Normalisation Example using Internal CTCF Peaks as a Standard (No-linear Fit)

Andrew Holding

7/8/2017

Introduction

This is a worked example of our alternative method to normalise ER ChIP-seq data using CTCF binding to provide an internal control. Unlike our previous example, this normalisation is using only DESeq Size Factors. This method was found to be less effective but was used for comparision in the supporting publication. This markdown builds on Example 001 and therefore that should be read first. Example 001 also explains the pre-processing steps.

Load convenience functions

These functions facilitate the normalisation of data.

```
source('../package/brundle.R')
```

Apply settings

```
jg.controlMinOverlap      <- 5
jg.controlSampleSheet     <- "samplesheet/samplesheet_SLX14438_hs_CTCF_DBA.csv"
jg.experimentSampleSheet  <- "samplesheet/samplesheet_SLX14438_hs_ER_DBA.csv"
```

Load control and experimental counts from Example 001

In this example DiffBind was simply used as a convenient way to extract peak counts and provided consisteancy for comparision between methods.

To keep file sizes down, counts are provided as a Rdata file rather than as raw data.

```
filename<-"Rdata/example_001_SLX-14438_dba_human_ER_CTCF.rda"
if(!file.exists(filename)){
  dbaExperiment <- jg.getDba(jg.experimentSampleSheet, bRemoveDuplicates=TRUE)
  dbaControl    <- jg.getDba(jg.controlSampleSheet, bRemoveDuplicates=TRUE)
  save(dbaExperiment,dbaControl,file=filename)
} else {
  load(filename)
}
```

The actual DESeq Pipeline starts simply needs a matrix so we convert from DiffBind and save as a CSV file to provide a starting point.

```
filename<-"csv/experimentalPeakset.csv"
if(!file.exists(filename)){
  ## Extract Peak set from DiffBind
  jg.experimentPeakset <- jg.dbaGetPeakset(dbaExperiment)
```

```
  #Convert Peakset to DESeq Workflow
  jg.experimentPeaksetDeSeq<-jg.convertPeakset(jg.experimentPeakset)
  #Save to CSV

  write.csv(jg.experimentPeaksetDeSeq,file=filename)
}

#Repeat for control samples.

filename<-"csv/controlPeakset.csv"
if(!file.exists(filename)){
jg.controlPeakset    <- jg.dbaGetPeakset(dbaControl)
jg.controlPeaksetDeSeq<-jg.convertPeakset(jg.controlPeakset)
write.csv(jg.controlPeaksetDeSeq,   file=filename)
}
```

DESeq Pipeline starting from CSV

Load CSV dataset in DESeq format

Here we use the CSV files that the example code above generated; however, it is possible to generate the input data through a wide variety of methods and we have provided only one.

```
jg.controlPeaksetDeSeq <- read.csv( file="csv/controlPeakset.csv"
                                    ,check.names=FALSE, row.names = 1)
jg.experimentPeaksetDeSeq<- read.csv(file="csv/experimentalPeakset.csv"
                                    ,check.names=FALSE, row.names = 1)
```

To normalize the data, we use the DESeq2 function to estimate the size factors. This is a very simple method, and it provides some degree of normalization. We also use the original sample sheet to get the different conditions.

```
#Establish size factors directly from control data
jg.controlSizeFactors = estimateSizeFactorsForMatrix(jg.controlPeaksetDeSeq)

#Get conditions dataframe for DeSeq
jg.conditions <- read.csv(file=jg.controlSampleSheet, header=TRUE, sep=",")['Condition']

#Run DeSeq on control
jg.controlDeSeq<-jg.runDeSeq(jg.controlPeaksetDeSeq, jg.conditions,jg.controlSizeFactors)
jg.controlResultsDeseq   = results(jg.controlDeSeq)

#Run DeSeq on experiment
jg.experimentDeSeq<-jg.runDeSeq(jg.experimentPeaksetDeSeq, jg.conditions,jg.controlSizeFactors)
jg.experimentResultsDeseq   = results(jg.experimentDeSeq)
```

