DASC user guide (PDF)

Haidong Yi†, Ayush T. Raman†, Han Zhang, Genevera Allen, Zhandong Liu

30 October 2017

Abstract

Batch effects are one of the major source of technical variations in high throughput studies such as omics profiling. It has been well established that batch effects can be caused by different experimental platforms, laboratory conditions, different sources of samples and personnel differences. These differences can confound the outcomes of interest and lead to spurious results. A critical input for batch correction algorithms are the knowledge of batch factors, which in many cases are unknown or inaccurate. Hence, the primary motivation of our paper is to detect hidden batch factors that can be used in standard techniques to accurately capture the relationship between expression and other modeled variables of interest. Here, we present *DASC*, a novel algorithm that is based on convex clustering and semi-NMF for the detection of unknown batch effects.

Package

DASC 1.0.0

Contents

1 Getting started

DASC is an R package distributed as part of the Bioconductor project. To install the package, start R and enter:

```
source("http://bioconductor.org/biocLite.R")
biocLite("DASC")
```

2 Introduction

DASC is used for identifying batches and classifying samples into different batches in a high dimensional gene expression dataset. The batch information can be further used as a covariate in conjunction with other variables of interest among standard bioinformatics analysis like differential expression analysis.

2.1 Citation info

If you use *DASC* for your analysis, please cite it as here below. To cite package 'DASC' in publications use:

```
@Manual{,
    title = {DASC: Detecting hidden batch factors through data adaptive
        adjustment for biological effects.},
    author = {Haidong Yi, Ayush T. Raman, Han Zhang, Genevera I. Allen and
        Zhandong Liu},
    year = {2017},
    note = {R package version 0.1.0},
}
```

3 Quick Example

```
library(DASC)
data("esGolub")
samples <- c(20,21,28,30)
dat <- exprs(esGolub)[1:100,samples]
pdat <- pData(esGolub)[samples,]

## use nrun = 50 or more for better convergence of results
res <- DASC(edata = dat, pdata = pdat, factor = pdat$Cell, method = 'ama',
                type = 3, lambda = 1, rank = 2:3, nrun = 5,
                annotation="esGolub Dataset")
#consensusmap(res)
#plot(res)
```

4 Setting up the data

The first step in using DASC package is to properly format the data. For example, in case of gene expression data, it should be a matrix with features (genes, transcripts) in the rows and samples in the columns. DASC then requires the information for the variable of interest to model the gene expression data effectively.Variable of interest could be a genotype or treatment information.

4.1 Stanford RNA-Seq Dataset

Below is an example of Stanford gene expression dataset (Chen et. al. PNAS, 2015; Gilad et. al. F1000 Research, 2015). It is a filtered raw counts dataset which was published by Gilad et al. F1000 Research. 30% of genes with the lowest expression & mitochondrial genes were removed (Gilad et al.F1000 Research).

```
## libraries
set.seed(99999)
library(DESeq2)
library(ggplot2)
library(pcaExplorer)

## dataset
rawCounts <- stanfordData$rawCounts
metadata <- stanfordData$metadata
```

```
## Using a smaller dataset
idx <- which(metadata$tissue %in% c("adipose", "adrenal", "sigmoid"))
rawCounts <- rawCounts[,idx]
metadata <- metadata[idx,]
```

