluded from the analysis.

s.g.i represents the values $S_G(i)$. s.g.i is an array used as a measure of biological significance for each gene. The output is ranked by this quantity.

z.g.i represents the values $Z_G(i)$. z.g.i is an array used as a measure of statistical significance for each gene.

contribution is a matrix indicating the contribution of each experiment to each gene result. For a given gene and a given experiment, contribution indicates how strongly the experiment suggests that the gene should be high ranking. Using notation from the article, contribution is defined as $\bar{Y}_{Q,j} \times Y_{ij}$.

4

5 Notes

The results from this package will differ somewhat from the results generated from the code used in [1]. This is because *geneRecommender* uses an incremental method for determining the number of experiments to include, whereas the code described in [1] uses a less accurate grid approach. Also, for the sake of robustness, *geneRecommender* sets $\bar{Y}_{Q,j}$ to be $median(Y_{i,j} : i \in Q)$ rather than $mean(Y_{i,j} : i \in Q)$.

6 Session Information

The version number of R and packages loaded for generating the vignette were:

- R version 4.4.1 (2024-06-14), `x86_64-pc-linux-gnu`

- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`

- Time zone: `Etc/UTC`

- TZcode source: `system (glibc)`

- Running under: `Ubuntu 24.04.1 LTS`

- Matrix products: default

- BLAS: `/usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3`

- LAPACK: `/usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.26.so` ; LAPACK version3.12.0

- Base packages: base, datasets, grDevices, graphics, methods, stats, utils

- Other packages: Biobase 2.65.1, BiocGenerics 0.51.1, geneRecommender 1.77.0

- Loaded via a namespace (and not attached): buildtools 1.0.0, compiler 4.4.1, knitr 1.48, maketools 1.3.0, sys 3.4.2, tools 4.4.1, xfun 0.47

References

[1] Art B. Owen, Josh Stuart, Kathy Mach, Anne M. Villeneuve, and Stuart Kim. "A Gene Recommender Algorithm to Identify Coexpressed Genes in C. elegans." *Genome Research* 13:1828-1837, 2003.