Supplementary Code: *Myc Depletion Induces a Pluripotent Dormant State Mimicking Diapause*

Scognamiglio et al.

Abstract

This document contains the documented workflow used to analyse the RNA-seq data presented in the manuscript titled: *Myc Depletion Induces a Pluripotent Dormant State Mimicking Diapause*.

The data package accompanying this vignette, *Scognamiglio2015*, provides the DESeq2DataSet objects summarizing the information from the RNA-seq experiment.

1 Testing for differential expression with DESeq2

We test for differential expression between the different conditions using the clone of the experiments as a blocking factor. Since the *DESeq2* version changed from the time the analysis was done to the publication of the manuscript, this section demonstrates the code that was used for the analysis, but the number of differentially expressed genes might slightly vary depending on the *DESeq2* version.

```
data("DXD_MYC")
```

```
se <- estimateSizeFactors( se )
se <- estimateDispersions( se )
se <- nbinomLRT( se, reduced= ~ clone, full=~ clone + mix )
```

We load the pre-computed result table with the differential expression results.

```
path <- system.file(package="Scognamiglio2015")
deseq2Results <-
    read.delim(
        file.path(path, "extdata", "resultNumbers.txt"),
        header=TRUE)

deGenes <-
    rownames( deseq2Results )[which(deseq2Results$padj < 0.1)]
```

We group the data according to their expression levels relative to the mean expression level across all the conditions.

```
sp <-
    split( seq(along=colData(se)$mix),
          as.character(colData(se)$mix ))

vst <- varianceStabilizingTransformation( se, blind=TRUE)
vsd <- assay(vst)

mns <- sapply( sp, function(x){
   rowMeans( vsd[,x] )
})

relative <-  mns - rowMeans(mns)
relative <- relative[deGenes,]

classes <-
    do.call(expand.grid,
            sapply( colnames(relative),
                   function(ti) c( -1, 1 ), simplify=FALSE ) )

geneClasses <-
    apply(relative %*% t( as.matrix(classes) ), 1, which.max )

df <- data.frame( classes, number=tabulate( geneClasses, 64) )
#df
#write.table( df, file="classesDef.txt")

colnames(df) <- c(colnames(classes), "number")
splitted <- split( names( geneClasses ), geneClasses )


ens <- as.character(deseq2Results$geneName)
names(ens) <- rownames(deseq2Results)

splittedGenes <- lapply( splitted, function(x){ ens[x] } )
```

2 Testing for GO enrichments

We load and define the functions to test for enrichment of GO cathegories within our set of differentially expressed genes, as compared to a background set of genes matched for expression strength.

```
source(file.path(path, "extfunction", "berndts_go_code.R"))

testForGOEnrichments <-
```

```
    function( cds, deGenes, back=NULL, GO=NULL){
  if( is.null(back) ){
    df <- data.frame(
        significant=as.numeric( names(rowData(cds)) %in% deGenes),
        counts=rowMeans( counts(cds) ) )
    mm <- matchit( significant ~ counts,
                  df, method="nearest",
                  distance="mahalanobis" )
    back <- mm$match.matrix[,1]
  }
  if( is.null( GO )){
    GO <- downloadGO(background=c(back, deGenes))
  }
  GO3 <- GO[as.character( GO$ensembl_gene_id ) %in% deGenes,]
  toTest <- unique( GO3[,"go_id"] )
  toTest <- toTest[!toTest %in% c("", "all")]
  alltests <- mclapply( toTest, function(test){
  gos <-
      as.character(GO$ensembl_gene_id)[
         which( as.character(GO$go_id) %in% test )]
  gos3 <- as.character(GO3$ensembl_gene_id)[
             which( as.character(GO3$go_id) %in% test )]
  a <- sum( deGenes %in% gos )
  b <- sum( back %in% gos )
  mat <- rbind(de=c(a, length(deGenes)-a), back=c(b, length(back) -  b))
    colnames(mat) <- c("inGO", "notinGO")
    ft <- fisher.test(mat, alternative="greater")
      c(de=mat["de",],
        back=mat["back",],  ft$estimate,
        `pvalue`=ft$p.value,
        `genes`=paste(ens[gos3], collapse=",") )
    }, mc.cores=10 )
    names(alltests) <- toTest
    goInfo <-
        read.delim(
           file.path(path,
                     "extdata",
                     "gene_ontology_ext_formated.tab"),
           header=FALSE, stringsAsFactors=FALSE)
    rownames( goInfo ) <- goInfo$V1
    enriched <- as.data.frame( do.call(rbind, alltests) )
    rownames(enriched) <- toTest
    enriched[,"pvalue"] <-
        as.numeric( as.character(enriched[,"pvalue"]) )
```

```r
    enriched$padjust <-
        p.adjust( enriched[,"pvalue"], method="BH")
    enriched$name <-
        goInfo[rownames(enriched), "V3"]
    enriched$description <-
        goInfo[rownames(enriched), "V4"]
    enriched
}

data(GO)
```