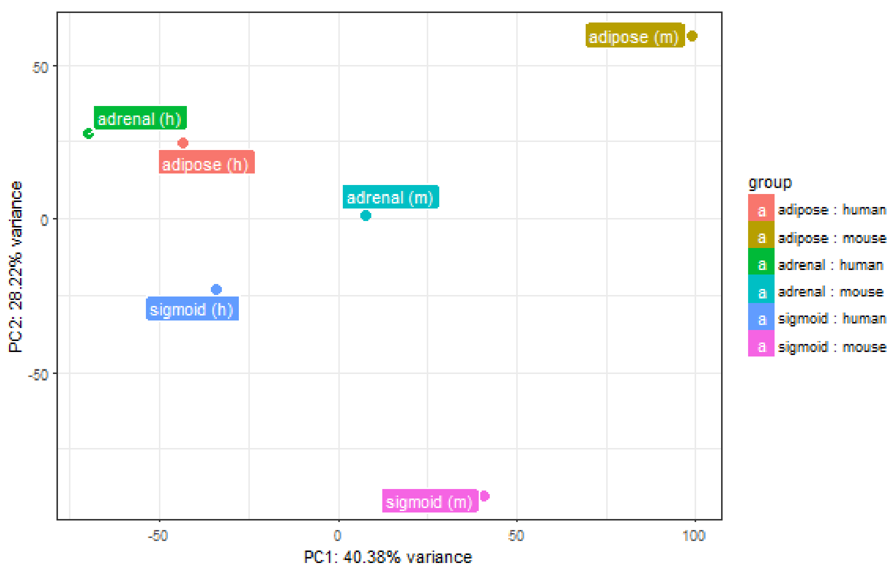
```
head(rawCounts)
##          adipose (h) adrenal (h) sigmoid (h) adipose (m) adrenal (m)
## STAG2           1430        4707        4392        3223        8235
## STAG1            835        2362        1687        2750        2732
## GOSR2            142         891          97        1599        1430
## C1orf43         1856        9591        2611         706         498
## ART5               1           4           0           0           0
## ART1               0           0           0           0           1
##          sigmoid (m)
## STAG2          10435
## STAG1           2833
## GOSR2            887
## C1orf43          753
## ART5               0
## ART1               0
head(metadata)
##                 setname                 seqBatch species  tissue
## adipose (h) adipose (h) D87PMJN1:253:D2GUAACXX:8   human adipose
## adrenal (h) adrenal (h) D87PMJN1:253:D2GUAACXX:8   human adrenal
```

```
## sigmoid (h) sigmoid (h) D87PMJN1:253:D2GUAACXX:8   human sigmoid
## adipose (m) adipose (m) D4LHBFN1:276:C2HKJACXX:4   mouse adipose
## adrenal (m) adrenal (m) D4LHBFN1:276:C2HKJACXX:4   mouse adrenal
## sigmoid (m) sigmoid (m) D4LHBFN1:276:C2HKJACXX:4   mouse sigmoid
```

5 Batch detection using PCA Analysis

```
## Normalizing the dataset using DESeq2
dds <- DESeqDataSetFromMatrix(rawCounts, metadata, design = ~ species+tissue)
dds <- estimateSizeFactors(dds)
dat <- counts(dds, normalized = TRUE)
lognormalizedCounts <- log2(dat + 1)
```

```
## PCA plot using
rld.dds <- rlog(dds)
pcaplot(rld.dds, intgroup=c("tissue","species"), ntop=1000, pcX=1, pcY=2)
```

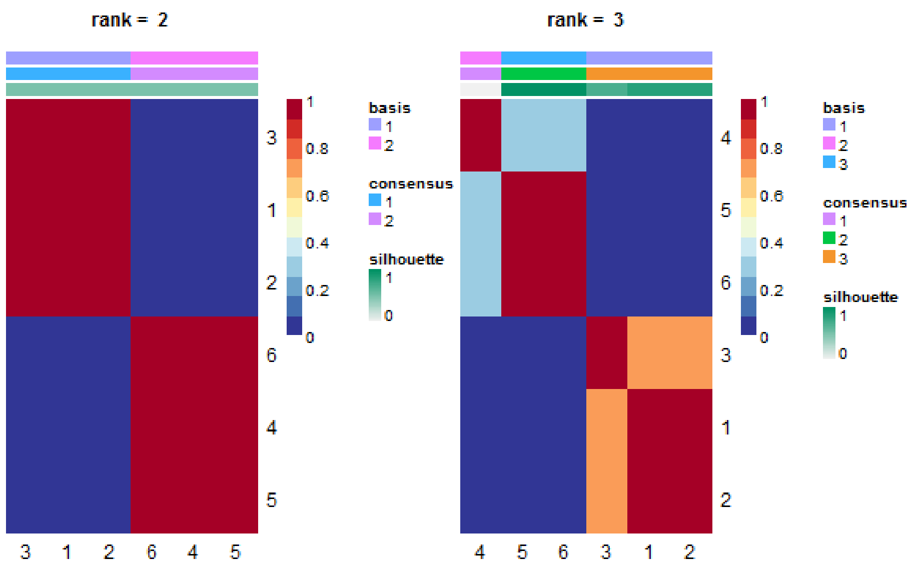


In the PCA plot, PC1 shows the differences between the species. PC2 shows the differences between the species i.e. samples clustering based on tissues.

