How to perform module-based differential correlation analysis and GO enrichment using DGCA

Andrew McKenzie, Bin Zhang

Oct 24, 2016

Contents

1 Introduction

This vignette shows how to perform module-based differential correlation analysis, module-based GO enrichment, and differential correlation module detection from DGCA results through integration with MEGENA. To learn how to perform the differential correlation pipeline step-by-step, as well as explore some of the different options available in DGCA, please see the extended vignette.

2 Data loading and module construction

First, we will load the package and read in example data from single-cell RNA-sequencing data from neurons and oligodendrocytes, generated in Darmanis *et al.* (Darmanis 2015), cleaned for this analysis, and put in the data folder.

```
library(DGCA, quietly = TRUE)
data(darmanis)
data(design_mat)
```

We will now construct toy modules for this data set. However, these modules could also be constructed by a complementary module-detection method, such as WGCNA or MEGENA.

```
module_genes = list(
  mod1 = rownames(darmanis)[1:100],
  mod2 = rownames(darmanis)[90:190],
  mod3 = rownames(darmanis)[190:290],
  mod4 = rownames(darmanis)[330:340],
  mod5 = rownames(darmanis)[350:360],
  mod6 = rownames(darmanis)[400:405])
modules = stack(module_genes)
modules$ind = as.character(modules$ind)
str(modules)
```

```
## 'data.frame':    330 obs. of  2 variables:
##  $ values: chr  "AASDHPPT" "ACSL3" "ACTB" "ACTG1" ...
##  $ ind   : chr  "mod1" "mod1" "mod1" "mod1" ...
```

```
head(modules)
```

```
##     values  ind
## 1 AASDHPPT mod1
## 2    ACSL3 mod1
## 3     ACTB mod1
## 4    ACTG1 mod1
## 5     ADD1 mod1
## 6     ADD3 mod1
```

Note that the genes in these modules are partially overlapping; this is allowed in the subsequent analyses, although not required.

3 Module-based differential correlation

Having an expression matrix, a design matrix, and a set of modules allows us to perform module-based differential correlation analysis. This analysis finds the average (median or mean) change in correlation between gene symbols in the two conditions, the significance of that change in correlation, as well as the top genes with a gain and/or loss in correlation with the other genes in the module between the conditions, if any of them are significant.

```
moduleDC_res = moduleDC(inputMat = darmanis, design = design_mat,
                        compare = c("oligodendrocyte", "neuron"), genes = modules$values,
                        labels = modules$ind, nPerm = 50, number_DC_genes = 3,
                        dCorAvgMethod = "median")
head(moduleDC_res)
```

```
##   Module Size      MeDC pVal                    Top_GOC Top_LOC
## 1   mod1  100 0.5697351 0.00      ATP5B, ATP6V1A, CADM1
## 2   mod2  101 0.4495679 0.02         GAPDH, DDX24, CYCS
## 3   mod3  101 0.4234082 0.00 HSP90AB1, HMGB1, LOC647979
## 4   mod4   11 0.5013138 0.08
## 5   mod5   11 0.6319075 0.10
## 6   mod6    6 1.1471509 0.20
```

It is also possible to take one module and measure differential correlation strength for each of its genes compared to all of the others in the module:

```
mod1_genes = modules[modules$ind == "mod1", "values"]
darmanis_mod1 = darmanis[mod1_genes, ]
moduleDC_res = ddcorAll(inputMat = darmanis_mod1, design = design_mat,
                        compare = c("oligodendrocyte", "neuron"), nPerm = 50,
                        getDCorAvg = TRUE, dCorAvgType = "gene_average",
                        dCorAvgMethod = "median")
head(moduleDC_res[["avg_dcor"]])
```

```
##       Gene  avgZDiff empirical_pVal   pVal_adj
## 44   ATP5B 1.4499040           0.00 0.00000000
## 52 ATP6V1A 1.3890802           0.00 0.00000000
## 74   CADM1 1.1907264           0.00 0.00000000
## 22   APLP2 1.0913738           0.00 0.00000000
## 11   ALDOA 1.0367740           0.00 0.00000000
## 77   CALM2 0.9736599           0.02 0.04545455
```

```
tail(moduleDC_res[["avg_dcor"]])
```

```
##       Gene     avgZDiff empirical_pVal  pVal_adj
## 15 ANKRD10  0.146732102           0.74 0.7789474
## 86 CCDC88A -0.135183780           0.58 0.6373626
## 96  CEP290 -0.100703128           0.76 0.7916667
## 7     AFF4  0.086361428           0.80 0.8247423
## 26  ARGLU1  0.072837635           0.96 0.9600000
## 85  CCDC82  0.006523057           0.96 0.9600000
```

4 Module-based gene ontology (GO) enrichment

This function returns a list for each module, containing a list of data frames for the enrichment of all of the terms in each of the GO categories chosen (each of BP, MF, and CC for the default of "all"). Notably, if you want to extract all of the categories regardless of the p-values, which is helpful for downstream applications that compare across groups, you should set pval_GO_cutoff = 1 (the default value). If you only want to extract the GO terms with significant p-values (unadjusted), you can make this number lower.