We define a background with the same expression strength distribution as the genes detected to be differentially expressed.

```r
library(MatchIt)

df <-
    data.frame(
        significant=as.numeric(rownames(se) %in% deGenes),
        means=rowMeans(counts(se, normalized=TRUE)))
mm <-
    matchit(
        significant ~ means, df,
        method="nearest", distance="mahalanobis" )

back <-
    mm$match.matrix[,1]
```

For each of the gene clusters, we test for over-representation of GO terms in our different groups of genes as compared to the background set of genes.

```r
enriched <- lapply( splitted, function(x){
   GOgood <- GO[GO$`ensembl_gene_id` %in% c(x, back),]
   testForGOEnrichments(se, x, back, GOgood)
})

names(enriched) <- names( splitted )

for( i in seq_along(enriched)){
  enriched[[i]]$pvalue[
      as.numeric( as.character(enriched[[i]]$`de.inGO` )) < 5] <- NA
  enriched[[i]]$padjust <-
      p.adjust(enriched[[i]]$pvalue, method="BH")
}

lapply(c("44", "48", "21", "17"),
       function(set){
  tb <-
      enriched[[set]][
         which(enriched[[set]]$padj < 0.1),
         c("name", "de.inGO", "genes", "padjust")]
  write.table( tb,
              quote=FALSE, row.names=TRUE,
              col.names=TRUE, sep="\t",
              file=paste0(set, "_go.table"))
})
```

```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
```

3 Pathways downregulated or up-regulated

Then, we estimate the DEseq2 shrinked log fold changes (base 2) between the dKO and the c-myc$^{(\Delta/\Delta)}$ N-myc$^{(\Delta/fl)}$. We use this fold changes in order to test, for each pathway annotated in wikipathways, if there is a difference on the distribution of log fold changes that is significantly different from zero.

```
data("entrez")
data("wikipathways")

seSub <- se[,grep("72h", colData(se)$mix)]
seSub <- DESeqDataSetFromMatrix( counts(seSub),
            DataFrame( condition=factor(colData(seSub)$mix) ), ~condition )
seSub <- estimateSizeFactors( seSub )
seSub <- estimateDispersions( seSub )
seSub <- nbinomWaldTest( seSub )


toTest <- unique( as.character( processesDF[,1] ) )

testForPathways <- function(dxdObject, processDataFrame,
                            convertID, tested, dfcolname="pathway",
                            dfcolname2="entrez", cores=1){
    pvals <-mclapply( tested, function(x){
        pr <- processDataFrame[processDataFrame[,dfcolname] %in% x,dfcolname2]
        ensPr <- names( convertID[convertID %in% pr] )
        rs <- dxdObject[rownames(dxdObject) %in% ensPr,"log2FoldChange"]
        rs <- rs[!is.na(rs)]
        if( length(rs) > 3 ){
            t.test( rs )$p.value
        }else{
            return(NA)
        }
    }, mc.cores=cores)
    pvals
}

pvals <- testForPathways( results(seSub), processesDF, entrez, toTest)

enriched <-
    toTest[
       which( p.adjust( unlist( pvals ), method="BH" ) < 0.1 )]


deseqRes <- results(seSub)

#deseqRes[1,]
#tapply(counts(seSub, normalized=TRUE)[1,], colData(seSub)£condition, mean)
#deseqRes

prepareFCDataFrame <- function(deseqResults){
    allPatterns <- c("PluriNetWork",
                     "Cytoplasmic_Ribosomal_Proteins",
                     "Translation_Factors",
                     "Transcription_Initiation",
                     "DNA_Replication",
                     "Focal_Adhesion",
                     "Integrin-mediated",
                     "IL-6")
    pathFC <- lapply(allPatterns,
                     function(pattern){
      pathway <-
          unique(processesDF[
             grep(pattern, processesDF$pathway),"pathway"])
      ent <- processesDF[
                processesDF$pathway %in% pathway,
                "entrez"]
      ensPath <- rownames(deseqRes)[
                    rownames(deseqRes) %in% names( entrez[entrez %in% ent] )]
      color <-
          ifelse( mean( deseqRes[ensPath,"log2FoldChange"], na.rm=TRUE) > 0,
                 "#cc000090", "#0000cc90")
      data.frame(
          `ensembl_gene_id`=ensPath,
          `log_fold_change`=deseqRes[ensPath,"log2FoldChange"],
          `pathway`=gsub("_", " ", gsub("Mm_|_WP\\S+", "", pathway)),
          `color`=color)
    })
    pathFC
    pathFC <- do.call( rbind, pathFC )
    pathFC$color <- as.character( pathFC$color )
    pathFC[pathFC$pathway %in% "PluriNetWork","color"] <- "#000000"
    pathFC
}

pathFC <- prepareFCDataFrame( deseqRes )
```

```
library(ggplot2)

plotPathwayDistributions <- function(fcDataFrame){
    p <- ggplot(fcDataFrame, aes(pathway, log_fold_change, fill=color))
    p <- p +
        geom_violin(trim=TRUE, scale="area") +
        geom_boxplot(width=.3, notch=.1, outlier.shape = NA) +
        scale_y_continuous(limits = c(-.6, .6)) +
        geom_abline(intercept = 0, slope = 0, col="#00000099", lwd=1) +
        xlab("") +
        scale_fill_manual( values = c("darkgray", "#377eb8", "#e41a1c")) +
        ylab(
 expression(log[2]~"("~frac("dKO (96h)","c-myc"^paste(Delta,"/",Delta)~~"N-myc"^paste(Delta
        theme( panel.background = element_blank(),
          panel.border = element_rect(colour = "black", fill=NA),
              legend.position="none",
          axis.ticks = element_line(colour="black"),
          axis.title.y=element_text(size=14),
          axis.text.x = element_text(size=13.5, colour="black",
              angle = 20, hjust = 1),
          axis.text.y = element_text(size=14, colour="black" ))
    p
}

plotPathwayDistributions( pathFC )
```