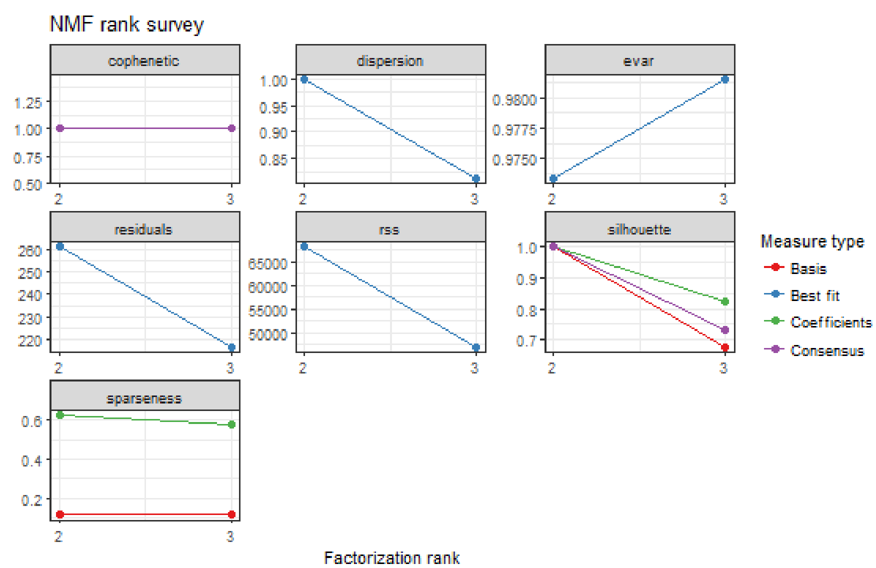
6 Batch detection using DASC

```
res <- DASC(edata = dat, pdata = metadata, factor = metadata$tissue,
                method = 'ama', type = 3, lambda = 1, rank = 2:3, nrun = 10,
                annotation = 'Stanford Dataset')
## Compute NMF rank= 2  ... + measures ... OK
## Compute NMF rank= 3  ... + measures ... OK
```

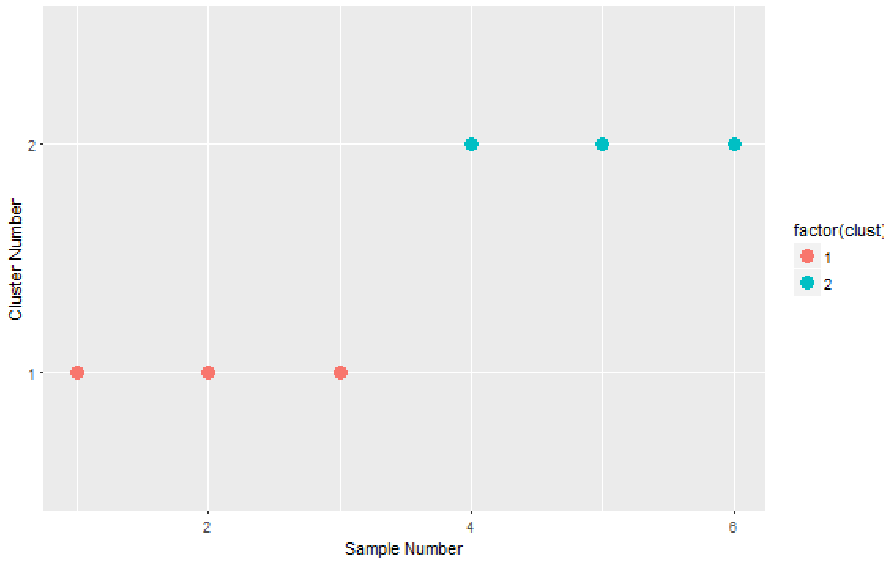
```r
## Consensus plot
consensusmap(res)
```




```r
## Residual plot
plot(res)
```




```r
## Batches -- dataset has 6 batches
sample.clust <- data.frame(sample.name = colnames(lognormalizedCounts),
                            clust = as.vector(predict(res$fit$`2`)),
                            batch = metadata$seqBatch)
ggplot(data = sample.clust, aes(x=c(1:6), y=clust, color=factor(clust))) +
    geom_point(size = 4) + xlab("Sample Number") + ylab("Cluster Number")
```



Based on the above plots, we observe that the dataset has 2 batches. This can further be compared with the sequencing platform or `metadata$seqBatch`. The results suggest that differences in platform led to batch effects. Batch number can be used as another covariate, when differential expression analyses using `DESeq2`,`edgeR` or `limma` are performed.

