Transcriptomic and physiological responses to seasonal and diurnal cycles in Ostreococcus

Francisco J. Romero-Campero & Ana B. Romero-Losada

August 11, 2020

Experimental design

In this project we study the transcriptomic and physiological responses to seasonal and diurnal cycles in the picoeukaryote *Ostreococcus tauri*. Our favourite microalgae was grown in 1.8L column photochemostats under long day conditions (16 hours light : 8 hours dark) representing a summer day and under short day conditions (8 hours light : 16 hours dark) simulating a winter day. After four weeks of entrainment under each condition samples were collected for three days every four hours. Then the program controlling the light in the photochemostats was set to free running conditions consisting on continuous light and samples were again collected every four hours for two days.

Load data and principal component analysis.

The matrix containing the gene expression data analyzed in this study can be downloaded from **this link from the GEO data base**. Make sure to uncompress this file and rename it to **gene_expression.tsv**. First, the gene expression data is loaded and converted into a numeric matrix setting the rownames to gene ids.

```
gene.expression <- read.table(file ="gene_expression.tsv",sep="\t",header=T,as.is=T)
head(gene.expression[,1:7])
```

```
##               X  ld_zt00_1  ld_zt04_1 ld_zt08_1 ld_zt12_1 ld_zt16_1 ld_zt20_1
## 1 ostta01g00010   9.664395  25.045279  30.81788  44.97020  32.61381  30.13931
## 2 ostta01g00020  15.688867   9.913202  10.36669  14.64151  11.66096  21.09974
## 3 ostta01g00030  16.108133  11.134813  13.85848  38.54982  24.16540  16.82981
## 4 ostta01g00040  59.247765  32.837433  27.26293  42.82092  52.59695  87.45710
## 5 ostta01g00050  27.909069  14.945981  12.04561  20.26003  28.25942  25.53805
## 6 ostta01g00060 248.044205 145.486374  68.38238  66.96495 224.52632 246.65002
```

```
gene.ids <- gene.expression$X

gene.expression <- as.matrix(gene.expression[,2:ncol(gene.expression)])
rownames(gene.expression) <- gene.ids
head(gene.expression[,1:6])
```

```
##                ld_zt00_1  ld_zt04_1 ld_zt08_1 ld_zt12_1 ld_zt16_1 ld_zt20_1
## ostta01g00010   9.664395  25.045279  30.81788  44.97020  32.61381  30.13931
## ostta01g00020  15.688867   9.913202  10.36669  14.64151  11.66096  21.09974
## ostta01g00030  16.108133  11.134813  13.85848  38.54982  24.16540  16.82981
## ostta01g00040  59.247765  32.837433  27.26293  42.82092  52.59695  87.45710
## ostta01g00050  27.909069  14.945981  12.04561  20.26003  28.25942  25.53805
## ostta01g00060 248.044205 145.486374  68.38238  66.96495 224.52632 246.65002
```

The current version of *Ostreococcus tauri* genome available from **here** identifies 7668 genes. In our experiment

only 8 genes were never expressed and 40 genes never presented an expression level greater than 1 FPKM. This shows that practically the entire transcriptome of Ostreococcus is expressed under the seasonal and diurnal cycles studied in this project.

```
ostta.genes <- rownames(gene.expression)
number.genes <- length(ostta.genes)
number.genes
```

```
## [1] 7668
```

```
length(which(apply(X = gene.expression,MARGIN = 1,FUN = max) == 0))
```

```
## [1] 8
```

```
length(which(apply(X = gene.expression,MARGIN = 1,FUN = max) < 1))
```

```
## [1] 40
```

We focus on the data generated under long and short days cycles by extracting them from the gene expression matrix. The resulting matrix has 7668 rows representing genes and 36 columns. This number of columns correspond to 36 different data points, each day is represented by 6 data points and we took samples for three days under both long and short day conditions.

```
ld.zt <- paste("ld",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")
ld.zt.i <- sapply(X = ld.zt,FUN = function(x){ paste(x,1:3,sep="_")})
sd.zt <- paste("sd",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")
sd.zt.i <- sapply(X = sd.zt,FUN = function(x){ paste(x,1:3,sep="_")})

ld.sd.gene.expression <- gene.expression[,c(ld.zt.i,sd.zt.i)]
head(ld.sd.gene.expression[,1:6])
```

```
##                ld_zt00_1  ld_zt00_2  ld_zt00_3  ld_zt04_1  ld_zt04_2 ld_zt04_3
## ostta01g00010   9.664395   8.753878  21.967098  25.045279  16.192572  29.56268
## ostta01g00020  15.688867  14.411269  15.126039   9.913202   9.555948  17.26569
## ostta01g00030  16.108133  18.037844   5.178177  11.134813  18.246208  13.43679
## ostta01g00040  59.247765  62.436951  50.582394  32.837433  20.027222  27.73791
## ostta01g00050  27.909069  27.673790  28.704229  14.945981  18.921703  23.65231
## ostta01g00060 248.044205 304.855804 101.986931 145.486374 134.537064 113.76479
```

```
dim(ld.sd.gene.expression)
```

```
## [1] 7668   36
```

We perform **Principal Component Analysis** and a **Hierarchical clustering** in order to uncover the underlying structure in our data. We use the packages FactoMineR and factoextra and reformat the data as needed for the function PCA.

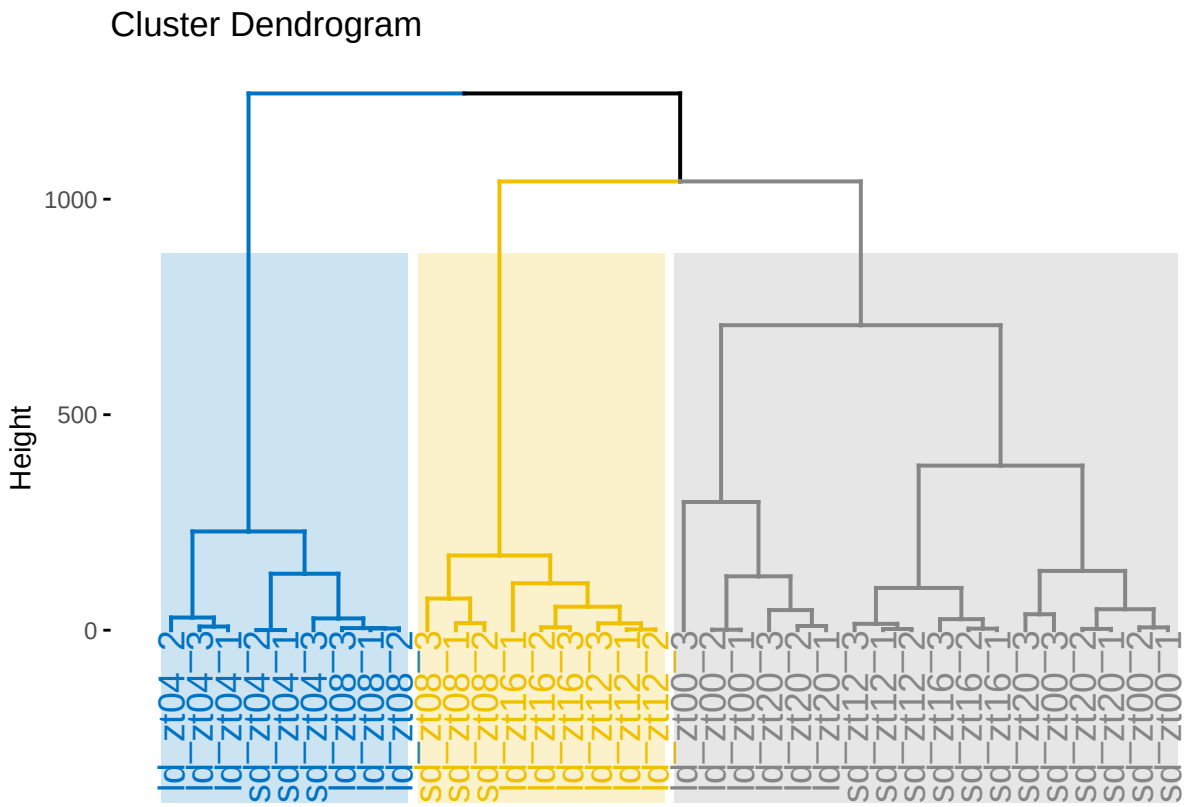
```
library(FactoMineR)
library(factoextra)
```

```
## Loading required package: ggplot2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
pca.gene.expression.ld.sd <- data.frame(colnames(ld.sd.gene.expression),t(ld.sd.gene.expression))
colnames(pca.gene.expression.ld.sd)[1] <- "Time point"

res.pca.ld.sd <- PCA(pca.gene.expression.ld.sd, graph = FALSE,scale.unit = TRUE,quali.sup = 1 )
res.hcpc.ld.sd <- HCPC(res.pca.ld.sd, graph=FALSE)
fviz_dend(res.hcpc.ld.sd,k=3,
```

```
          cex = 1,                          # Label size
          palette = "jco",                  # Color palette see ?ggpubr::ggpar
          rect = TRUE, rect_fill = TRUE,    # Add rectangle around groups
          rect_border = "jco",              # Rectangle color
          type="rectangle",
          labels_track_height = 400         # Augment the room for labels
)
```

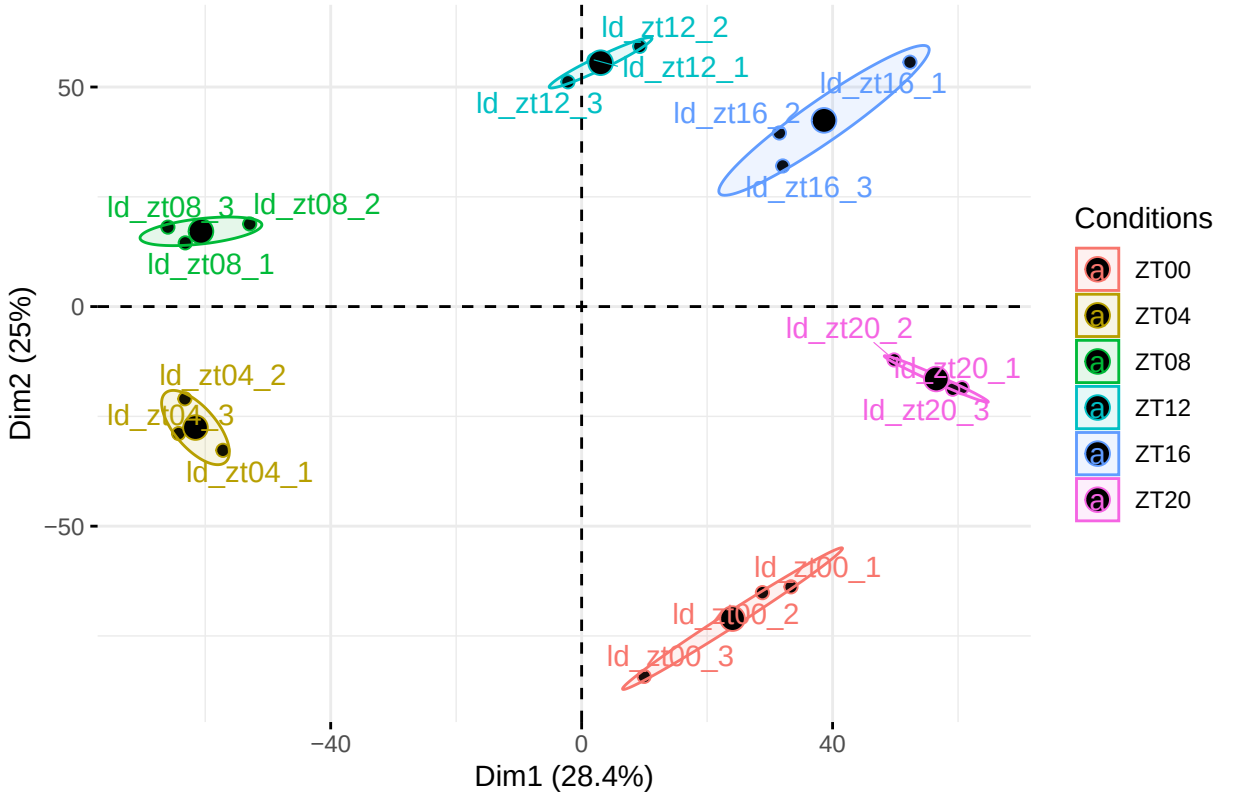


The transcriptomes corresponding to the same time during the three different days under both LD and SD conditions tend to cluster together. This indicates a high circadian synchronization in our cultures. Using hierarchical clustering, the 36 transcriptomes under LD and SD conditions assemble together into three different groups. The first cluster corresponds to **midday**. The transcriptomes at time points ZT4 and ZT8 under LD and ZT4 under SD constitute this cluster. These time points correspond to the moments of maximal incident light irradiance under both LD and SD conditions. The second cluster conforms the **dusk** group. Here the transcriptomes at time points ZT12 and ZT16 under LD and ZT8 under SD are found. These time points coincide with the end of the light period in both LD and SD conditions when light irradiance is low. The third cluster represents **night/dawn** and comprises the transcriptomes at time points ZT20, ZT0 under LD and ZT12, ZT16, ZT20 and ZT0 under SD. The transcriptomes at time points in the LD and SD nights or dark periods constitute two distinct groups suggesting noticeable differences in the transcriptomic responses during the night under LD and SD conditions. It is also noteworthy the higher similarity between the dusk, night/dawn transcriptomes when compare to the midday ones.