4 Comparison with diapaused embryo gene expression

We used processed data from diapaused embryos generated by *Boroviak et al* in order to compare their gene expression signature with our data. In particular, we chose some of the gene onthology cathegories that were described to be commonly downregulated in both the diapaused embryos as compared to embryos in E4.5 and the dKO experiment with respect to the WT embryonic stem cells.

We first estimated the moderated fold changes from DESeq2 on both datasets independently:

```
data("dxdDiapaused")
dxdDiapaused <- estimateSizeFactors( dxdDiapaused )
filter <- rowSums(counts(dxdDiapaused, normalized=TRUE)) > 0
dxdDiapaused <- dxdDiapaused[filter,]
both <- intersect(rownames(dxdDiapaused), rownames( se ) )
dxdDiapaused <- dxdDiapaused[rownames(dxdDiapaused) %in% both,]
dxdDiapaused <- estimateSizeFactors( dxdDiapaused )
dxdDiapaused <- estimateDispersions(dxdDiapaused)
dxdDiapaused <- nbinomWaldTest( dxdDiapaused )

diapausedRes <- results(dxdDiapaused, independentFiltering = FALSE)

seSub2 <- se[,colData(se)$mix %in% c( "CREplus72h", "ESB8NT0h")]
colData(seSub2) <- droplevels(colData(seSub2))
seSub2 <- seSub2[rownames( seSub2 ) %in% both,]
seSub2 <- estimateSizeFactors( seSub2 )
seSub2 <- estimateDispersions( seSub2 )
seSub2 <- nbinomWaldTest( seSub2 )
resultsdKOvsWT <- results( seSub2, independentFiltering=FALSE )
```

