Getting started with *goTools* package

Agnes Paquet[1] and (Jean) Yee Hwa Yang[2]

April 15, 2025

1. `paquetagnes@yahoo.com`
2. Department of Medicine, University of California, San Francisco,
`http://www.biostat.ucsf.edu/jean`

Contents

1 Getting started

This document provides a tutorial for the *goTools* package, which allows graphical comparisons of functional groups between two sets of genes.

Installing the package: To install the *goTools* package, go to the Bioconductor installation web site `http://www.bioconductor.org/help/faq` for more detailed instructions.

Help files: As with any R package, detailed information on functions, classes and methods can be obtained in the help files. For instance, to view the help file for the function `ontoCompare` in a browser, use `help.start()` followed by `?ontoCompare`.
We demonstrate the functionality with a randomly selected set of probe IDs from both Affymetrix hgu133a chip (`affylist`). To load the `probeID` dataset, use `data(probeID)`, and to view a description of the experiments and data, type `?probeID`.

Sweave: This document was generated using the `Sweave` function from the R *tools* package. The source file is in the `/inst.doc` directory of the package *goTools*.

To begin, let's load the package and the probeID datasets into your R session.

```
> library("goTools", verbose=FALSE)
> data(probeID)
```

As shown below, `affylist` is a vector of lists containing 3 list of vectors of probe ids from Affymetrix hgu133a chip.

```
> class(affylist)

[1] "list"

> length(affylist)

[1] 3

> affylist[[1]][1:5]

[1] "215828_at"   "201849_at"   "219719_at"   "213690_s_at" "203172_at"
```

2 Graphical comparisons of two sets of genes

Gene Ontology is a Direct Acyclic Graph (DAG) that provides three structured networks of defined terms to describe gene product attributes: Molecular Function (MF), Biological Process (BP) and Cellular Component (CC). A gene product has one or more molecular functions and is used in one or more biological processes; it might be associated with one or more cellular components. To learn more about GO and DAG, please refer to Gene Ontology web site `http://www.geneontology.org/`.

We have created a set of R functions that use GO structure to describe and compare the composition of sets of genes (or probes). We use the following algorithm:

1. Read in a **list** of sets of probe id you want to compare.
2. Map each probe id to corresponding ontologies in the GO tree, if any.
3. Create the set of GO ids of interest used to compare your datasets (`endnode`). The function `EndNodeList()` will create a set of nodes of the DAG located one level under MF, BP or CC, but you can use any sets of GO ids.
4. For each GO id, go up the GO tree until reaching the nodes in `endnode`. Search may be limited to MF, BP or CC if specified in `goType`.
5. Compute the percentage of direct children found under each node in `endnode`.
6. Return the results. Plot them if `plot=TRUE`.

2.1 How to use goTools

The main function that we provide is `ontoCompare`. It takes as argument **a list** of probe ids. Their type must be specified in the argument `probeType`. For more details about it,you can refer to the corresponding help file by typing: `?ontoCompare`.

```
> library(GO.db)
> subset=c(L1=list(affylist[[1]][1:5]),L2=list(affylist[[2]][1:5]))
> res <-ontoCompare(subset, probeType="hgu133a")
```

2.1.1 Methods for computing percentages

`ontoCompare` allows you to choose from 3 different methods to estimate the percentage of probes under each element of `endnode`. The default method is `TGenes`.

1. `TGenes`: for each end node, return the number of direct children found / total number of probe ids.
   This includes oligos which do not have GO annotations.

2. `TIDS`: for each end node, return the number of direct children found / total number of GO ids describing the list.

3. `none`: for each end node, return the number of direct children found.

2.2 Plotting the results

The plots are produced using the function `ontoPlot`. It is called by `ontoCompare` when you set `plot=TRUE`. You can also call it directly, passing as argument `ontoCompare` results. If only one set of genes is passed to `ontoCompare`, `ontoPlot` will return a pie chart. In other cases, it will return a bargraph. You can modify `ontoPlot` layout parameters using usual R graphics layout parameters. For more details, type `?par`.

```
> library(GO.db)
> subset=c(L1=list(affylist[[1]][1:5]),L2=list(affylist[[2]][1:5]))
> res <- ontoCompare(subset, probeType="hgu133a", plot=TRUE)
```

3 How to set up the "end nodes"

3.1 Default list

The default end nodes list is defined by a call to the function `EndNodeList`. It contains all children of MF(GO:0003674), BP(GO:0008150) and CC (GO:0005575).

```
> EndNodeList()

        "GO:0003674"        "GO:0005575"        "GO:0008150"
                 isa                 isa                 isa
        "GO:0003774"        "GO:0003824"        "GO:0005198"
```

```
                   isa                    isa                    isa
           "GO:0005215"           "GO:0005488"           "GO:0009055"
                   isa                    isa                    isa
           "GO:0016209"           "GO:0038024"           "GO:0044183"
                   isa                    isa                    isa
           "GO:0045182"           "GO:0045735"           "GO:0060089"
                   isa                    isa                    isa
           "GO:0060090"           "GO:0090729"           "GO:0098772"
                   isa                    isa                    isa
           "GO:0102910"           "GO:0140104"           "GO:0140110"
                   isa                    isa                    isa
           "GO:0140223"           "GO:0140313"           "GO:0140489"
                   isa                    isa                    isa
           "GO:0140522"           "GO:0140657"           "GO:0140691"
                   isa                    isa                    isa
           "GO:0140776"           "GO:0140777"           "GO:0140911"
                   isa                    isa                    isa
           "GO:0140912"           "GO:0141047"           "GO:0180020"
                   isa                    isa                    isa
           "GO:0180024"           "GO:0180051"           "GO:0032991"
                   isa                    isa                    isa
           "GO:0044423"           "GO:0110165"           "GO:0002376"
                   isa                    isa                    isa
           "GO:0009987"           "GO:0016032"           "GO:0022414"
                   isa                    isa                    isa
           "GO:0032501"           "GO:0032502"           "GO:0040007"
                   isa                    isa                    isa
           "GO:0040011"           "GO:0042592"           "GO:0043473"
                   isa                    isa                    isa
           "GO:0044419"           "GO:0044848"           "GO:0048511"
positively regulates negatively regulates              regulates
           "GO:0048518"           "GO:0048519"           "GO:0050789"
                   isa                    isa                    isa
           "GO:0050896"           "GO:0051179"           "GO:0051703"
                   isa                    isa
           "GO:0065007"           "GO:0098754"
```

3.2 Customized end node list

If you want to use more ontologies to describe your set of genes, you can use the function `CustomEndNodeList(id,rank)` to create a bigger set of end nodes. It returns all GO ids children of `id` up to `rank` levels below `id`.

```
> MFendnode <- CustomEndNodeList("GO:0003674", rank=2)
```

Finally, the code below shows you how to use a custom end node list, and also how to modify the `goType` argument to select only Molecular Function (MF) ontologies.

```
> res <- ontoCompare(subset, probeType="hgu133a", endnode=MFendnode, goType="MF")
```

You can also create a list of GO ids of nodes of interest and pass it directly to the `endnode` argument in `ontoCompare`. GO ids must be in the following format: "GO:*XXXXXXX*."