In order to obtain a more precise understanding of the underlying structure in our data we preform PCA separately for the LD and SD data.

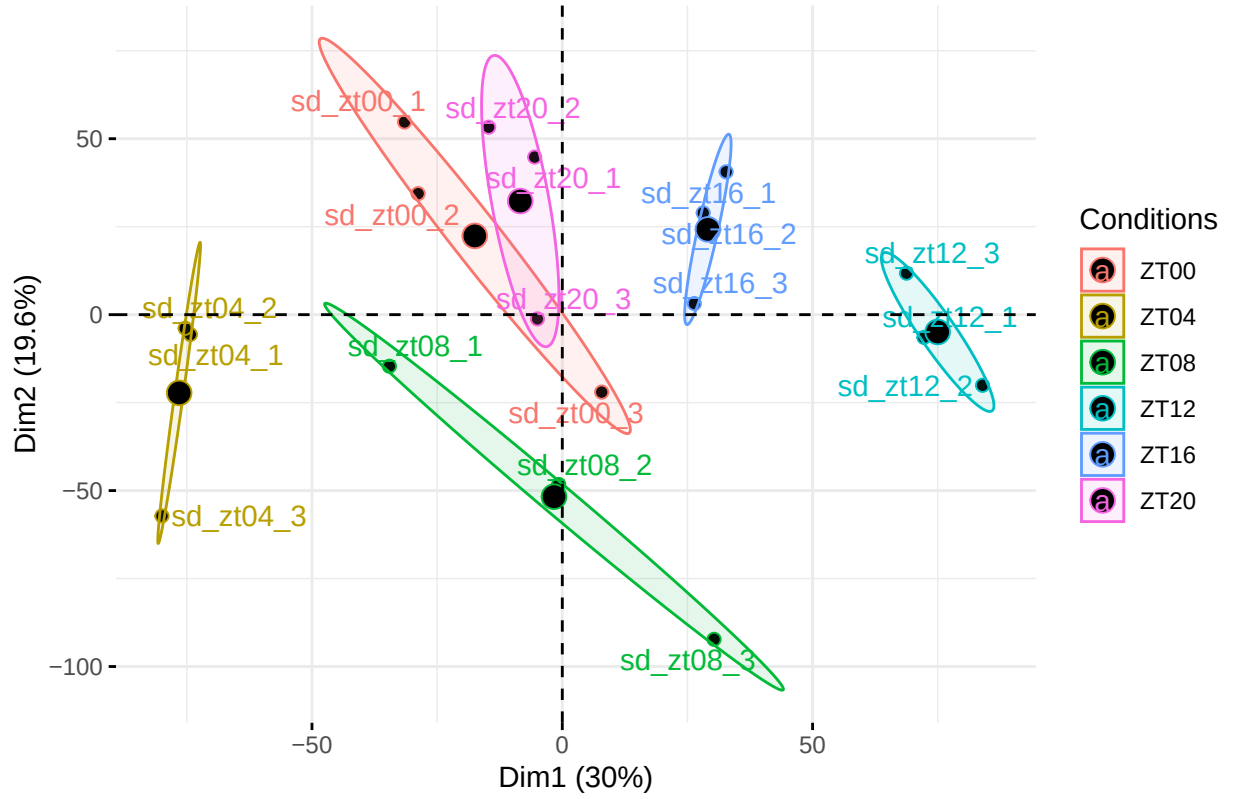
```
ld.gene.expression <- gene.expression[,ld.zt.i]
pca.gene.expression.ld <- data.frame(colnames(ld.gene.expression),t(ld.gene.expression))
colnames(pca.gene.expression.ld)[1] <- "Time point"
```

```
res.pca.ld <- PCA(pca.gene.expression.ld, graph = FALSE,scale.unit = TRUE,quali.sup = 1 )
fviz_pca_ind(res.pca.ld, col.ind = c("ZT00", "ZT00", "ZT00", "ZT04", "ZT04", "ZT04",
                                     "ZT08", "ZT08", "ZT08", "ZT12", "ZT12", "ZT12",
                                     "ZT16", "ZT16", "ZT16", "ZT20", "ZT20", "ZT20"),
             pointsize=2, pointshape=21,fill="black",
             repel = TRUE,
             addEllipses = TRUE,ellipse.type = "confidence",
             legend.title="Conditions",
             title="",
             show_legend=TRUE,show_guide=TRUE)
```



Under LD conditions, we observed that the transcriptomes corresponding to the same time point in the three different days tightly cluster together globally constituting a circular structure. Next, we analyze the SD data.

```
sd.gene.expression <- gene.expression[,sd.zt.i]
pca.gene.expression.sd <- data.frame(colnames(sd.gene.expression),t(sd.gene.expression))
colnames(pca.gene.expression.sd)[1] <- "Time point"

res.pca.sd <- PCA(pca.gene.expression.sd, graph = FALSE,scale.unit = TRUE,quali.sup = 1 )
fviz_pca_ind(res.pca.sd, col.ind = c("ZT00", "ZT00", "ZT00", "ZT04", "ZT04", "ZT04",
                                     "ZT08", "ZT08", "ZT08", "ZT12", "ZT12", "ZT12",
                                     "ZT16", "ZT16", "ZT16", "ZT20", "ZT20", "ZT20"),
             pointsize=2, pointshape=21,fill="black",
             repel = TRUE,
             addEllipses = TRUE,ellipse.type = "confidence",
             legend.title="Conditions",
             title="",
             show_legend=TRUE,show_guide=TRUE)
```



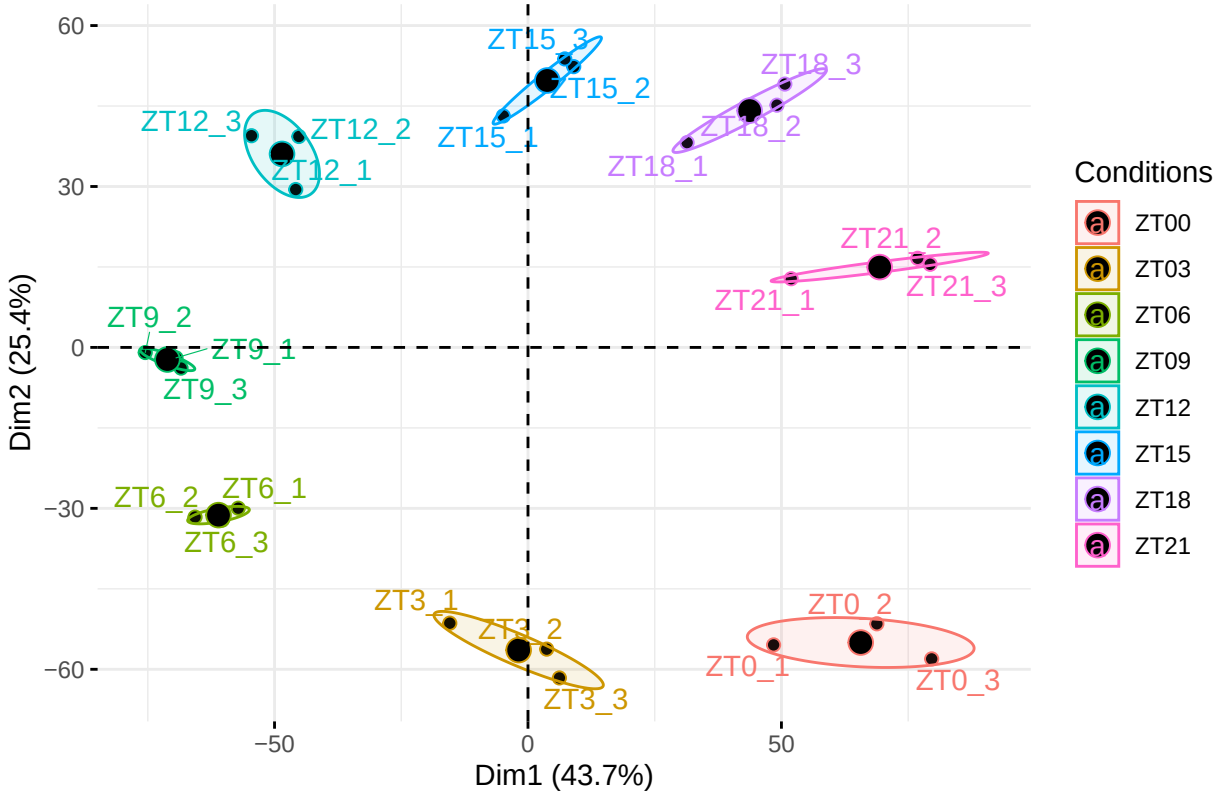
Under SD conditions more variability is observed and the time point transcriptomes form a structure resembling an ellipse. This could indicate that whereas in LD conditions gene expression in globally cycling precisely with a similar period a more complex behavior is expected under SD conditions. Also, it is remarkable the high similarity between the transcriptomes corresponding to ZT0 and ZT20 under SD conditions that is not present under LD conditions. This suggests that the transcriptomic response at the end of a SD night is already preparing all molecular systems for the incoming light availability at dawn whereas this anticipation is not as clearly observed under LD conditions. In overall, these results support that our experimental design grants a high level of synchronization in our data allowing us to proceed to the identification and comparison of genes exhibiting rhythmic expression patterns under LD and SD conditions.

```
nd.gene.expression <- as.matrix(read.table(file ="nd_gene_expression.tsv",header=T,sep="\t",as.is=T))
head(nd.gene.expression[,1:7])
```

```
##                  ZT0_1    ZT0_2    ZT0_3    ZT3_1    ZT3_2    ZT3_3    ZT6_1
## ostta01g00020 8.213914 8.020356 8.366347 7.955635 7.544027 7.155111 7.946078
## ostta01g00040 7.255037 6.829411 6.515928 7.474926 7.150623 7.026621 7.852350
## ostta01g00050 6.513748 5.564406 5.266403 7.327766 7.957293 7.823486 7.660194
## ostta01g00060 7.457919 7.567142 7.591113 6.803145 7.264232 7.600612 5.803992
## ostta01g00070 8.302257 8.270483 8.208690 7.444778 8.322443 8.346271 6.597210
## ostta01g00080 5.905527 6.739822 5.827514 7.114188 7.039538 6.416175 8.211618
```

```
pca.gene.expression.nd <- data.frame(colnames(nd.gene.expression),t(nd.gene.expression))
colnames(pca.gene.expression.nd)[1] <- "Time point"

res.pca.nd <- PCA(pca.gene.expression.nd, graph = FALSE,scale.unit = TRUE,quali.sup = 1 )
fviz_pca_ind(res.pca.nd, col.ind = c("ZT00", "ZT00", "ZT00", "ZT03", "ZT03", "ZT03",
                                     "ZT06", "ZT06", "ZT06", "ZT09", "ZT09", "ZT09",
                                     "ZT12", "ZT12", "ZT12", "ZT15", "ZT15", "ZT15",
```

```
                                        "ZT18", "ZT18", "ZT18", "ZT21", "ZT21", "ZT21"),
          pointsize=2, pointshape=21,fill="black",
          repel = TRUE,
          addEllipses = TRUE,ellipse.type = "confidence",
          legend.title="Conditions",
          title="",
          show_legend=TRUE,show_guide=TRUE)
```