We define a function to plot the fold changes for both datasets, highligh genes from specific GO cathegories.

```
library(ggplot2)
library(gtable)
library(plyr)
library(dplyr)
#data(mtcars)

scatterBoxplot <- function(dataFrame, highlight=NULL){
    if( mean( dataFrame$x[highlight], na.rm=TRUE ) > 0 ){
       col2 <- "#e41a1c80"
    }else{
       col2 <- "#2166ac80"
    }
    if( mean( dataFrame$y[highlight], na.rm=TRUE ) > 0 ){
       col3 <- "#e41a1c80"
   }else{
       col3 <- "#2166ac80"
    }
    dataFrame$col <- ifelse( highlight, col3, "#87878730" )
    dataFrame <- dataFrame[rowSums( is.na(dataFrame) ) == 0,]
    p1 <- ggplot(dataFrame) +
        geom_point(aes(x, y), color="#87878730") +
        geom_point(aes(x, y), colour=col3, subset=.(col == col3) ) +
    geom_hline(yintercept=0, colour="#00000090", lwd=1.2) +
    geom_vline(xintercept=0, colour="#00000090", lwd=1.2) +
    xlab(expression(log[2]~"fold change ( Diapaused / E4.5 )")) +
    ylab(expression(log[2]~"fold change ( dKO / WT )"))+
    theme(plot.margin = unit(c(0.2, 0.2, 0.5, 0.5), "lines"),
          panel.background = element_rect(colour="black", fill="white"),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          axis.text=element_text(size=14,color="black"),
          axis.title=element_text(size=14,color="black"),
          legend.position="none") + ylim(-2.5, 2.5) +
    xlim(-7.5, 7.5)
    p2 <- ggplot(dataFrame, aes(x = factor(1), y = x)) +
        geom_violin(subset=.(col == col3), fill=col2) +
            geom_boxplot(outlier.colour = NA, width=.5,subset=.(col == col3),fill=col2) +
            geom_hline(yintercept=0, colour="#00000090", lwd=1.2) + ylim(-7.5, 7.5) +
    coord_flip() +
        theme(axis.text = element_blank(),
              axis.title = element_blank(),
              axis.ticks = element_blank(),
              plot.margin = unit(c(1, 0.2, -0.5, 0.5), "lines"),
              panel.background = element_rect(colour="black", fill="white"),
              panel.grid.major = element_blank(),
              panel.grid.minor = element_blank())
                                                  # Vertical marginal boxplot - to appear at the ri
    p3 <- ggplot(dataFrame, aes(x = factor(1), y = y)) +
        geom_violin(subset=.(col == col3), fill=col3) +
            geom_boxplot(outlier.colour = NA, width=.5,subset=.(col == col3), fill=col3) +
            geom_hline(yintercept=0, colour="#00000090", lwd=1.2) +
    theme(axis.text = element_blank(),
          axis.title = element_blank(),
          axis.ticks = element_blank(),
          plot.margin = unit(c(0.2, 1, 0.5, -0.5), "lines"),
          panel.background = element_rect(colour="black", fill="white"),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank()) + ylim(-2.5, 2.5)
    gt1 <- ggplot_gtable(ggplot_build(p1))
    gt2 <- ggplot_gtable(ggplot_build(p2))
    gt3 <- ggplot_gtable(ggplot_build(p3))
    maxWidth <- unit.pmax(gt1$widths[2:3], gt2$widths[2:3])
    maxHeight <- unit.pmax(gt1$heights[4:5], gt3$heights[4:5])
    # Set the maximums in the gtables for gt1, gt2 and gt3
    gt1$widths[2:3] <- as.list(maxWidth)
    gt2$widths[2:3] <- as.list(maxWidth)
    gt1$heights[4:5] <- as.list(maxHeight)
    gt3$heights[4:5] <- as.list(maxHeight)
    gt <- gtable(widths = unit(c(7, 1.5), "null"),
                 height = unit(c(1.5, 7), "null"))
    gt <- gtable_add_grob(gt, gt1, 2, 1)
    gt <- gtable_add_grob(gt, gt2, 1, 1)
    gt <- gtable_add_grob(gt, gt3, 2, 2)
    # And render the plot
    grid.newpage()
    grid.draw(gt)
}
```