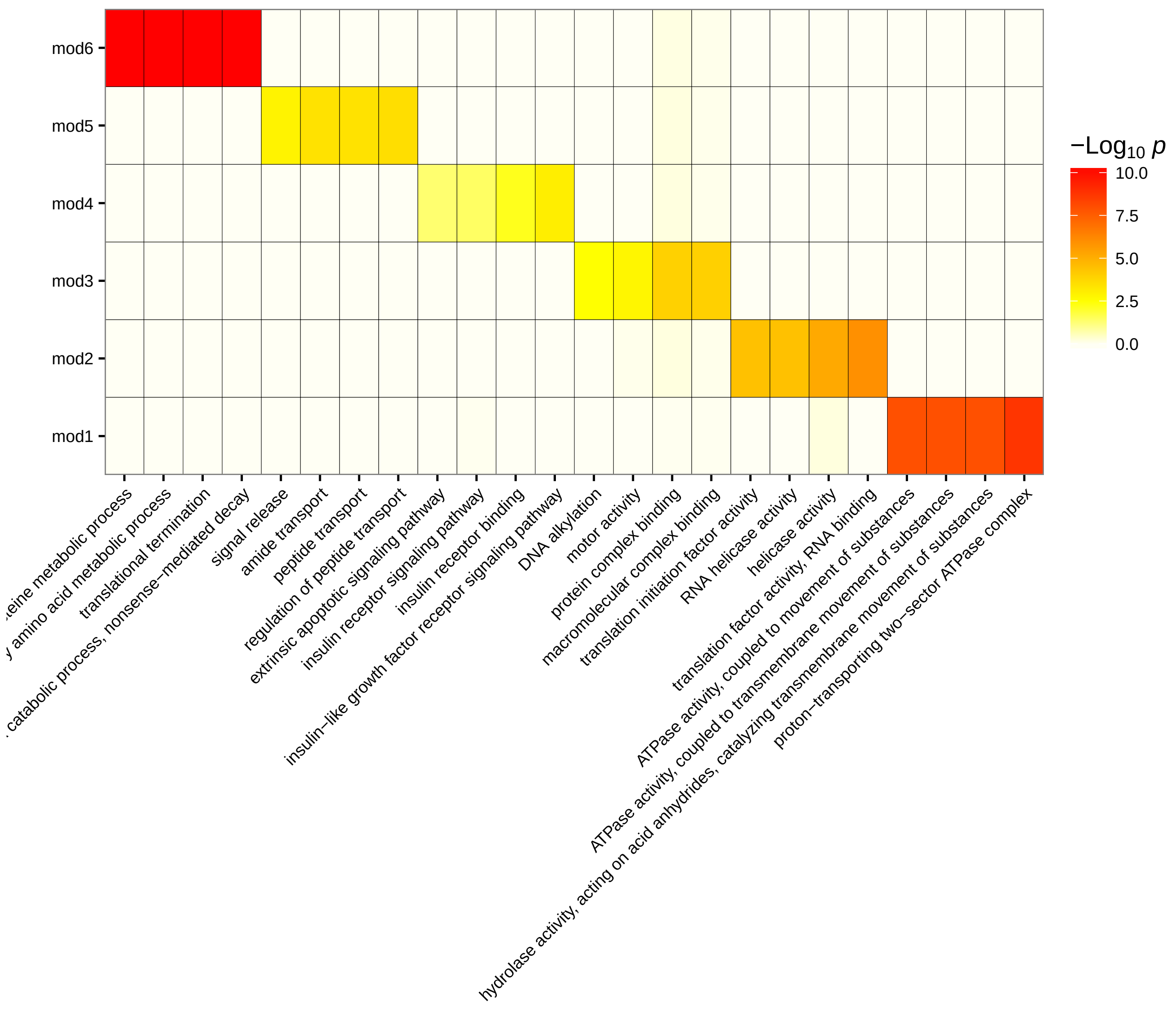
```
library(GOstats, quietly = TRUE)
library(HGNChelper, quietly = TRUE)
library(org.Hs.eg.db, quietly = TRUE)
moduleGO_res = moduleGO(genes = modules$values, labels = modules$ind,
                        universe = rownames(darmanis), pval_GO_cutoff = 1)
```