Identification of genes exhibiting rhythmic expression patterns under different photoperiods

Using the bioconductor library **RAIN (Rhythmicity Analysis Incorporating Nonparametric methods)** we identified genes exhibiting periodic or rhythmic expression patterns under the different photoperiods. A p-value of 0.05 was used in all the analysis. Simple rhythmic expression patterns consisting of a single expression peak were identified by searching for patterns with a period of 24 hours. More complex cyclic gene expression patterns such as genes cycling twice or three times in a day were searched by fixing periods of 12 and 8 hours.

```
library(rain)
```

```
## Loading required package: gmp
```

```
##
## Attaching package: 'gmp'
```

```
## The following objects are masked from 'package:base':
##
##     %*%, apply, crossprod, matrix, tcrossprod
```

```
## Loading required package: multtest
```

```
## 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:gmp':
##
##     which.max, which.min
```

```
## 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
```

```
## 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")'.
```

```
results.ld <- rain(t(ld.gene.expression), deltat=4, period=24, verbose=FALSE, nr.series=3)
sum(results.ld$pVal < 0.05)/number.genes
```

```
## [1] 0.7921231
```

```
rhythmic.genes.ld <- rownames(subset(results.ld, pVal < 0.05))
length(rhythmic.genes.ld)
```

```
## [1] 6074
```

```
results.ld.12 <- rain(t(ld.gene.expression), deltat=4, period=12, verbose=FALSE, nr.series=3)
sum(results.ld.12$pVal < 0.05)/number.genes
```

```
## [1] 0.04903495
```

```
rhythmic.genes.ld.12 <- rownames(subset(results.ld.12, pVal < 0.05))
length(rhythmic.genes.ld.12)
```

```
## [1] 376
```

```
complete.ld.rhythmic.genes <- unique(c(rhythmic.genes.ld,rhythmic.genes.ld.12))
length(complete.ld.rhythmic.genes)/number.genes
```

```
## [1] 0.8086854
```

```
non.rhythmic.ld.genes <- setdiff(ostta.genes,complete.ld.rhythmic.genes)
length(non.rhythmic.ld.genes)
```

```
## [1] 1467
```

```
length(non.rhythmic.ld.genes)/number.genes
```

```
## [1] 0.1913146
```

```
length(setdiff(rhythmic.genes.ld,rhythmic.genes.ld.12))
```

```
## [1] 5825
```

```
length(setdiff(rhythmic.genes.ld,rhythmic.genes.ld.12))/number.genes
```

```
## [1] 0.7596505
```

```
results.ld.8 <- rain(t(ld.gene.expression), deltat=4, period=8, verbose=FALSE, nr.series=3)
sum(results.ld.8$pVal < 0.05)/number.genes
```

```
## [1] 0.01056338
```

Under LD conditions we found that 6201 genes constituting 80.87% of the *Ostreococcus tauri* genome exhibit diurnal cycling expression patterns. Most of these genes, 5825 genes covering 75.97% of the entire genome presented a simple rhythmic expression pattern with a 24 hours period and a single expression peak. Only 376 genes presented a more complex rhythmic expression pattern cycling twice in a day presenting two expression peaks with an apparent period of 12 hours. The number of genes with expression patterns cycling three times in a day with an apparent period of 8 hours was negligible, 81 genes representing less than 1% of the genome. We also identified 1467 genes comprising 19.13% of genome that did not present a detectable rhythmic expression pattern.

```
results.nd <- rain(t(nd.gene.expression), deltat=3, period=24, verbose=FALSE, nr.series=3)
sum(results.nd$pVal < 0.05)/number.genes
```

```
## [1] 0.8035994
```

```
rhythmic.genes.nd <- intersect(rownames(subset(results.nd, pVal < 0.05)),ostta.genes)
length(rhythmic.genes.nd)
```

```
## [1] 6162
```

```
results.nd.12 <- rain(t(nd.gene.expression), deltat=3, period=12, verbose=FALSE, nr.series=3)
sum(results.nd.12$pVal < 0.05)/number.genes
```

```
## [1] 0.1527126
```

```
rhythmic.genes.nd.12 <- rownames(subset(results.nd.12, pVal < 0.05))
length(rhythmic.genes.nd.12)
```

```
## [1] 1171
```

```
complete.nd.rhythmic.genes <- unique(c(rhythmic.genes.nd,rhythmic.genes.nd.12))
length(complete.nd.rhythmic.genes)
```

```
## [1] 6372
```

```
length(complete.nd.rhythmic.genes)/number.genes
```

```
## [1] 0.8309859
```

```
non.rhythmic.nd.genes <- setdiff(ostta.genes,complete.nd.rhythmic.genes)
length(non.rhythmic.nd.genes)
```

```
## [1] 1296
```

```
length(setdiff(rhythmic.genes.nd,rhythmic.genes.nd.12))
```

```
## [1] 5201
```

```
length(setdiff(rhythmic.genes.nd,rhythmic.genes.nd.12))/number.genes
```

```
## [1] 0.6782733
```

```
results.nd.8 <- rain(t(nd.gene.expression), deltat=4, period=8, verbose=FALSE, nr.series=3)
sum(results.nd.8$pVal < 0.05)/number.genes ## 0.8%
```

```
## [1] 0.007955138
```

Under ND conditions we found that 6372 genes constituting 83.1% of the *Ostreococcus tauri* genome exhibit diurnal cycling expression patterns. Most of these genes, 5201 genes covering 67.82%% of the entire genome presented a simple rhythmic expression pattern with a 24 hours period and a single expression peak. An increasing number of genes, namely 1171 genes, presented a more complex rhythmic expression pattern cycling twice in a day presenting two expression peaks with an apparent period of 12 hours. The number of genes with expression patterns cycling three times in a day with an apparent period of 8 hours was negligible, 61 genes representing less than 1% of the genome. We also identified 1274 genes comprising 16.61% of genome that did not present a detectable rhythmic expression pattern.

```
results.sd <- rain(t(sd.gene.expression), deltat=4, period=24, verbose=FALSE, nr.series=3)
sum(results.sd$pVal < 0.05)/number.genes
```

```
## [1] 0.7037037
```

```
rhythmic.genes.sd <- rownames(subset(results.sd, pVal < 0.05))
length(rhythmic.genes.sd)
```

```
## [1] 5396
```

```
results.sd.12 <- rain(t(sd.gene.expression), deltat=4, period=12, verbose=FALSE, nr.series=3)
sum(results.sd.12$pVal < 0.05)/number.genes
```

```
## [1] 0.2419144
```

```
rhythmic.genes.sd.12 <- rownames(subset(results.sd.12, pVal < 0.05))
length(rhythmic.genes.sd.12)
```

```
## [1] 1855
```

```
complete.sd.rhythmic.genes <- unique(c(rhythmic.genes.sd,rhythmic.genes.sd.12))
length(complete.sd.rhythmic.genes)
```

```
## [1] 6104
```

```
length(complete.sd.rhythmic.genes)/number.genes
```

```
## [1] 0.7960355
```

```
non.rhythmic.sd.genes <- setdiff(ostta.genes,complete.sd.rhythmic.genes)
length(non.rhythmic.sd.genes)
```

```
## [1] 1564
```

```
length(non.rhythmic.sd.genes)/number.genes
```

```
## [1] 0.2039645
```

```
results.sd.8 <- rain(t(sd.gene.expression), deltat=4, period=8, verbose=FALSE, nr.series=3)
sum(results.sd.8$pVal < 0.05)
```

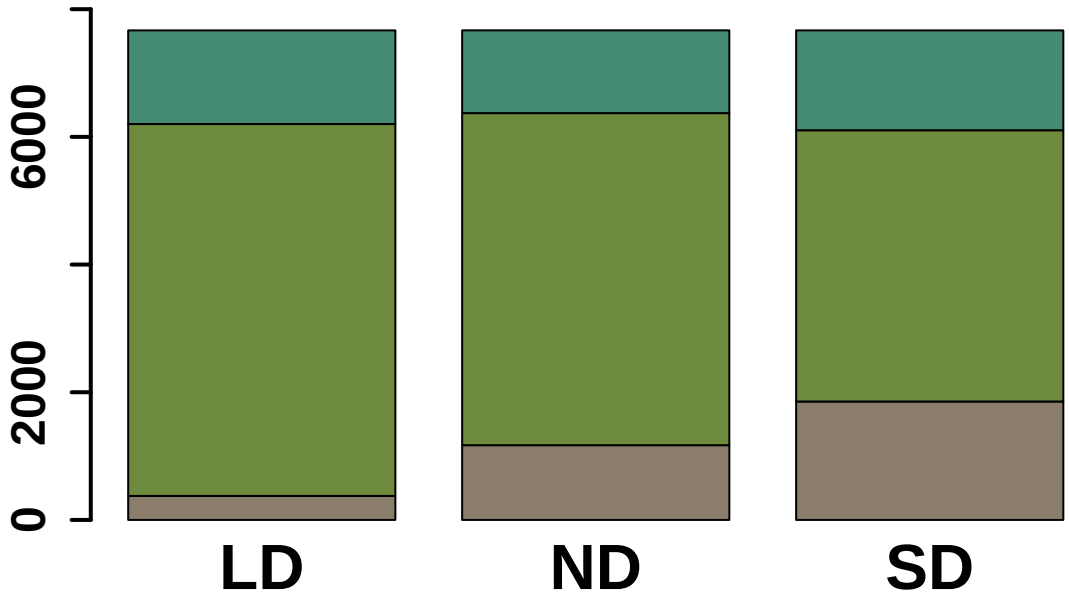
```
## [1] 23
```

```
sum(results.sd.8$pVal < 0.05)/number.genes
```

```
## [1] 0.002999478
```

Under SD conditions we found that 6104 genes constituting 79.6% of the *Ostreococcus tauri* genome exhibit diurnal cycling expression patterns. A substantially reduced number of these genes, 4249 genes covering 55.41%% of the entire genome presented a simple rhythmic expression pattern with a 24 hours period and a single expression peak. An increasing number of genes, namely 1855 genes, presented a more complex rhythmic expression pattern cycling twice in a day presenting two expression peaks with an apparent period of 12 hours. The number of genes with expression patterns cycling three times in a day with an apparent period of 8 hours was negligible, 23 genes representing less than 1% of the genome. We also identified 1564 genes comprising 20.41% of genome that did not present a detectable rhythmic expression pattern.

```
data <- matrix(c(length(rhythmic.genes.ld.12),length(setdiff(rhythmic.genes.ld,rhythmic.genes.ld.12)),l
length(rhythmic.genes.nd.12),length(setdiff(rhythmic.genes.nd,rhythmic.genes.nd.12)),length(non.rhythmi
length(rhythmic.genes.sd.12),length(setdiff(rhythmic.genes.sd,rhythmic.genes.sd.12)),length(non.rhythmi

barplot(data, lwd=2, names.arg = c("LD","ND","SD"),cex.names = 2,cex.axis = 1.5,
        col=colors()[c(23,89,12)],
        border="black",
        space=0.25,
        font.axis=2,
        xlab="",ylim=c(0,8000))
```