We test if the moderated fold changes (dKO vs WT) distribution of the genes from a specific GO cathegory deviate significantly from zero. We also tested if these GO cathegories were changing significantly in the diapaused embryos as compared to the E4.5 embryos. We selected and plotted a subset of these cathegories that explain the functional similarities that we observed between dKO and Diapaused embryos, e.g. dormancy-like states.

```
ensemblIDs <- names(ens)
names(ensemblIDs) <- names(ens)
goToTest <- unique(GO$go_id)
pvalsGOWTvsdKO <- testForPathways( as.data.frame( resultsdKOvsWT ),
                                  GO, ensemblIDs, goToTest, dfcolname="go_id",
                                  dfcolname2="ensembl_gene_id", cores=10)

pvalsGODiapaused <- testForPathways( as.data.frame( diapausedRes ),
                                  GO, ensemblIDs, goToTest, dfcolname="go_id",
                                  dfcolname2="ensembl_gene_id", cores=10)

enrichedGO <- goToTest[which(p.adjust( pvalsGOWTvsdKO, method="BH") < 0.001)]
enrichedGO <- enrichedGO[!enrichedGO %in% c("", "all")]
```

```
dfFoldChanges <-
    data.frame( x=-diapausedRes$log2FoldChange,
                y=-resultsdKOvsWT$log2FoldChange )
rownames( dfFoldChanges ) <- rownames(diapausedRes)
```

Selection of GOs,

```
selectedGOs <- c("GO:0006260", "GO:0006396", "GO:0042254",
                 "GO:0006412", "GO:0008380", "GO:0005681",
                 "GO:0005178", "GO:0038023")
fileNames <- lapply(selectedGOs, function(pa){
   print( pa )
    ensemblInPathway <- GO[GO$go_id %in% pa,1]
    name <- gsub(":", "", pa)
    png( file.path("reports", "goSelection",
                   paste0(name, ".png")),
        height=4.6, width=4.7, unit="in", res=200)
    scatterBoxplot( dfFoldChanges,
        highlight=rownames(diapausedRes) %in% ensemblInPathway)
    dev.off()
})

## [1] "GO:0006260"
## [1] "GO:0006396"
## [1] "GO:0042254"
## [1] "GO:0006412"
## [1] "GO:0008380"
## [1] "GO:0005681"
## [1] "GO:0005178"
## [1] "GO:0038023"
```

5 c-myc and N-myc expression

```
df <- do.call(rbind,
        lapply(c("Myc", "Mycn"), function(x){
  df <- data.frame(
      counts=counts(se, normalized=TRUE)[names( ens[ ens %in% x] ),],
      gene=x,
      clone=colData(se)$clone,
      mix=colData(se)$mix)
  return(df)
}) )
levels( df$clone ) <- c("Replicate 1", "Replicate 2")
levels( df$gene ) <- c("c-myc", "N-myc")
```

```
df$mix <- factor(df$mix, levels( df$mix )[c(6, 1, 2, 4, 3, 5)])
ggplot(df, aes(x=mix, y=counts, fill=mix, colour="black")) +
    geom_bar(stat="identity") +
    facet_grid(gene~clone, scale="free") + xlab("") +
    ylab("Number of read fragments normalized\nfor sequencing depth") +
    scale_fill_manual( values = c("#000000","#00000090",
                            "#a6cee3", "#1f78b4", "#fb9a99", "#e31a1c")) +
    scale_colour_manual(values="black") +
    theme( panel.background = element_blank(),
          panel.border = element_rect(colour = "black", fill=NA),
          legend.position="none",
          axis.ticks = element_line(colour="black"),
          axis.title.y=element_text(size=14),
          axis.text.x = element_blank(),
          axis.ticks.x = element_blank(),
          strip.text = element_text(size=14),
          strip.text.y= element_text(face = "italic"),
          axis.text.y = element_text(size=14, colour="black" ) )
```