Export results as CSV

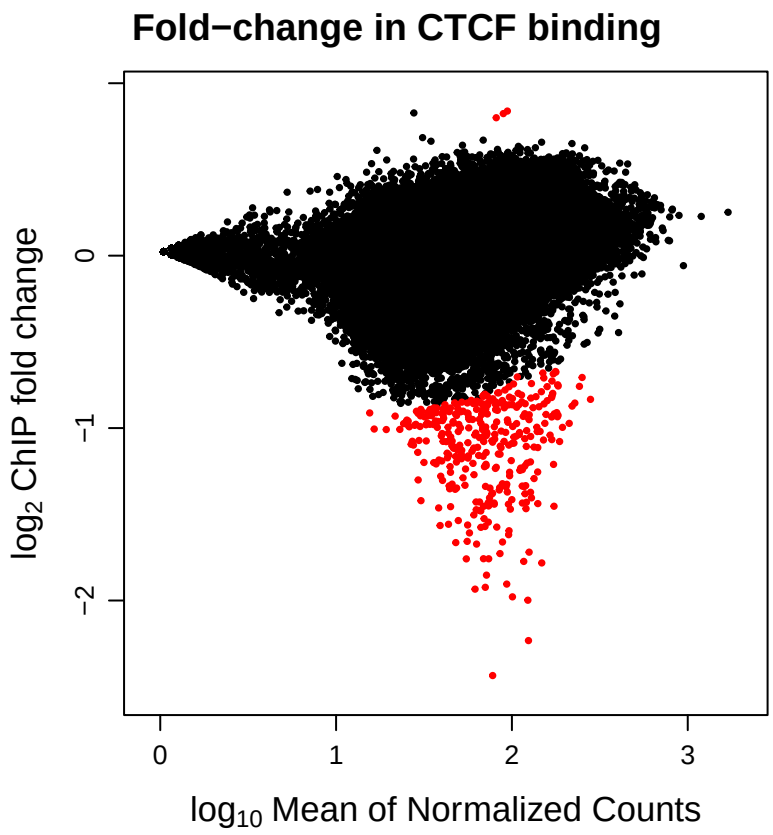
We can export the DESeq2 results in CSV format.

```
write.csv(file="results/Example_002.csv", jg.experimentResultsDeseq)
```

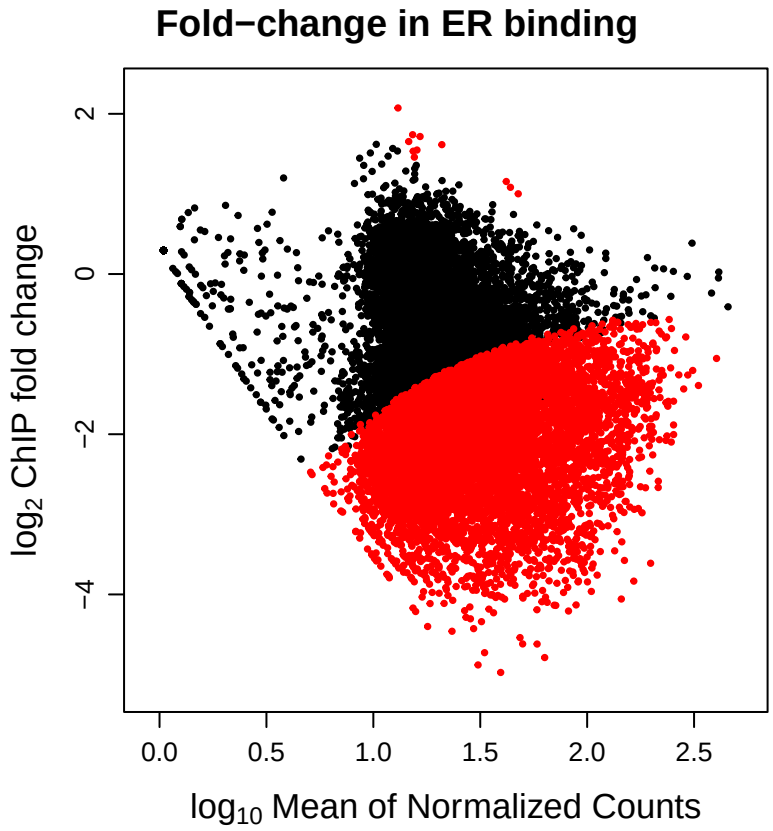
Plots

The Brundle package also contains some convenience functions for plotting the output of DESeq2 in place of DiffBind. These generate the following figures.

```
jg.plotDeSeq(jg.controlResultsDeseq,
               p=0.01,
               title.main="Fold-change in CTCF binding",
               flip=T
               )
```




```
jg.plotDeSeq(jg.experimentResultsDeseq,
              p=0.01,
              title.main="Fold-change in ER binding",
              flip=T
              )
```




```
#Draw Combined figure.
  jg.plotDeSeqCombined(jg.controlResultsDeseq,
                       jg.experimentResultsDeseq,
                       title.main="ER and CTCF Binding Folding changes on ER treatment",
                       p=0.01,flip=TRUE)
```

ER and CTCF Binding Folding changes on ER treatment