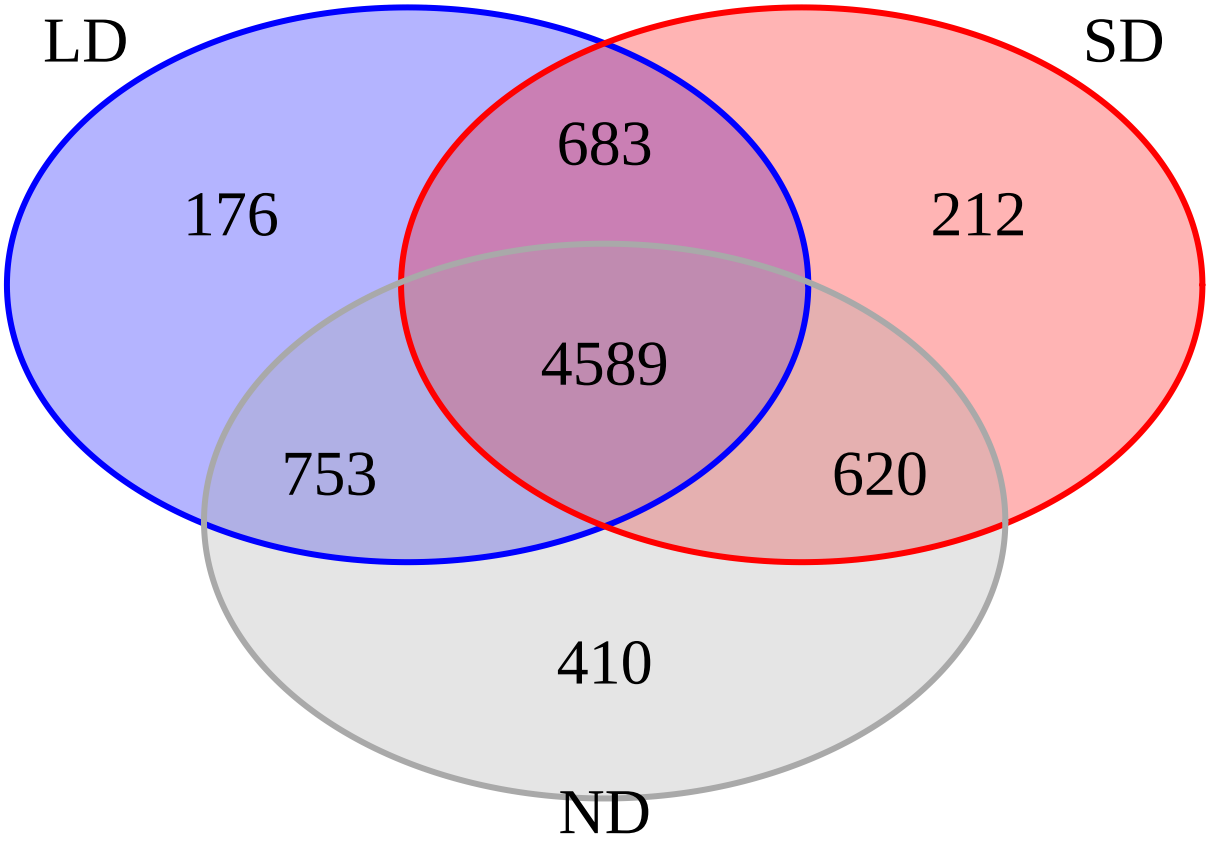
We observed that practically the same number of genes are identified as rhythmic under the three different conditions. Independently from the length of the photoperiod we identified approximately 80% of the genome of *Ostreococcus tauri* as rhythmically expressed. Remarkably, the number of rhythmic genes exhibiting two peaks during a day increases as the photoperiod shortens whereas the number of genes with a single peak in a day decreases. Furthermore, the number of genes that did not exhibit a diurnal rhythmic pattern was similar under the three different photoperiods.

```
library(VennDiagram)
```

```
## Loading required package: grid
```

```
## Loading required package: futile.logger
```

```
grid.newpage()
draw.triple.venn(area1 = length(complete.ld.rhythmic.genes),area2 = length(complete.sd.rhythmic.genes),a
```




```
## (polygon[GRID.polygon.616], polygon[GRID.polygon.617], polygon[GRID.polygon.618], polygon[GRID.polygo
```

```
length(intersect(intersect(complete.sd.rhythmic.genes, complete.ld.rhythmic.genes),complete.nd.rhythmic
```

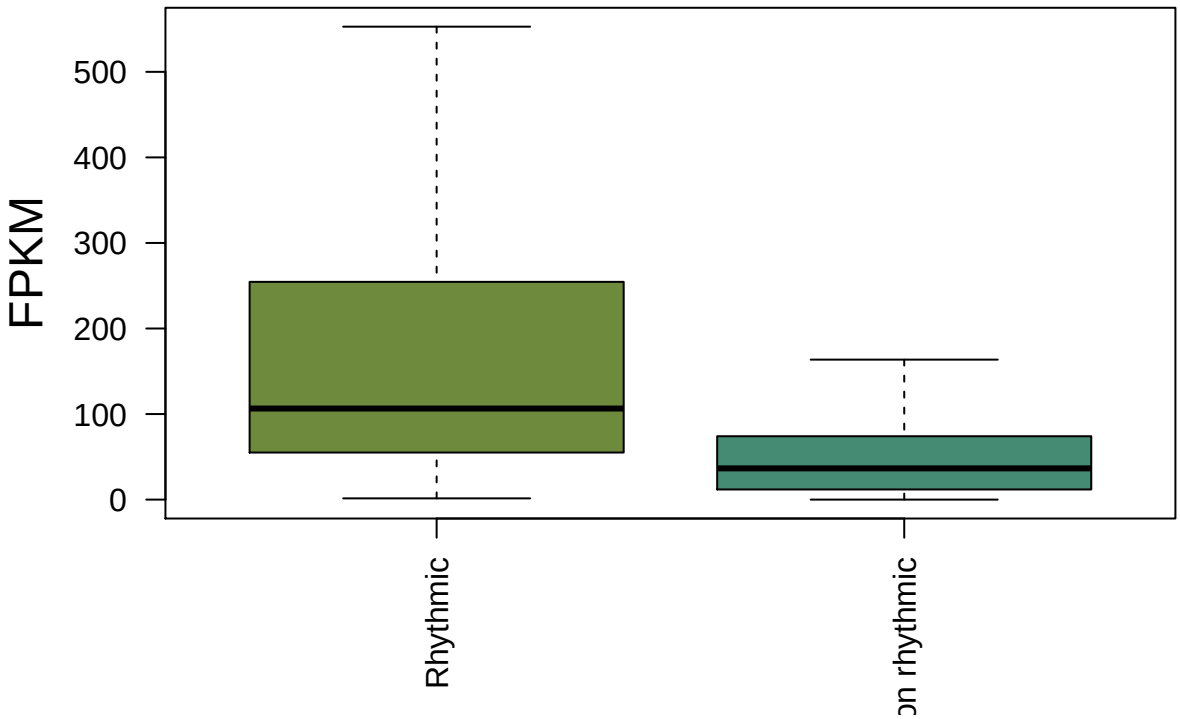
```
## [1] 4589
```

```
length(intersect(intersect(complete.sd.rhythmic.genes, complete.ld.rhythmic.genes),complete.nd.rhythmic
```

```
## [1] 0.5984611
```

When comparing rhythmically expressed genes under LD, ND and SD conditions we found that 4589 genes comprising 59.84% of the genome exhibit rhythmic expression patterns. Whereas only a few hundreds genes only presented a cycling expression pattern under a specific photoperiod. In order to further explore the differences between rhythmic a non rhythmic genes we compared their expression level under LD and SD conditions. We restrict our analysis to these two conditions since here expression levels are estimated using the same technology namely RNA-seq whereas for ND microarrays were used.

```
rhythmic.max.expression <- apply(X = gene.expression[intersect(complete.ld.rhythmic.genes,complete.sd.rh
non.rhythmic.max.expression <- apply(X = gene.expression[intersect(non.rhythmic.ld.genes,non.rhythmic.sd
boxplot(rhythmic.max.expression, non.rhythmic.max.expression, outline=F,col=colors()[c(89,12)],names=c(
```




```
wilcox.test(rhythmic.max.expression,3*non.rhythmic.max.expression,alternative="greater")
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  rhythmic.max.expression and 3 * non.rhythmic.max.expression
## W = 1821003, p-value = 0.0001448
## alternative hypothesis: true location shift is greater than 0
```

We found that genes exhibiting rhythmic expression patterns under both LD and SD conditions present a maximal level of expression three times greater than genes identified as non rhythmic. This difference was significant according to p-value of 1.45e-4 computed using Mann-Whitney-Wilcoxson test. The current methods for detecting rhythmic gene expression are known to only perform optimally for highly expressed genes. Therefore, the genes identified as non rhythmic in this study could indeed be rhythmic although their low expression level have prevented our methods to detect them.

Genes maintaining their rhythmic expression pattern in continuous light regimes (LL conditions) independently from the alternating light and dark periods are considered here bona fide circadian genes. Our experimental design includes two days under continuous light representing free running conditions after the three days under either LD or SD conditions. We use RAIN to identify circadian genes. In order to avoid a bias towards days under LD or days under continuous light we selected for this analysis two days under LD followed by two days under continuous light.

```
ld.zt <- paste("ld",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")
ld.ll.zt.i <- sapply(X = ld.zt,FUN = function(x){ paste(x,2:5,sep="_")})

ld.ll.gene.expression <- gene.expression[,ld.ll.zt.i]
dim(ld.ll.gene.expression)
```

```
## [1] 7668   24
```

```
results.ld.ll <- rain(t(ld.ll.gene.expression), deltat=4, period=24, verbose=FALSE, nr.series=4)
head(results.ld.ll)
```

```
##                       pVal phase peak.shape period
## ostta01g00010 6.532494e-05    20          8     24
## ostta01g00020 4.435461e-04    20         16     24
## ostta01g00030 7.720299e-01    16         16     24
## ostta01g00040 4.568522e-01    20         12     24
## ostta01g00050 1.241564e-02     4          8     24
## ostta01g00060 6.732110e-04    24         12     24
```

```
sum(results.ld.ll$pVal < 0.05)/number.genes
```

```
## [1] 0.6888367
```

```
rhythmic.genes.ld.ll <- rownames(subset(results.ld.ll, pVal < 0.05))
length(rhythmic.genes.ld.ll)
```

```
## [1] 5282
```

```
rhythmic.genes.ld.ll <- intersect(complete.ld.rhythmic.genes,rhythmic.genes.ld.ll)
length(rhythmic.genes.ld.ll)/number.genes
```

```
## [1] 0.6580595
```

```
results.ld.ll.12 <- rain(t(ld.ll.gene.expression), deltat=4, period=12, verbose=FALSE, nr.series=4)
sum(results.ld.ll.12$pVal < 0.05)/number.genes
```

```
## [1] 0.03482003
```

```
rhythmic.genes.ld.ll.12 <- rownames(subset(results.ld.ll.12, pVal < 0.05))
length(rhythmic.genes.ld.ll.12)
```

```
## [1] 267
```

```
rhythmic.genes.ld.ll.12 <- intersect(rhythmic.genes.ld.ll.12,complete.ld.rhythmic.genes)
length(rhythmic.genes.ld.ll.12)
```

```
## [1] 233
```

```
complete.ld.ll.rhythmic.genes <- unique(c(rhythmic.genes.ld.ll,rhythmic.genes.ld.ll.12))

length(complete.ld.ll.rhythmic.genes)
```

```
## [1] 5129
```

```
length(setdiff(complete.ld.rhythmic.genes, complete.ld.ll.rhythmic.genes))
```

```
## [1] 1072
```

```
length(complete.ld.ll.rhythmic.genes)/number.genes
```

```
## [1] 0.6688837
```

```
length(complete.ld.ll.rhythmic.genes)/length(complete.ld.rhythmic.genes)
```

```
## [1] 0.8271247
```

