

Primer: Preparing NChannelSet objects with differential expression scores

May 2, 2019

This document exemplifies the processing of differential expression data using small, simulated datasets shipped with the `gCMAP` package. To see real-life examples with data from available from public databases, please refer to the documentation of the `gCMAPWeb` companion package.

1 Differential expression analysis

The `gCMAP` package offers a the `generate_gCMAP_NChannelSet` function to process multiple instances differential expression experiments with two classes (e.g. cases vs controls). For microarray data, the `limma` package is used to calculate a moderated t-statistic (default). Optionally, a standard t-test can be computed instead. For RNAseq data, the `DESeq` package is used instead.

Data preprocessing differs considerably between different technologies and array platforms and needs to be performed beforehand. Normalized microarray data and accompanying annotation is passed to `generate_gCMAP_NChannelSet` as a list of `ExpressionSet` objects, RNAseq data can be passed as a list of `CountDataSet` objects instead.

To generate a set of 3 example `CountDataSets` , we use the `makeExampleCountDataSet` function from the `DESeq` package.

```
> library(gCMAP)
> library(DESeq)
> set.seed( 123 )
> cds.list <- lapply( 1:3, function(n) {
+   cds <- makeExampleCountDataSet()
+   featureNames(cds) <- paste("gene",1:10000, sep="_")
+   cds
+ })
> names(cds.list) <- paste("Instance", 1:3, sep="")
> sapply(cds.list, dim)
```

```
      Instance1 Instance2 Instance3
Features    10000    10000    10000
Samples         5         5         5
```

```
> sapply(cds.list, function(n) pData(n)$condition )
```

```
      Instance1 Instance2 Instance3
[1,] "A"        "A"        "A"
[2,] "A"        "A"        "A"
[3,] "B"        "B"        "B"
[4,] "B"        "B"        "B"
[5,] "B"        "B"        "B"
```

By default, each `CountDataSet` object contains counts for 10000 genes from five samples. Each sample is assigned to one of two conditions, A or B, in the `phenoData` slot of the `CountDataSet`. The `pData` column containing group membership information (e.g. "condition") is provided as the `control_perturb_col` parameter. The levels associated with control and treatment groups are specified as "control" and "perturb" character strings.

Each of the three `CountDataSet` instances is analyzed individually by `generate_gCMAP_NChannelSet`. To assemble the results into a single `NChannelSet`, the input `ExpressionSet` or `CountDataSet` objects must contain measurements for the same features (e.g. the vectors returned by "featureNames" must be identical across all instances).

To include information about the instances in the `NChannelSet`, a 'sample.annotation' data.frame can be provided, containing exactly one row for each element of the input list of `ExpressionSet` / `CountDataSet` objects.

```
> ## this step takes a little time
> cde <- generate_gCMAP_NChannelSet(cds.list,
+                               uids=1:3,
+                               sample.annotation=NULL,
+                               platform.annotation="Entrez",
+                               control_perturb_col="condition",
+                               control="A",
+                               perturb="B")
> channelNames(cde)

[1] "exprs" "log_fc" "mod_fc" "p" "z"
```

For array data, a `NChannelSet` with slots "exprs", "z", "p", and "log_fc" is returned, containing the average intensity across all samples within the instance, z-scores, (raw) p-values and log2 fold changes, respectively. If count data is processed, an additional "mod_fc" channel is returned, providing the moderated fold change, calculated after performing variance-stabilising transformation across all input instances. (Please consult the DESeq vignette for details.)

1.1 Storing assayData as BigMatrix objects on disk

When large numbers of instances are processed, the resulting `NChannelSet` objects can require large amounts of memory. The `bigmemory` and `bigmemoryExtras` packages can be used to create `BigMatrix` objects, allowing methods to subset large datasets without having to load them fully into memory first.

Note: at the time of writing, the `bigmemory` package was only available for Unix and Mac OS X operating systems but not for Windows. Windows users can take advantage of `gCMAP`'s functionality but datasets must be fully loaded into memory first.

If the `bigmemory` and `bigmemoryExtras` packages are available and a file name is provided via the "big.matrix" parameter, `generate_gCMAP_NChannelSet` uses the `BigMatrix` package to store data from each channel on disk. In the future, individual channels and / or subsets of the datasets can then be loaded without requiring the full object to be read into memory again.

