Filtering

Mikhail Dozmorov

2016-10-03

genefilter & collapseRows - removing genes with certain properties before DE analysis

Filtering any unimportant genes before differential expression (DE) analysis is quite desirable because it increases the statistical power of your differential expression analysis. In other words, a smaller gene set suffers less of a hit from multiple testing correction.

Another potential use for filtering is reducing the feature space for prediction analysis. This will cause models to be trained faster and probably to have less variance.

Of course, the tradeoff in both cases is that the genes you filter may be false negatives: that they may be truly differentially expressed or may carry unique predictive power. Filtering also reduces variability needed for proper functionality of the `limma` package https://www.ncbi.nlm.nih.gov/pubmed/20460310.

```
library(ALL)
library(genefilter)
data(ALL)
```

Looks like log2-transformed, not normalized

```
summary(exprs(ALL)[, 1:3])
```

```
##      01005            01010            03002
##  Min.   : 2.435   Min.   : 2.423   Min.   : 2.271
##  1st Qu.: 4.111   1st Qu.: 4.139   1st Qu.: 4.118
##  Median : 5.455   Median : 5.532   Median : 5.479
##  Mean   : 5.630   Mean   : 5.648   Mean   : 5.633
##  3rd Qu.: 6.826   3rd Qu.: 6.867   3rd Qu.: 6.860
##  Max.   :13.455   Max.   :13.674   Max.   :13.796
```

`kOverA` filters out probes which do not have at least "k" experiments in which they are over expression level "A". The idea here is that the variability of low-expressed genes is harder to distinguish from the baseline noise level.

```
ka.filter <- filterfun(kOverA(5, 4))
ix.ka <- genefilter(exprs(ALL), ka.filter)
nrow(exprs(ALL))
```

```
## [1] 12625
```

```
sum(ix.ka)
```

```
## [1] 11294
```

you could now alter the original ALL structure by

```
exprs(ALL) <- exprs(ALL)[ix.ka, ]
```

`nsFilter` stands for "non-specific filter". Its main purpose is to remove probes which do not map to genes or probes which do not have GO categories.

```
filtered <- nsFilter(ALL, require.entrez = T)
filtered$filter.log
```

```
## $numDupsRemoved
## [1] 2370
##
## $numLowVar
## [1] 3949
##
## $numRemoved.ENTREZID
## [1] 1008
##
## $feature.exclude
## [1] 19
```

```
filtered$eset  # The new ExpressionSet
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 3948 features, 128 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: 01005 01010 ... LAL4 (128 total)
##   varLabels: cod diagnosis ... date last seen (21 total)
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
##   pubMedIds: 14684422 16243790
## Annotation: hgu95av2
```

Lots of other filters exist. For example, you can filter by Anova or T-tests with "Anova" or "colttests" filters. See some more options:

```
ls("package:genefilter")
```

```
##  [1] "Anova"          "area"           "AUC"            "colFtests"      "colttests"      "coxf
## [13] "filtered_p"     "filtered_R"     "filterfun"      "findLargest"    "gapFilter"      "gene
## [25] "isESet"         "kappa_p"        "kappa_t"        "kOverA"         "maxA"           "nsFi
## [37] "rowFtests"      "rowpAUCs"       "rowSds"         "rowttests"      "rowVars"        "sens
## [49] "ttest"          "varFilter"
```

`collapseRows` from the WGCNA package serves a similar purpose. It takes an expression matrix, a vector of probe IDs, and a vector of corresponding category IDs (usually Entrez Gene or Ensembl IDs) and collapses all the probes mapping to a given gene into one row.

It has many possible methods for collapsing the rows (in fact, an entire paper was written just about this one function. By default, rows are collapsed by max mean.

```
library(WGCNA)
library(hgu95av2.db)

M <- exprs(ALL)
probes <- rownames(M)
xx <- as.list(hgu95av2SYMBOL)
symbols <- unlist(xx[probes])

clr <- collapseRows(M, symbols, probes)
```

```
## Warning in collapseRows(M, symbols, probes): The argument rowGroup contains missing data. It is stro
##     that you remove these rows before calling the function. Or redefine rowGroup
##     so that it has no missing data. But for convenience, we remove these data.
```

```
names(clr)
```

```
## [1] "datETcollapsed" "group2row"      "selectedRow"
```

```
M.genes <- clr$datETcollapsed
```