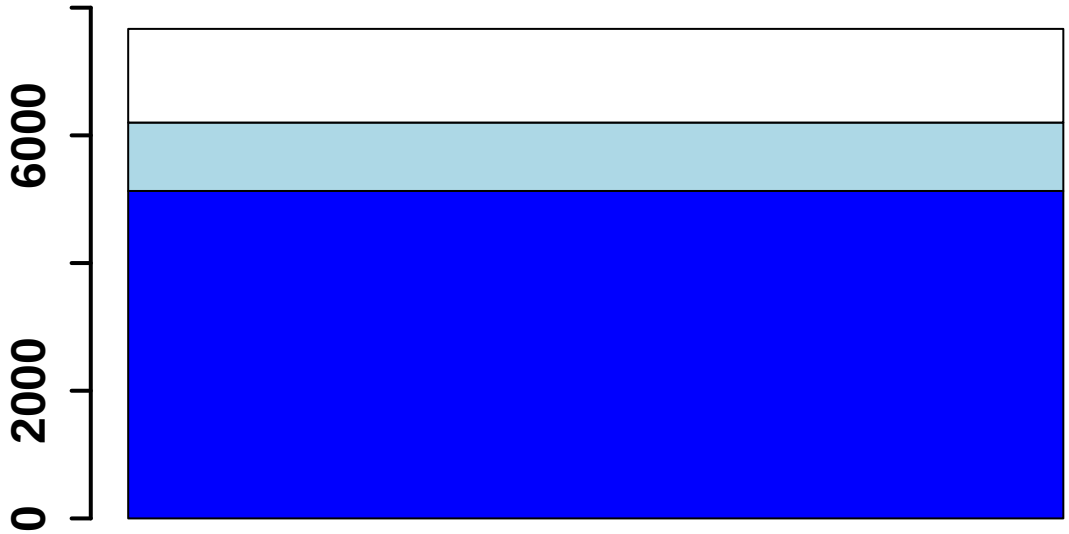
```
length(setdiff(complete.ld.rhythmic.genes,complete.ld.ll.rhythmic.genes))
```

```
## [1] 1072
```

```
data.ld.ll <- matrix(c(length(complete.ld.ll.rhythmic.genes),length(setdiff(complete.ld.rhythmic.genes,c

sum(c(length(complete.ld.ll.rhythmic.genes),length(setdiff(complete.ld.rhythmic.genes,complete.ld.ll.rh
```

```
## [1] 7668
```

```
barplot(data.ld.ll, lwd=2, names.arg = "",cex.names = 2,cex.axis = 1.5,
        col=c("blue","lightblue","white"),
        border="black",
        font.axis=2,
        xlab="",ylim=c(0,8000))
```



We found that 82.71% of the genes identified previously as rhythmic under LD conditions still presented cycling expression patterns when transferred to free running conditions. Specifically, we identified 5129 circadian genes under LD conditions comprising 66.89% of the entire genome. The 1072 remaining genes cycling under LD conditions presented either a flat or noisy expression profile under LL conditions.

```
sd.zt <- paste("sd",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")
sd.ll.zt.i <- sapply(X = sd.zt,FUN = function(x){ paste(x,2:5,sep="_")})

sd.ll.gene.expression <- gene.expression[,sd.ll.zt.i]

results.sd.ll <- rain(t(sd.ll.gene.expression), deltat=4, period=24, verbose=FALSE, nr.series=4)
head(results.sd.ll)
```

```
##                      pVal phase peak.shape period
## ostta01g00010 0.22624909    12          8     24
## ostta01g00020 0.59259899    16          4     24
## ostta01g00030 0.92106854    12         20     24
## ostta01g00040 0.87754181     4          4     24
## ostta01g00050 0.01031291    20         12     24
## ostta01g00060 0.25784227    16         16     24
```

```
sum(results.sd.ll$pVal < 0.05)/number.genes ## 31.53%
```

```
## [1] 0.3153365
```

```
rhythmic.genes.sd.ll <- rownames(subset(results.sd.ll, pVal < 0.05))
rhythmic.genes.sd.ll <- intersect(rhythmic.genes.sd.ll, complete.sd.rhythmic.genes)
length(rhythmic.genes.sd.ll)
```

```
## [1] 2303
```

```
results.sd.ll.12 <- rain(t(sd.ll.gene.expression), deltat=4, period=12, verbose=FALSE, nr.series=4)
sum(results.sd.ll.12$pVal < 0.05)/number.genes ## 11.2%
```

```
## [1] 0.112024
```

```
rhythmic.genes.sd.ll.12 <- rownames(subset(results.sd.ll.12, pVal < 0.05))
rhythmic.genes.sd.ll.12 <- intersect(rhythmic.genes.sd.ll.12, complete.sd.rhythmic.genes)
length(rhythmic.genes.sd.ll.12)
```

```
## [1] 761
```

```
length(intersect(rhythmic.genes.sd.ll.12,rhythmic.genes.sd.12))
```

```
## [1] 618
```

```
length(intersect(rhythmic.genes.sd.ll.12,rhythmic.genes.sd.12)) / length(rhythmic.genes.sd.12) #33.32%
```

```
## [1] 0.3331536
```

```
complete.sd.ll.rhythmic.genes <- unique(c(rhythmic.genes.sd.ll,rhythmic.genes.sd.ll.12))
length(complete.sd.ll.rhythmic.genes)
```

```
## [1] 2874
```

```
length(complete.sd.ll.rhythmic.genes)/length(complete.sd.rhythmic.genes)
```

```
## [1] 0.4708388
```

```
length(complete.sd.ll.rhythmic.genes)/number.genes
```

```
## [1] 0.3748044
```

```
length(setdiff(complete.sd.rhythmic.genes,complete.sd.ll.rhythmic.genes))
```

```
## [1] 3230
```

```
data.sd.ll <- matrix(c(length(complete.sd.ll.rhythmic.genes),length(setdiff(complete.sd.rhythmic.genes,
sum(c(length(complete.sd.ll.rhythmic.genes),length(setdiff(complete.sd.rhythmic.genes,complete.sd.ll.rh
```

```
## [1] 7668
```

```
barplot(data.sd.ll, lwd=2, names.arg = "",cex.names = 2,cex.axis = 1.5,
        col=c("red","lightpink","white"),
        border="black",
        font.axis=2,
        xlab="",ylim=c(0,8000))
```

```
length(intersect(complete.ld.ll.rhythmic.genes,complete.sd.ll.rhythmic.genes))
```

```
## [1] 2354
```

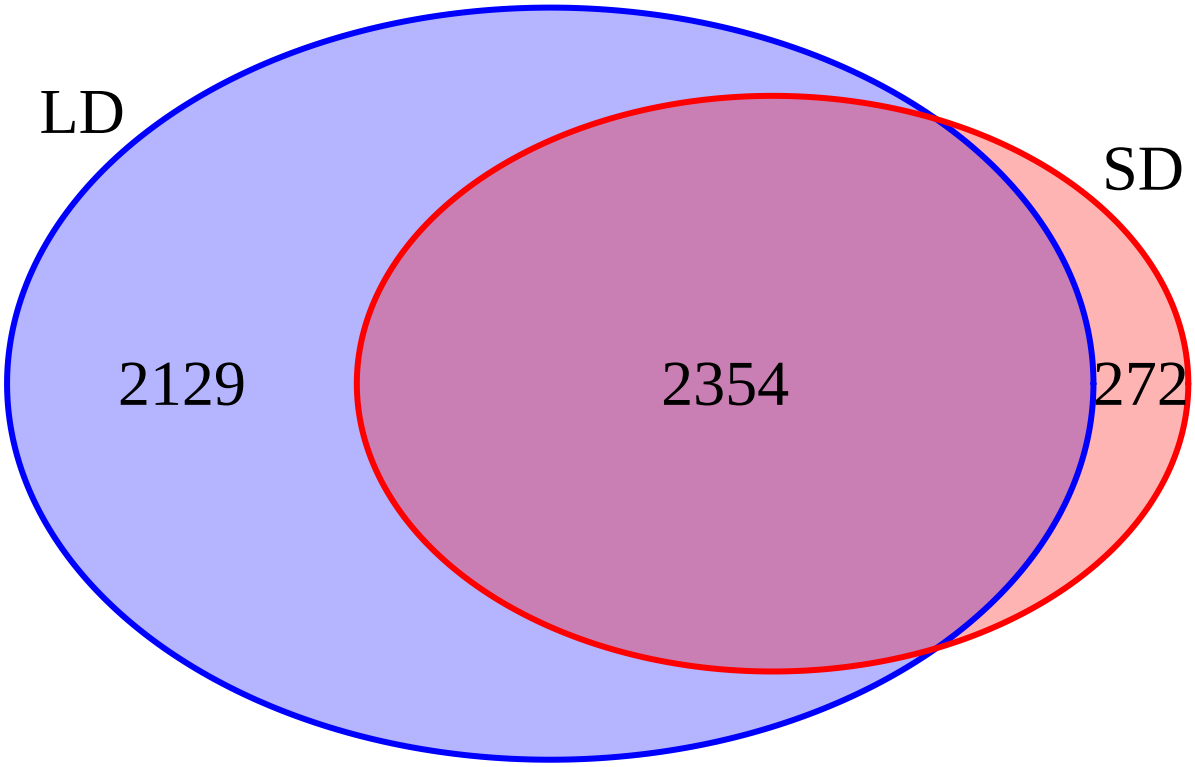
```
length(setdiff(complete.ld.ll.rhythmic.genes,complete.sd.ll.rhythmic.genes))
```

```
## [1] 2775
```

```
length(setdiff(complete.sd.ll.rhythmic.genes,complete.ld.ll.rhythmic.genes))
```

```
## [1] 520
```

```
complete.ld.sd.rhythmic.genes <- intersect(complete.ld.rhythmic.genes,complete.sd.rhythmic.genes)

grid.newpage()
draw.pairwise.venn(area1 = length(intersect(complete.ld.sd.rhythmic.genes,complete.ld.ll.rhythmic.genes)
```




```
## (polygon[GRID.polygon.632], polygon[GRID.polygon.633], polygon[GRID.polygon.634], polygon[GRID.polyg
```

```
length(intersect(intersect(complete.ld.rhythmic.genes,complete.sd.rhythmic.genes),setdiff(complete.sd.ll
```

```
## [1] 272
```

```
length(intersect(intersect(complete.ld.rhythmic.genes,complete.sd.rhythmic.genes),setdiff(complete.ld.ll
```

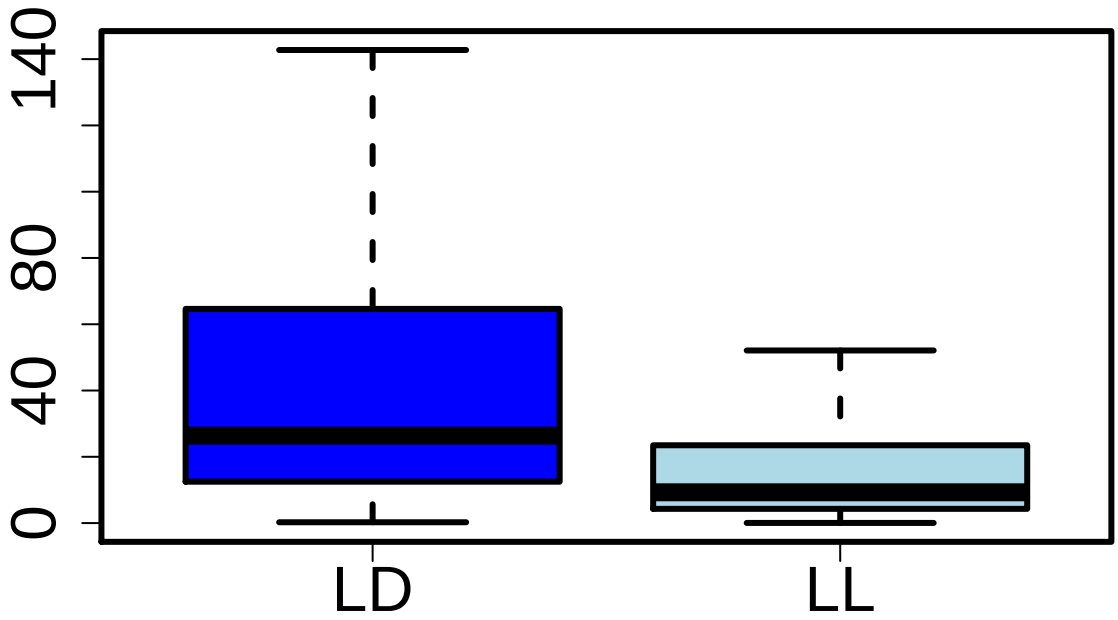
```
## [1] 2129
```

Unlike the analysis for LD conditions, we found that only 47.1% of the genes identified previously as rhythmic under SD conditions still presented cycling expression patterns when transferred to free running conditions. Specifically, we only identified 2874 circadian genes under SD conditions comprising 37.48% of the entire genome. When comparing circadian genes under LD and SD conditions we found that most circadian genes in SD conditions are also circadian under LD conditions. Nonetheless, 2129 circadian genes were specific under LD conditions. This suggests that rhythmic expression under SD conditions is very dependent on the

presence of a long period of dark whereas rhythmic expression under LD conditions due to the presence of a short period of dark is better maintained under continuous light.