6 Heatmaps

```
pl <-
   ens[match( c("Zfp42", "Nanog", "Pou5f1",
                "Esrrb", "Tbx3","Klf4","Sox2","Klf2"), ens)]
pl2 <-
   ens[match( c("Sall4", "Kit", "Pecam1",
                "Utf1", "Dppa4","Stat3","Nr0b1","Sfrp1",
                "Crebbp", "Tcl1", "Dazl",
                "Fgfr1", "Dppa2", "Mycn", "Fgf4",
                "Lin28a","Klf5", "Tert", "Tcf3",
                "Eras", "Id2", "Prdm16", "Id1",
                "Fgf10","Lin28b", "Bmp7", "Sox4",
                "Myc", "Wnt7a", "Sfpi1", "Zhx2",
                "Nog"), ens)]
gl <- ens[match( c("Ifitm3", "Prdm14",
                   "Prdm1", "Ctcfl"), ens )]
ed <- ens[match( c("Gdf3", "Ctnnb1", "Nodal",
                   "Nckap1", "Arc", "Ext1",
                   "Hhex", "Hnf1b", "Gata4",
                   "Gsc", "Sox7", "Cfc1",
                   "Gata6", "Foxa2"), ens) ]
ec <- ens[match( c("Krt18", "Pax6", "Tgm3",
                   "Nes", "Col5a1", "Dsp",
                   "Otx2", "Evpl", "Fgf5",
                   "Tcf15", "Runx3", "Sox18"), ens) ]
ms <- ens[match( c("T", "Srf", "Bmp4",
                   "Bmpr2", "Runx1", "Msx2",
                   "Cdx4", "Bmp6", "Cebpa",
                   "Tal1"), ens) ]
all <- c(names(pl), names(pl2),
         names(gl), names(ed),
         names(ec), names(ms))


rlogData <- rlog(se)

sp <-
    split( seq(along=colData(se)$mix),
          as.character(colData(se)$mix ))

vsd <- assay(rlogData)

mns <- sapply( sp, function(x){
   rowMeans( vsd[,x] )
})


relative <-  mns - rowMeans(mns)
plExpr <- relative[all,]
plExpr <- mns[all,]
plur <- rep( rep(
    c("PL", "PL2", "GL", "EN", "EC", "MS"),
    c(length(pl), length(pl2),
      length(gl), length(ed),
      length(ec), length(ms)) ),
    ncol(plExpr))
dfPlur <- data.frame(
    expression=as.vector(plExpr),
    condition=rep(colnames(plExpr), each=nrow(plExpr)),
    gene=rep(ens[all], ncol(plExpr)),
    signature=plur
    )
dfPlur$gene <- factor( dfPlur$gene, levels=rev(ens[all]) )
dfPlur$condition <- factor(dfPlur$condition,
                           levels( dfPlur$condition )[c(6, 1, 2, 4, 3, 5)])
toAdd <- ifelse( names( ens[match( levels( dfPlur$gene ), ens )] ) %in%
                deGenes, "*", "")
levels(dfPlur$gene) <- paste0(levels(dfPlur$gene), toAdd)
library(RColorBrewer)
cols <- colorRampPalette(brewer.pal(9, "YlOrBr"))(100)
library(lattice)
library(scales)
library(latticeExtra)
theme.novpadding <-
   list(
       layout.heights =
        list(top.padding = 0,
            main.key.padding = 0,
            key.axis.padding = 0,
            axis.xlab.padding = 0,
            xlab.key.padding = 0,
            key.sub.padding = 0,
            bottom.padding = 0),
        layout.widths =
        list(left.padding = 0,
            key.ylab.padding = 0,
            ylab.axis.padding = 0,
            axis.key.padding = 0,
            right.padding = 0))
allLattice <- lapply( levels(dfPlur$signature), function(x){
    dfPlur2 <- dfPlur[dfPlur$signature %in% x,]
    dfPlur2 <- droplevels(dfPlur2)
    lp <- levelplot(expression~condition * gene| signature,
                    dfPlur2, aspect="iso", colorkey=NULL, col.regions=cols,
                    strip=strip.custom(bg="#E8E8E8"),
                    at=seq(0, 16.7, length.out=101),
                    ylab=list(label=""),
                    xlab=list(label=""),
                    scales=list(
                        x=list(at=NULL),
                        y=list(cex=1.15, alternating = 1, fontface="italic") ),
                    par.strip.text=list(cex=1.3),
                    par.settings = theme.novpadding)
})
names(allLattice) <- levels(dfPlur$signature)
print( allLattice[["PL"]],
      position=c(0, .75, .5, 1), more=TRUE)
print( allLattice[["PL2"]],
      position=c(0, 0.03, .5, .75), more=TRUE)
print( allLattice[["GL"]],
      position=c(.4, .835, .9, 1), more=TRUE)
print( allLattice[["ED"]],
      position=c(.4, .525, .9, .865), more=TRUE)

## NULL

print( allLattice[["EC"]],
      position=c(.4, .267, .9, .575), more=TRUE)
print( allLattice[["MS"]],
      position=c(.4, .03, .9, .3), more=TRUE)
draw.colorkey(
  key=list(
    col=cols,
    at=seq( 0, 16.7, length.out=101),
    labels=list(labels=comma(as.integer(2**seq( 0, 16.7, length.out=6))),
                  at=seq( 0, 16.7, length.out=6),
                  cex=1.15)
  ),
  draw=TRUE,
  vp=grid::viewport(
    x=grid::unit(0.915, "npc"),
    y=grid::unit(0.5, "npc"),
    height=grid::unit(0.2, "npc")
  )
)

## frame[plot_05.colorkey.frame]
```