In order to extract information from this, DGCA contains a function to convert this result into a data frame:

```
moduleGO_df = extractModuleGO(moduleGO_res)
```

Next, this data frame can be inputted into a heatmap plotting function in order to visualize the top GO term enrichments in each group groups. Note that this requires the ggplot2 R package.

```
library(ggplot2, quietly = TRUE)
plotModuleGO(moduleGO_df, nTerms = 4, text_size = 8, coord_flip = TRUE)
```



5 Differential correlation module detection through integration with MEGENA

Given results from a DGCA differential correlation analysis, it is also possible to identify modules using MEGENA. For demonstration purposes, we identified

```r
library(MEGENA, quietly = TRUE)
ddcor_res = ddcorAll(inputMat = darmanis, design = design_mat,
    compare = c("oligodendrocyte", "neuron"),
    adjust = "none", heatmapPlot = FALSE, nPerm = 0, nPairs = "all")
str(ddcor_res)
```

```
## 'data.frame':    163306 obs. of  9 variables:
##  $ Gene1               : chr  "CACYBP" "CACYBP" "NDUFB9" "CACYBP" ...
##  $ Gene2               : chr  "NACA" "SSB" "SSB" "NDUFB9" ...
##  $ oligodendrocyte_cor : num  -0.07026 -0.05529 -0.00967 0.33362 -0.10442 ...
##  $ oligodendrocyte_pVal: num  0.6751 0.7416 0.9541 0.0407 0.5327 ...
##  $ neuron_cor          : num  0.957 0.958 0.949 0.971 0.925 ...
##  $ neuron_pVal         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ zScoreDiff          : num  10.26 10.25 9.52 9.15 8.95 ...
##  $ pValDiff            : num  1.10e-24 1.16e-24 1.81e-21 5.92e-20 3.50e-19 ...
##  $ Classes             : Factor w/ 10 levels "NonSig","+/+",..: 5 5 5 2 5 5 5 5 5 5 ...
```

Using the DGCA results without adjustments for multiple hypothesis tests (adjusted = FALSE), DGCA offers a convenience function for integrating with MEGENA. This function will extract the gene pairs less than a certain p-value, construct a prefuse force network using those gene pairs, detect modules in that prefuse force network, and calculate hub genes within those modules. By default, the MEGENA integration function evaluates its detected modules by their compactness. Note also that if you do not want MEGENA to report its computations, then you need to use suppressMessages() around it.

```
megena_res = ddMEGENA(ddcor_res, adjusted = FALSE, evalCompactness = TRUE)
```

```
str(megena_res$modules)
```

```
## 'data.frame':    1233 obs. of  2 variables:
##  $ Genes  : chr  "CACYBP" "NDUFB9" "NACA" "COX6A1" ...
##  $ Modules: Factor w/ 23 levels "c1_10","c1_11",..: 11 11 11 11 11 11 11 11 11 11 ...
```

```
head(megena_res$modules)
```

```
##    Genes Modules
## 1 CACYBP    c1_2
## 2 NDUFB9    c1_2
## 3   NACA    c1_2
## 4 COX6A1    c1_2
## 5    NCL    c1_2
## 6    SSB    c1_2
```

However, in many cases (such as when average module size is small), using the compactness evaluation may lead to problems in MEGENA. If so, it may be advisable to turn off the module compactness evaluation step.

```
megena_res = ddMEGENA(ddcor_res, adjusted = FALSE, evalCompactness = FALSE)
```

```
str(megena_res$modules)
```

```
## 'data.frame':    1828 obs. of  2 variables:
##  $ Genes  : chr  "CACYBP" "NDUFB9" "NACA" "COX6A1" ...
##  $ Modules: chr  "c1_1" "c1_1" "c1_1" "c1_1" ...
```

```
head(megena_res$modules)
```

```
##     Genes Modules
## 1  CACYBP    c1_1
## 2  NDUFB9    c1_1
## 3    NACA    c1_1
## 4  COX6A1    c1_1
## 5 ATP6V1D    c1_1
## 6     NCL    c1_1
```

For more options in integrating DGCA with MEGENA, please see help(MEGENA) and browse its associated vignette.

References

Darmanis. 2015. "A Survey of Human Brain Transcriptome Diversity at the Single Cell Level." *PNAS* 112 (23): 7285–90. doi:10.1073/pnas.1507125112.