In order to explore the effect of the transition to continuous light on rhythmic gene expression patterns we compared three rhythmic characteristics of the expression profiles under LD or SD to the corresponding ones under LL. Specifically we compared amplitude or half of the difference between the peak and trough of the expression profile; mesor or mean expression level around which the expression profile oscillates and phase or time at which the expression profile peaks. This was carried out using the R package circacompare that performs a fitting to the data as cosinusoidal curve with a particular parametrization that is used to test for statistical significant differences between the two circadian patterns under comparison.

```
library(circacompare)
ld.zt <- paste("ld",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")

circacompare.ld.ll <- matrix(nrow=length(complete.ld.ll.rhythmic.genes),ncol=15)
rownames(circacompare.ld.ll) <- complete.ld.ll.rhythmic.genes

for(i in 1:length(complete.ld.ll.rhythmic.genes))
{
  gene.i <- complete.ld.ll.rhythmic.genes[i]

  ld.expression.i <- ld.ll.gene.expression[gene.i,c(paste(ld.zt,2,sep="_"),paste(ld.zt,3,sep="_"))]
  ll.expression.i <- ld.ll.gene.expression[gene.i,c(paste(ld.zt,4,sep="_"),paste(ld.zt,5,sep="_"))]

  time.points <- seq(from=0,by=4,length.out = 12)

  ld.ll.df <- data.frame(time=c(time.points,time.points),
                         measure=c(ld.expression.i, ll.expression.i),
                         group=c(rep("ld",12),rep("ll",12)))

  out.i <- circacompare(x = ld.ll.df, col_time = "time", col_group = "group", col_outcome = "measure",al
  circacompare.ld.ll[i,] <- out.i[[2]][,2]
 }

colnames(circacompare.ld.ll) <- out.i[[2]][,1]

par(lwd=3)
boxplot(circacompare.ld.ll[,"ld amplitude estimate" ],circacompare.ld.ll[,"ll amplitude estimate" ],outl
```




```
sum(circacompare.ld.ll[,"Amplitude difference estimate" ] < 0)
```

```
## [1] 5010
```

```
sum(circacompare.ld.ll[,"Amplitude difference estimate" ] < 0) / length(complete.ld.ll.rhythmic.genes)
```

```
## [1] 0.9767986
```

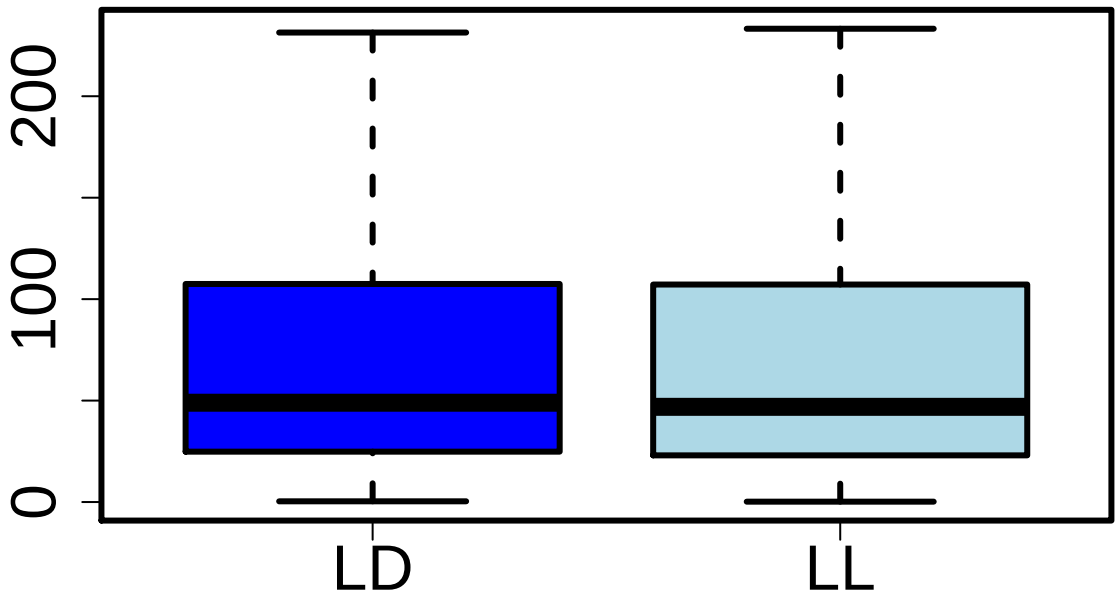
```
sum(circacompare.ld.ll[,"P-value for amplitude difference" ] < 0.05)
```

```
## [1] 2791
```

```
sum(circacompare.ld.ll[,"P-value for amplitude difference" ] < 0.05)/length(complete.ld.ll.rhythmic.gene
```

```
## [1] 0.5441607
```

```
boxplot(circacompare.ld.ll[,"ld mesor estimate" ],circacompare.ld.ll[,"ll mesor estimate" ],outline=F,c
```



```
sum(circacompare.ld.ll[,"Mesor difference estimate" ] < 0) /length(complete.ld.ll.rhythmic.genes)
```

```
## [1] 0.5381166
```

```
sum(circacompare.ld.ll[,"Mesor difference estimate" ] > 0) /length(complete.ld.ll.rhythmic.genes)
```

```
## [1] 0.4618834
```

```
sum(circacompare.ld.ll[,"P-value for mesor difference" ] < 0.05)/length(complete.ld.ll.rhythmic.genes)
```

```
## [1] 0.2288945
```

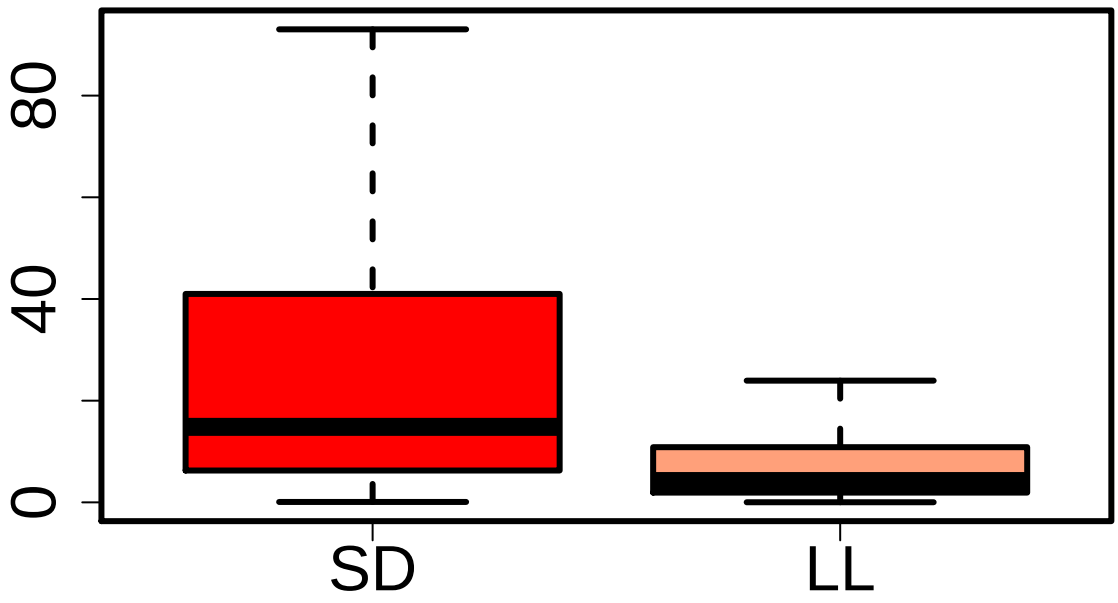
```
sum(circacompare.ld.ll[,"P-value for difference in phase" ] < 0.05)
```

```
## [1] 460
```

Most circadian genes under LD conditions, 97.68%, presented a sharp decrease in amplitude when transferred to LL, being significant in more than half of them with p-value < 0.05. Nevertheless, no clear effect was observed over the mesor or phase of gene expression profiles that remained in LL similar to the values in LD.

Next we perform the analysis for circadian genes under SD conditions. It is important to notice that the results of circacompare sorts the samples in alphabetical order. Specifically, the LL conditions is first and SD condition is second. Therefore, to check for a decrease in amplitude in LL with respect to SD we have to check for positive values.

```
sd.zt <- paste("sd",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")

circacompare.sd.ll <- matrix(nrow=length(complete.sd.ll.rhythmic.genes),ncol=15)
rownames(circacompare.sd.ll) <- complete.sd.ll.rhythmic.genes

for(i in 1:length(complete.sd.ll.rhythmic.genes))
{
  gene.i <- complete.sd.ll.rhythmic.genes[i]

  sd.expression.i <- sd.ll.gene.expression[gene.i,c(paste(sd.zt,2,sep="_"),paste(sd.zt,3,sep="_"))]
  ll.expression.i <- sd.ll.gene.expression[gene.i,c(paste(sd.zt,4,sep="_"),paste(sd.zt,5,sep="_"))]

  time.points <- seq(from=0,by=4,length.out = 12)

  sd.ll.df <- data.frame(time=c(time.points,time.points),
                         measure=c(sd.expression.i, ll.expression.i),
                         group=c(rep("sd",12),rep("ll",12)))

  out.i <- circacompare(x = sd.ll.df, col_time = "time", col_group = "group", col_outcome = "measure",al
  circacompare.sd.ll[i,] <- out.i[[2]][,2]
}

colnames(circacompare.sd.ll) <- out.i[[2]][,1]

par(lwd=3)
boxplot(circacompare.sd.ll[,"sd amplitude estimate" ],circacompare.sd.ll[,"ll amplitude estimate" ],outl
```




```
sum(circacompare.sd.ll[,"Amplitude difference estimate" ] > 0)
```

```
## [1] 2676
```

```
sum(circacompare.sd.ll[,"Amplitude difference estimate" ] > 0) / length(complete.sd.ll.rhythmic.genes)
```

```
## [1] 0.9311065
```