7 Session Info

```
sessionInfo()
## R version 3.4.2 Patched (2017-10-07 r73498)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows Server 2012 R2 x64 (build 9600)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=C
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] RColorBrewer_1.1-2         pcaExplorer_2.4.0
##  [3] ggplot2_2.2.1              DESeq2_1.18.0
##  [5] SummarizedExperiment_1.8.0 DelayedArray_0.4.0
##  [7] matrixStats_0.52.2         GenomicRanges_1.30.0
```

```
##  [9] GenomeInfoDb_1.14.0        IRanges_2.12.0
## [11] S4Vectors_0.16.0           doParallel_1.0.11
## [13] iterators_1.0.8            foreach_1.4.3
## [15] DASC_1.0.0                 cvxclustr_1.1.1
## [17] igraph_1.1.2               Matrix_1.2-11
## [19] NMF_0.20.6                 cluster_2.0.6
## [21] rngtools_1.2.4             pkgmaker_0.22
## [23] registry_0.3               Biobase_2.38.0
## [25] BiocGenerics_0.24.0        knitr_1.17
## [27] BiocStyle_2.6.0
##
## loaded via a namespace (and not attached):
##  [1] colorspace_1.3-2       rprojroot_1.2           htmlTable_1.9
##  [4] XVector_0.18.0         base64enc_0.1-3         d3heatmap_0.6.1.1
##  [7] topGO_2.30.0           ggrepel_0.7.0           DT_0.2
## [10] bit64_0.9-7            AnnotationDbi_1.40.0    codetools_0.2-15
## [13] splines_3.4.2          geneplotter_1.56.0      Formula_1.2-2
## [16] gridBase_0.4-7         annotate_1.56.0         GO.db_3.4.2
## [19] png_0.1-7              pheatmap_1.0.8          shinydashboard_0.6.1
## [22] graph_1.56.0           shiny_1.0.5             compiler_3.4.2
## [25] GOstats_2.44.0         backports_1.1.1         assertthat_0.2.0
## [28] lazyeval_0.2.1         limma_3.34.0            acepack_1.4.1
## [31] htmltools_0.3.6        prettyunits_1.0.2       tools_3.4.2
## [34] Category_2.44.0        gtable_0.2.0            glue_1.2.0
## [37] GenomeInfoDbData_0.99.1 reshape2_1.4.2         Rcpp_0.12.13
## [40] crosstalk_1.0.0        stringr_1.2.0           mime_0.5
## [43] XML_3.98-1.9           shinyAce_0.2.1          zlibbioc_1.24.0
## [46] scales_0.5.0           shinyBS_0.61            RBGL_1.54.0
## [49] SparseM_1.77           yaml_2.1.14             memoise_1.1.0
## [52] gridExtra_2.3          biomaRt_2.34.0          rpart_4.1-11
## [55] latticeExtra_0.6-28    stringi_1.1.5           RSQLite_2.0
## [58] genefilter_1.60.0      checkmate_1.8.5         BiocParallel_1.12.0
## [61] rlang_0.1.2            pkgconfig_2.0.1         bitops_1.0-6
## [64] evaluate_0.10.1        lattice_0.20-35         purrr_0.2.4
## [67] labeling_0.3           htmlwidgets_0.9         bit_1.1-12
## [70] AnnotationForge_1.20.0 GSEABase_1.40.0         plyr_1.8.4
## [73] magrittr_1.5           bookdown_0.5            R6_2.2.2
## [76] Hmisc_4.0-3            DBI_0.7                 foreign_0.8-69
## [79] survival_2.41-3        RCurl_1.95-4.8          nnet_7.3-12
## [82] tibble_1.3.4           rmarkdown_1.6           progress_1.1.2
## [85] locfit_1.5-9.1         grid_3.4.2              data.table_1.10.4-3
## [88] Rgraphviz_2.22.0       blob_1.1.0              threejs_0.3.1
## [91] digest_0.6.12          xtable_1.8-2            tidyr_0.7.2
## [94] httpuv_1.3.5           munsell_0.4.3
```