7 Cell cycle genes heatmap

```
gns <- c("Ncl", "Psmd1", "Hmga1",
         "Rfc1", "Dnajc11", "Pola1",
         "Mak16",
         "Syncrip", "Orc1", "Mki67ip",
         "Gabpa", "Pcm1", "Kif15",
         "Atm", "Skp2", "Otud4",
         "Myo19", "Aatf", "Mre11a",
         "Mycn",
         "Mphosph9", "Dis3", "E2f3",
         "Sgk3")
plExpr <- relative[names( ens[match( gns, ens )] ),]
plExpr <- mns[names( ens[match( gns, ens )] ),
              c("ESB8NT0h","CREminus72h","CREplus72h")]
plExpr <- plExpr - rowMeans( plExpr )
```

```
cols <- colorRampPalette(brewer.pal(9, "RdBu"))(100)
dfPlur <- data.frame(
    expression=as.vector(plExpr),
    condition=rep(colnames(plExpr), each=nrow(plExpr)),
    gene=rep(ens[match(gns, ens)], ncol(plExpr))
    )
dfPlur$condition <- factor( dfPlur$condition,
                           levels(dfPlur$condition)[c(3, 1, 2)])
lp <- levelplot(expression~gene*condition,
                    dfPlur, aspect="iso", col.regions=cols,
                    strip=strip.custom(bg="#E8E8E8"),
                    at=seq(-1, 1, length.out=101),
                    ylab=list(label=""),
                    xlab=list(label=""),
                    colorkey=list(height=1.5),
                    par.settings=list(fontsize=list(text=18)),
                    scales=list(
                        y=list(at=NULL),
                        x=list(alternating = 1, rot=90, fontface="italic") ))
print(lp)
```

8 Session Info

```
sessionInfo()

## R Under development (unstable) (2015-11-17 r69646)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: CentOS release 6.5 (Final)
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
##  [4] LC_COLLATE=en_US.UTF-8     LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
##  [1] grid      parallel  stats4    stats     graphics  grDevices utils     datasets
##  [9] methods   base
##
## other attached packages:
##  [1] latticeExtra_0.6-26       scales_0.3.0               lattice_0.20-33
##  [4] RColorBrewer_1.1-2        dplyr_0.4.3                plyr_1.8.3
##  [7] gtable_0.1.2              ggplot2_1.0.1              Scognamiglio2015_1.0.1
## [10] DESeq2_1.11.1              RcppArmadillo_0.6.300.2.0  Rcpp_0.12.2
## [13] SummarizedExperiment_1.1.2 Biobase_2.31.0             GenomicRanges_1.23.3
## [16] GenomeInfoDb_1.7.3         IRanges_2.5.7              S4Vectors_0.9.11
## [19] BiocGenerics_0.17.2        knitr_1.11                 BiocInstaller_1.21.2
##
## loaded via a namespace (and not attached):
##  [1] genefilter_1.53.0    locfit_1.5-9.1       reshape2_1.4.1       splines_3.3.0
##  [5] colorspace_1.2-6     survival_2.38-3      XML_3.98-1.3         foreign_0.8-66
##  [9] DBI_0.3.1            BiocParallel_1.5.0   lambda.r_1.1.7       stringr_1.0.0
## [13] zlibbioc_1.17.0      munsell_0.4.2        futile.logger_1.4.1  evaluate_0.8
## [17] labeling_0.3         geneplotter_1.49.0   AnnotationDbi_1.33.1 highr_0.5.1
## [21] proto_0.3-10         acepack_1.3-3.3      xtable_1.8-0         formatR_1.2.1
## [25] Hmisc_3.17-0         annotate_1.49.0      XVector_0.11.1       gridExtra_2.0.0
## [29] BiocStyle_1.9.2      digest_0.6.8         stringi_1.0-1        tools_3.3.0
## [33] magrittr_1.5         RSQLite_1.0.0        Formula_1.2-1        cluster_2.0.3
## [37] futile.options_1.0.0 MASS_7.3-45          assertthat_0.1       R6_2.1.1
## [41] rpart_4.1-10         nnet_7.3-11
```