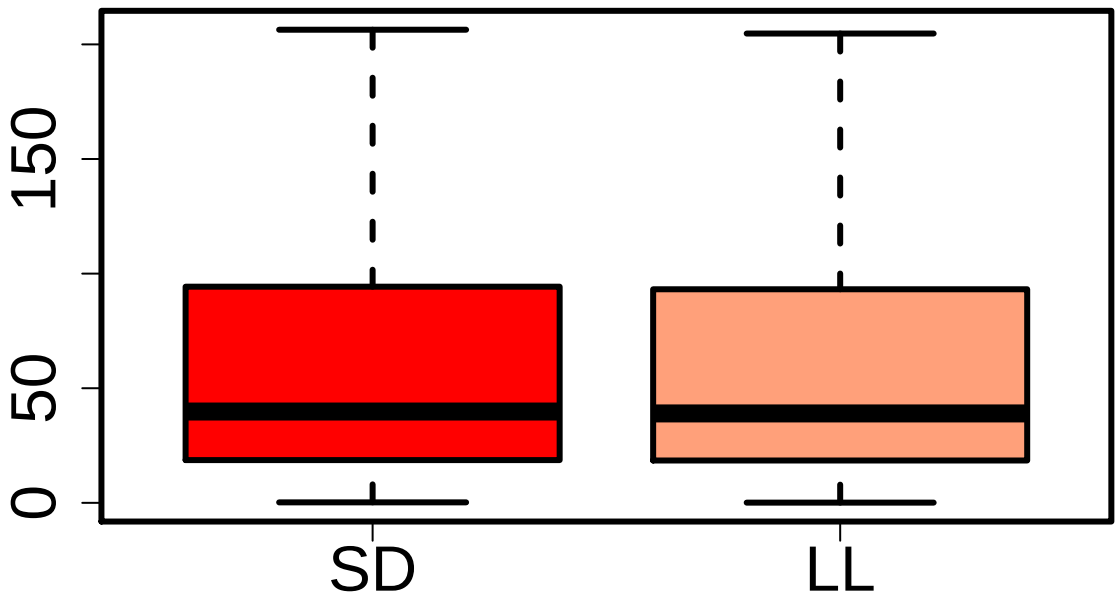
```
sum(circacompare.sd.ll[,"P-value for amplitude difference" ] < 0.05)
```

```
## [1] 1372
```

```
sum(circacompare.sd.ll[,"P-value for amplitude difference" ] < 0.05)/length(complete.sd.ll.rhythmic.gene
```

```
## [1] 0.4773834
```

```
boxplot(circacompare.sd.ll[,"sd mesor estimate" ],circacompare.sd.ll[,"ll mesor estimate" ],outline=F,c
```



```
sum(circacompare.sd.ll[,"Mesor difference estimate" ] < 0) /length(complete.sd.ll.rhythmic.genes)
```

```
## [1] 0.5038274
```

```
sum(circacompare.sd.ll[,"Mesor difference estimate" ] > 0) /length(complete.sd.ll.rhythmic.genes)
```

```
## [1] 0.4961726
```

```
sum(circacompare.sd.ll[,"P-value for mesor difference" ] < 0.05)/length(complete.sd.ll.rhythmic.genes)
```

```
## [1] 0.3204593
```

```
sum(circacompare.sd.ll[,"P-value for difference in phase" ] < 0.05)
```

```
## [1] 392
```

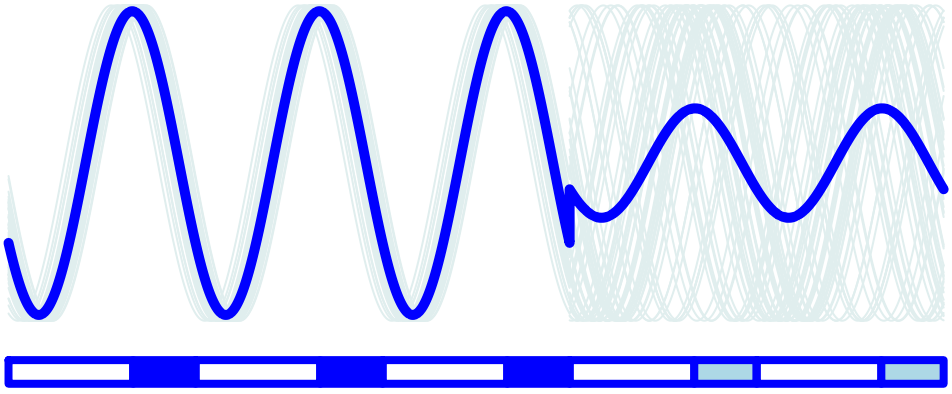
Similar to the results under LD conditons, we observed that most circadian genes under SD conditions, 93.11%, presented a sharp decrease in amplitude when transferred to LL, being significant in almost half of them with p-value < 0.05. Nevertheless, no clear effect was observed over the mesor or phase of gene expression profiles that remained in LL similar to the values in SD.

The observed drastic reduction in amplitude together with the conservation in mesor and phase of the rhythmic gene expression profiles when transferred from LD or SD to LL conditions could be explained by two different scenarios. Under LD and SD conditions the entire cell culture is highly synchronous with most cells presenting coordinated rhythmic gene expression profiles. The transition to continuous light could produce the cell culture to become asynchronous. Individual cells would still be presenting an almost completely rhythmic transcriptome with a conservation in amplitude and mesor but out of phase with respect to the other cells in the culture. Since our data only capture the average transcriptomic state of the entire culture we observe a decrease in amplitude but a conservation in mesor and phase. In the other plausible scenario the entire cell culture would remain synchronous and the continuous light condition would produce an actual reduction in amplitude reflecting a deterioration in individual cell rhythmic transcriptomes. In a third possible scenario a combination of the two previous scenarios would take place. In order to discriminate between these possible scenarios for each individual gene single cell RNA-seq data would be required.

The following instructions recreate the three possible scenarios and their effect over rhythmic gene expression patterns.

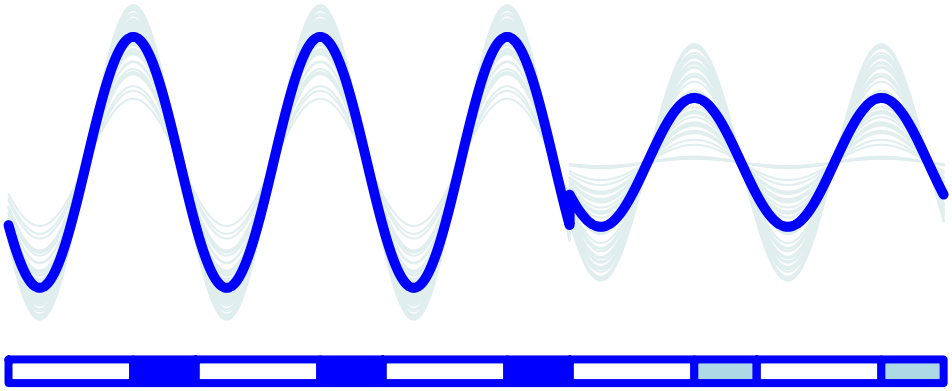
```
wave.form <- function(mesor, amplitude,period,phase,time=seq(from=0,to=48,by=0.01))
{
  y <- mesor + amplitude*cos(0.0174533*period*(time - phase))
  return(y)
}


## effect on synchronization
plot(x=0,y=0,col="white",ylim=c(20,80),xlim=c(0,120),xlab="",ylab="",axes=F)
time <- seq(from=0,to=72,by=0.01)
time.2 <- seq(from=72,to=120,by=0.01)
N <- 50
norm.random.1 <- rnorm(n = N,mean = 0,sd = 1)
norm.random.2 <- rnorm(n = N,mean = 0,sd = 6)
syn.waves <- matrix(data = 0,nrow = N,ncol = length(time))
asyn.waves <- matrix(data = 0,nrow = N,ncol = length(time.2))

for(i in 1:N)
{
  syn.waves[i,] <- wave.form(mesor = 50,amplitude = 20,period = 15,phase = 16+norm.random.1[i],time=time
  asyn.waves[i,] <- wave.form(mesor = 50,amplitude = 20,period = 15,phase = 16+norm.random.2[i],time=ti
```

```
  lines(time,syn.waves[i,],type="l",lwd=1,col="azure2")
  lines(time.2,asyn.waves[i,],type="l",lwd=1,col="azure2")

}

lines(x=c(time,time.2),y=c(colMeans(syn.waves),colMeans(asyn.waves)),type="l",lwd=5,col="blue")

polygon(x = c(0,16,16,0),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(16,24,24,16),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
polygon(x = c(24,40,40,24),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(40,48,48,40),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
polygon(x = c(48,64,64,48),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(64,72,72,64),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
polygon(x = c(72,88,88,72),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(88,96,96,88),y=c(25,25,22,22),lwd=4,border="blue",col="lightblue")
polygon(x = c(96,112,112,96),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(112,120,120,112),y=c(25,25,22,22),lwd=4,border="blue",col="lightblue")
```