To highlight this functionality, we derive three (arbitrary) instances from the `sample.ExpressionSet` object available from the `Biobase` package, process them and store the results in a temporary directory. Note: this section will only create the expected `big.matrix` files on disk if the `bigmemory` and `bigmemoryExtras` packages can be loaded.

```
> ## list of ExpressionSets
> if (require(bigmemory)) {
+   data("sample.ExpressionSet") ## from Biobase
+ }
```

```

+ es.list <- list( sample.ExpressionSet[,1:4],
+                 sample.ExpressionSet[,5:8],
+                 sample.ExpressionSet[,9:12])
+
+ ## three instances
+ names(es.list) <- paste( "Instance", 1:3, sep=".")
+
+ storage.file <- tempfile()
+ storage.file ## filename prefix for BigMatrices
+
+ de <- generate_gCMAP_NChannelSet(
+   es.list,
+   1:3,
+   platform.annotation = annotation(es.list[[1]]),
+   control_perturb_col="type",
+   control="Control",
+   perturb="Case",
+   big.matrix=storage.file)
+
+ channelNames(de)
+ head( assayDataElement(de, "z") )
+
+ dir(dirname( storage.file ))
+ }

[1] "file7f2e7a6f8452.rdata"      "file7f2e7a6f8452_exprs"
[3] "file7f2e7a6f8452_exprs.desc" "file7f2e7a6f8452_log_fc"
[5] "file7f2e7a6f8452_log_fc.desc" "file7f2e7a6f8452_p"
[7] "file7f2e7a6f8452_p.desc"     "file7f2e7a6f8452_z"
[9] "file7f2e7a6f8452_z.desc"

```

If the `bigmemoryExtras` package is available, it generated a `BigMatrix` objects containing pointers to three files in the temporary directory, one for each channel (identified by their suffices). If the package is unavailable, a standard `eSet` is saved to disk, which will be read fully into memory upon reload.

To demonstrate the use of disk-based `NChannelSet` objects, we will first delete the object from the current R workspace and reload it from disk.

Accessing the complete matrix in the `assayData` slots, e.g. for the "z" channel, returns another `BigMatrix` object with `assayData` slot pointing to the associated file on disk. Upon subsetting, only the requested part of the dataset is loaded into memory.

```

> if (require(bigmemory)) {
+   ## remove de object from R session and reload
+   rm( de )
+
+   de <-get( load( paste( storage.file, "rdata", sep=".") ) )
+   class( assayDataElement(de, "z") )
+   assayDataElement(de, "z")[1:10,] ## load subset
+ }

```

	1	2	3
AFFX-MurIL2_at	-1.36808562	0.04333555	-0.7255849
AFFX-MurIL10_at	1.56254427	-0.69203457	0.1589525

```

AFFX-MurIL4_at -0.65915229 -0.85080055 0.1804448
AFFX-MurFAS_at -0.31745996 0.43936805 0.2813885
AFFX-BioB-5_at -0.08767134 0.15619365 -0.2836740
AFFX-BioB-M_at -0.32253278 0.82819990 -0.5521458
AFFX-BioB-3_at -0.30488232 1.79473755 0.4374636
AFFX-BioC-5_at -0.29368831 0.34488031 0.0982909
AFFX-BioC-3_at 0.05507180 -1.89130218 0.2943413
AFFX-BioDn-5_at 0.78669240 0.74946863 1.0688364

```

The `memorize` function reads the complete `NChannelSet` into memory. In addition, one or more selected channels can be specified with the `'name'` parameter.

```

> if (require(bigmemory)) {
+   ## read z-score channel into memory
+   dem <- memorize( de, name="z" )
+   channelNames(dem)
+
+   class( assayDataElement(dem, "z") ) ## matrix
+   ## remove tempfiles
+   unlink(paste(storage.file,"*", sep=""))
+ }

```

```
> sessionInfo()
```

```

R version 3.6.0 (2019-04-26)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 18.04.2 LTS

```

```

Matrix products: default
BLAS: /home/biocbuild/bbs-3.9-bioc/R/lib/libRblas.so
LAPACK: /home/biocbuild/bbs-3.9-bioc/R/lib/libRlapack.so

```

```

locale:
 [1] LC_CTYPE=en_US.UTF-8      LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8      LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8  LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8     LC_NAME=C
 [9] LC_ADDRESS=C             LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

```

```

attached base packages:
 [1] stats4    parallel  stats     graphics  grDevices  utils      datasets
 [8] methods  base

```

```

other attached packages:
 [1] bigmemory_4.5.33    DESeq_1.36.0        lattice_0.20-38
 [4] locfit_1.5-9.1      gCMAP_1.28.0        limma_3.40.0
 [7] GSEABase_1.46.0     graph_1.62.0        annotate_1.62.0
[10] XML_3.98-1.19       AnnotationDbi_1.46.0 IRanges_2.18.0
[13] S4Vectors_0.22.0    Biobase_2.44.0      BiocGenerics_0.30.0

```

```
loaded via a namespace (and not attached):
```

[1]	Rcpp_1.0.1	compiler_3.6.0	RColorBrewer_1.1-2
[4]	bigmemoryExtras_1.32.0	bitops_1.0-6	tools_3.6.0
[7]	digest_0.6.18	bit_1.1-14	RSQLite_2.1.1
[10]	memoise_1.1.0	Matrix_1.2-17	DBI_1.0.0
[13]	Category_2.50.0	GSEAlm_1.44.0	genefilter_1.66.0
[16]	bit64_0.9-7	grid_3.6.0	bigmemory.sri_0.1.3
[19]	RBGL_1.60.0	survival_2.44-1.1	genefilter_1.62.0
[22]	blob_1.1.1	splines_3.6.0	xtable_1.8-4
[25]	RCurl_1.95-4.12		