

# Primer: Preparing NChannelSet objects with differential expression scores

April 16, 2015

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. Otherwise, a standard Rdata object is created and a warning is issued.

```
> ## list of ExpressionSets
> 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

[1] "/tmp/RtmpnGybmE/file4c3d78426904"

> 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)

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

> head( assayDataElement(de, "z") )

              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

> dir(dirname( storage.file ))

[1] "file4c3d78426904.rdata"

```

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.

```

> ## remove de object from R session and reload
> rm( de )
> de <-get( load( paste( storage.file, "rdata", sep=".") ) )
> class( assayDataElement(de, "z") )

[1] "matrix"

> 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.

```
> ## read z-score channel into memory
> dem <- memorize( de, name="z" )
> channelNames(dem)
```

```
[1] "z"
```

```
> class( assayDataElement(dem, "z") ) ## matrix
```

```
[1] "matrix"
```

```
> sessionInfo()
```

```
R version 3.2.0 (2015-04-16)
Platform: x86_64-unknown-linux-gnu (64-bit)
Running under: Ubuntu 14.04.2 LTS
```

```
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] DESeq_1.20.0      lattice_0.20-31    locfit_1.5-9.1
[4] gCMAP_1.12.0     limma_3.24.0      GSEABase_1.30.0
[7] graph_1.46.0     annotate_1.46.0    XML_3.98-1.1
[10] AnnotationDbi_1.30.0 GenomeInfoDb_1.4.0 IRanges_2.2.0
[13] S4Vectors_0.6.0   Biobase_2.28.0    BiocGenerics_0.14.0
```

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

```
[1] splines_3.2.0     GSEAlm_1.28.0     xtable_1.7-4      Category_2.34.0
[5] tools_3.2.0      grid_3.2.0        DBI_0.3.1         genefilter_1.50.0
```

[9] survival\_2.38-1 RBGL\_1.44.0 Matrix\_1.2-0 geneplotter\_1.46.0  
[13] RColorBrewer\_1.1-2 RSQLite\_1.0.0