How to Ruin Your Differential Expression Analysis in 2 Easy Steps

A study in factors screwing you up again

note: I am not associated with any of the DESeq2 team, I'm simply using their awesome tool

*note: The booby trap discussed in this write-up is not caused by DESeq2. It's caused by a notoriously finicky data-type in R that arose when corrupted data was **passed** to DESeq2*

I recently found myself dodging an analytical bullet after inheriting an old project that needed somebody to dust is off and wrap it up. The trap that lay in wait for me is simple enough to (accidentally) create and easy enough to step in that I thought I would write up a quick post-mortem case study.

First, let's run a simple example analysis using the awesome DESeq2 package to get our baseline. Aside from some personal-preference level formatting adjustments, this is a cut-and-paste example from the very thorough DESeq2 vignette

```
library(DESeq2)
```

```
## Loading required package: S4Vectors
```

```
## Loading required package: stats4
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which, which.max, which.min
```

```
##
## Attaching package: 'S4Vectors'
```

```
## The following object is masked from 'package:base':
##
##     expand.grid
```

```
## Loading required package: IRanges
## Loading required package: GenomicRanges
## Loading required package: GenomeInfoDb
## Loading required package: SummarizedExperiment
## Loading required package: Biobase
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
## Loading required package: DelayedArray
## Loading required package: matrixStats
##
## Attaching package: 'matrixStats'
## The following objects are masked from 'package:Biobase':
##
##     anyMissing, rowMedians
## Loading required package: BiocParallel
##
## Attaching package: 'DelayedArray'
## The following objects are masked from 'package:matrixStats':
##
##     colMaxs, colMins, colRanges, rowMaxs, rowMins, rowRanges
## The following objects are masked from 'package:base':
##
##     aperm, apply, rowsum
```

```
library(pasilla)

pasCts <- system.file("extdata", "pasilla_gene_counts.tsv",
                      package = "pasilla", mustWork = TRUE)

pasAnno <- system.file("extdata", "pasilla_sample_annotation.csv",
                       package = "pasilla", mustWork = TRUE)

cts <- as.matrix(read.csv(pasCts, sep = "\t", row.names = "gene_id"))

coldata <- read.csv(pasAnno, row.names = 1)

coldata <- coldata[, c("condition","type")]

rownames(coldata) <- sub("fb", "", rownames(coldata))

cts <- cts[, rownames(coldata)]

dds <- DESeqDataSetFromMatrix(countData = cts,
                              colData = coldata, design = ~ condition)
```

```r
dds$condition <- relevel(dds$condition, ref = "untreated")
```