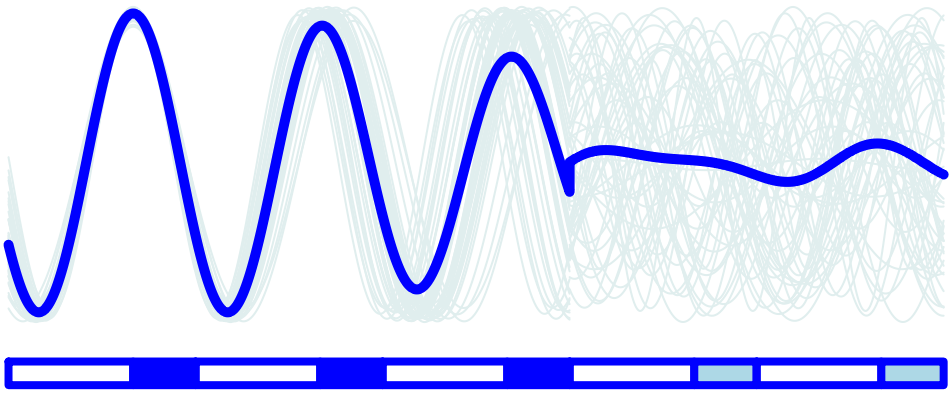


```
## effect on amplitude
plot(x=0,y=0,col="white",ylim=c(20,80),xlim=c(0,120),xlab="",ylab="",axes=F)
time <- seq(from=0,to=72,by=0.01)
time.2 <- seq(from=72,to=120,by=0.01)
N <- 50
norm.random.1 <- - abs(rnorm(n = N,mean = 0,sd = 5))
norm.random.2 <- -abs(rnorm(n = N,mean = 0,sd = 15))
syn.waves <- matrix(data = 0,nrow = N,ncol = length(time))
asyn.waves <- matrix(data = 0,nrow = N,ncol = length(time.2))
for(i in 1:N)
{
  syn.waves[i,] <- wave.form(mesor = 50,amplitude = 20+norm.random.1[i],period = 15,phase = 16,time=time
  asyn.waves[i,] <- wave.form(mesor = 50,amplitude = abs(15+norm.random.2[i]),period = 15,phase = 16,ti
  lines(time,syn.waves[i,],type="l",lwd=1,col="azure2")
  lines(time.2,asyn.waves[i,],type="l",lwd=1,col="azure2")

}

lines(x=c(time,time.2),y=c(colMeans(syn.waves),colMeans(asyn.waves)),type="l",lwd=5,col="blue")
polygon(x = c(0,16,16,0),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(16,24,24,16),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
polygon(x = c(24,40,40,24),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(40,48,48,40),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
```

```
polygon(x = c(48,64,64,48),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(64,72,72,64),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
polygon(x = c(72,88,88,72),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(88,96,96,88),y=c(25,25,22,22),lwd=4,border="blue",col="lightblue")
polygon(x = c(96,112,112,96),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(112,120,120,112),y=c(25,25,22,22),lwd=4,border="blue",col="lightblue")
```



```
## effect on synchronization + amplitude
plot(x=0,y=0,col="white",ylim=c(20,80),xlim=c(0,120),xlab="",ylab="",axes=F)
time <- seq(from=0,to=72,by=0.01)
time.2 <- seq(from=72,to=120,by=0.01)
N <- 50
norm.random.1 <- - abs(rnorm(n = N,mean = 0,sd = 1))
norm.random.2 <- -abs(rnorm(n = N,mean = 0,sd = 8))
norm.random.3 <- rnorm(n = N,mean = 0,sd = 1)
norm.random.4 <- rnorm(n = N,mean = 0,sd = 4)

syn.waves <- matrix(data = 0,nrow = N,ncol = length(time))
asyn.waves <- matrix(data = 0,nrow = N,ncol = length(time.2))
for(i in 1:N)
{
  syn.waves[i,] <- wave.form(mesor = 50,amplitude = 20+norm.random.1[i],period = 15+norm.random.3[i],pha
  asyn.waves[i,] <- wave.form(mesor = 50,amplitude = abs(20+norm.random.2[i]),period = 15+norm.random.4
  lines(time,syn.waves[i,],type="l",lwd=1,col="azure2")
  lines(time.2,asyn.waves[i,],type="l",lwd=1,col="azure2")

}

lines(x=c(time,time.2),y=c(colMeans(syn.waves),colMeans(asyn.waves)),type="l",lwd=5,col="blue")
polygon(x = c(0,16,16,0),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(16,24,24,16),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
polygon(x = c(24,40,40,24),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(40,48,48,40),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
polygon(x = c(48,64,64,48),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(64,72,72,64),y=c(25,25,22,22),lwd=4,border="blue",col="blue")
polygon(x = c(72,88,88,72),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(88,96,96,88),y=c(25,25,22,22),lwd=4,border="blue",col="lightblue")
polygon(x = c(96,112,112,96),y=c(25,25,22,22),lwd=4,border="blue")
polygon(x = c(112,120,120,112),y=c(25,25,22,22),lwd=4,border="blue",col="lightblue")
```



```
plot.ld.nd.sd <- function(gene.id, gene.name, ld.gene.expression,nd.gene.expression,sd.gene.expression,l
{
  if(nd)
  {
    scale <- T
  }

  current.gene.expression.ld <- 0
  current.gene.expression.nd <- 0
  current.gene.expression.sd <- 0

  if(ld)
  {
    ld.zt <- paste("ld",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")
    current.gene.expression.ld <- ld.gene.expression[gene.id,c(paste(ld.zt,1,sep="_"),paste(ld.zt,2,sep=
    if(scale)
    {
      current.gene.expression.ld <- scale(current.gene.expression.ld)
    }
  }

  if(sd)
  {
    sd.zt <- paste("sd",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")
    current.gene.expression.sd <-        sd.gene.expression[gene.id,c(paste(sd.zt,1,sep="_"),paste(sd.z
    if(scale)
    {
      current.gene.expression.sd <- scale(current.gene.expression.sd)
    }
}

  if(nd)
  {
  nd.zt <- paste0("ZT",seq(from=0,to=21,by=3))
  current.gene.expression.nd <- scale(nd.gene.expression[gene.id,c(paste(nd.zt,1,sep="_"),paste(nd.zt,2
}

  max.expr <- max(c(current.gene.expression.ld, current.gene.expression.nd, current.gene.expression.sd))
  min.expr <- min(c(current.gene.expression.ld, current.gene.expression.nd, current.gene.expression.sd))

  plot(x = -10,y= -10,axes=F,xlab="",ylab="",
```

```
       ylim=c(min.expr-2, max.expr),xlim=c(0,72),
       main=paste(gene.id, gene.name,sep=" - "),cex.main=2)

  if(!scale)
  {
    axis(side=2,lwd=3)
  }

  if(ld)
  {
  lines(x = seq(from=0,by=4,to=68),current.gene.expression.ld,type="o",lwd=3,col="blue")
}

  if(sd)
  {
  lines(x = seq(from=0,by=4,to=68),current.gene.expression.sd,type="o",lwd=3,col="red")
}

  if(nd)
  {
  lines(x = seq(from=0,by=3,to=69),current.gene.expression.nd,type="o",lwd=3,col="black")
}

  for(i in 0:2)
  {
  current.line <- 0.5

  if(ld)
  {
    polygon(x = c(24*i, 24*i+16, 24*i+16, 24*i),
            y=c(min.expr-current.line, min.expr-current.line, min.expr-(current.line+0.25), min.expr-(cu
            lwd=2,border="blue")
    polygon(x = c(24*i+16,24*(i+1),24*(i+1),24*i+16),
            y=c(min.expr-current.line, min.expr-current.line, min.expr-(current.line+0.25), min.expr-(cu
            lwd=2,border="blue",col="blue")
    current.line <- current.line + 0.5
  }

  if(nd)
  {
    polygon(x = c(24*i, 24*i+12, 24*i+12, 24*i),
            y=c(min.expr-current.line, min.expr-current.line, min.expr-(current.line+0.25), min.expr-(cu
            lwd=2,border="black")
    polygon(x = c(24*i+12,24*(i+1),24*(i+1),24*i+12),
            y=c(min.expr-current.line, min.expr-current.line, min.expr-(current.line+0.25), min.expr-(cu
            lwd=2,border="black",col="black")
    current.line <- current.line + 0.5
  }

  if(sd)
  {
    polygon(x = c(24*i, 24*i+8, 24*i+8, 24*i),
            y=c(min.expr-current.line, min.expr-current.line, min.expr-(current.line+0.25), min.expr-(cu
```

```
            lwd=2,border="red")
    polygon(x = c(24*i+8,24*(i+1),24*(i+1),24*i+8),
            y=c(min.expr-current.line, min.expr-current.line, min.expr-(current.line+0.25), min.expr-(cu
            lwd=2,border="red",col="red")
  }
}

  return(0)
}

plot.ld.nd.sd(gene.id = "ostta01g02580",gene.name = "MCM6", ld.gene.expression,nd.gene.expression,sd.ge

plot.ld.nd.sd(gene.id = "ostta04g00450",gene.name = "MCM5", ld.gene.expression,nd.gene.expression,sd.ge

plot.ld.nd.sd(gene.id = "ostta01g01370",gene.name = "???", ld.gene.expression,nd.gene.expression,sd.gene


# i <- 1
# plot.ld.nd.sd(gene.id = common.12.nd.sd[i],gene.name = "???", ld.gene.expression,nd.gene.expression,s
# i <- i + 1
#
# i <- i - 1
# plot.ld.nd.sd(gene.id = common.12.nd.sd[i],gene.name = "???", ld.gene.expression,nd.gene.expression,s

i <- 1
plot.ld.nd.sd(gene.id = sd.specific.rhythmic.genes[i],gene.name = "???", ld.gene.expression,nd.gene.exp
i <- i + 1
i <- i - 1


plot.ld.nd.sd(gene.id = "ostta13g01820",gene.name = "???", ld.gene.expression,nd.gene.expression,sd.gene



plot.ld.ll <- function(gene.id, gene.name, gene.expression)
{
    ld.zt <- paste("ld",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")
    current.gene.expression.ld.ll <- gene.expression[gene.id,c(paste(ld.zt,1,sep="_"),paste(ld.zt,2,sep

    min.expression <- min(current.gene.expression.ld.ll)
    max.expression <- max(current.gene.expression.ld.ll)
    range.expression <- max.expression - min.expression

    expression.step <- floor(range.expression / 5)

    plot(current.gene.expression.ld.ll,type="o",lwd=3,col="blue",axes=F,xlab="",ylab="FPKM",
         ylim=c(min.expression-expression.step,max.expression),
         cex.lab=1.3,main=paste(gene.id, gene.name,sep=" - "),cex.main=2)
    axis(side=2,lwd=3)
    axis(side = 1,pos = min.expression - 1.1*expression.step, at = seq(from=1,to=30),
         labels = rep(paste("ZT",seq(from=0,to=20,by=4)),5),las=2,lwd=3)

    polygon(x = c(1,5,5,1),y=c(min.expression-expression.step/2,
```

```
                                min.expression-expression.step/2,
                                min.expression-expression.step,
                                min.expression-expression.step),lwd=2,border="blue")

    polygon(x = c(5,7,7,5),y=c(min.expression-expression.step/2,
                                min.expression-expression.step/2,
                                min.expression-expression.step,
                                min.expression-expression.step),lwd=2,border="blue",col="blue")

    polygon(x = c(7,11,11,7),y=c(min.expression-expression.step/2,
                                  min.expression-expression.step/2,
                                  min.expression-expression.step,
                                  min.expression-expression.step),lwd=2,border="blue")

    polygon(x = c(11,13,13,11),y=c(min.expression-expression.step/2,
                                    min.expression-expression.step/2,
                                    min.expression-expression.step,
                                    min.expression-expression.step),lwd=2,border="blue",col="blue")

    polygon(x = c(13,17,17,13),y=c(min.expression-expression.step/2,
                                    min.expression-expression.step/2,
                                    min.expression-expression.step,
                                    min.expression-expression.step),lwd=2,border="blue")

    polygon(x = c(17,19,19,17),y=c(min.expression-expression.step/2,
                                    min.expression-expression.step/2,
                                    min.expression-expression.step,
                                    min.expression-expression.step),lwd=2,border="blue",col="blue")

    polygon(x = c(19,23,23,19),y=c(min.expression-expression.step/2,
                                    min.expression-expression.step/2,
                                    min.expression-expression.step,
                                    min.expression-expression.step),lwd=2,border="blue")

    polygon(x = c(23,25,25,23),y=c(min.expression-expression.step/2,
                                    min.expression-expression.step/2,
                                    min.expression-expression.step,
                                    min.expression-expression.step),lwd=2,border="blue",col="lightblue")

    polygon(x = c(25,29,29,25),y=c(min.expression-expression.step/2,
                                    min.expression-expression.step/2,
                                    min.expression-expression.step,
                                    min.expression-expression.step),lwd=2,border="blue")
    polygon(x = c(29,30,30,29),y=c(min.expression-expression.step/2,
                                    min.expression-expression.step/2,
                                    min.expression-expression.step,
                                    min.expression-expression.step),lwd=2,border="blue",col="lightblue")

}


i <- 1
plot.ld.ll(gene.id = "ostta04g00450",gene.name = "MCM5", gene.expression)
```

```
i <- 1
plot.ld.ll(gene.id = complete.ld.ll.rhythmic.genes[i],gene.name = i, gene.expression)
i <- i + 1


plot.sd.ll <- function(gene.id, gene.name, gene.expression)#, mean.expression.ld, mean.expression.sd)
{
    sd.zt <- paste("sd",paste0("zt",sprintf(fmt = "%02d",seq(from=0,to=20,by=4))),sep="_")
    current.gene.expression.sd.ll <- gene.expression[gene.id,c(paste(sd.zt,1,sep="_"),paste(sd.zt,2,sep=

  min.expression <- min(current.gene.expression.sd.ll)
  max.expression <- max(current.gene.expression.sd.ll)
  range.expression <- max.expression - min.expression

  expression.step <- floor(range.expression / 5)

  plot(current.gene.expression.sd.ll,type="o",lwd=3,col="red",axes=F,xlab="",ylab="FPKM",
       ylim=c(min.expression-expression.step,max.expression),
       cex.lab=1.3,main=paste(gene.id, gene.name,sep=" - "),cex.main=2)
  axis(side=2,lwd=3)
  axis(side = 1,pos = min.expression - 1.1*expression.step, at = seq(from=1,to=30),
       labels = rep(paste("ZT",seq(from=0,to=20,by=4)),5),las=2,lwd=3)

  polygon(x = c(1,3,3,1),y=c(min.expression-expression.step/2,
                             min.expression-expression.step/2,
                             min.expression-expression.step,
                             min.expression-expression.step),lwd=2,border="red")

  polygon(x = c(3,7,7,3),y=c(min.expression-expression.step/2,
                             min.expression-expression.step/2,
                             min.expression-expression.step,
                             min.expression-expression.step),lwd=2,border="red",col="red")

  polygon(x = c(7,9,9,7),y=c(min.expression-expression.step/2,
                               min.expression-expression.step/2,
                               min.expression-expression.step,
                               min.expression-expression.step),lwd=2,border="red")

  polygon(x = c(9,13,13,9),y=c(min.expression-expression.step/2,
                                 min.expression-expression.step/2,
                                 min.expression-expression.step,
                                 min.expression-expression.step),lwd=2,border="red",col="red")

  polygon(x = c(13,15,15,13),y=c(min.expression-expression.step/2,
                                 min.expression-expression.step/2,
                                 min.expression-expression.step,
                                 min.expression-expression.step),lwd=2,border="red")

  polygon(x = c(15,19,19,15),y=c(min.expression-expression.step/2,
                                 min.expression-expression.step/2,
                                 min.expression-expression.step,
                                 min.expression-expression.step),lwd=2,border="red",col="red")
```

```
  polygon(x = c(19,21,21,19),y=c(min.expression-expression.step/2,
                                 min.expression-expression.step/2,
                                 min.expression-expression.step,
                                 min.expression-expression.step),lwd=2,border="red")

  polygon(x = c(21,25,25,21),y=c(min.expression-expression.step/2,
                                 min.expression-expression.step/2,
                                 min.expression-expression.step,
                                 min.expression-expression.step),lwd=2,border="red",col="lightsalmon")

  polygon(x = c(25,27,27,25),y=c(min.expression-expression.step/2,
                                 min.expression-expression.step/2,
                                 min.expression-expression.step,
                                 min.expression-expression.step),lwd=2,border="red")
  polygon(x = c(27,30,30,27),y=c(min.expression-expression.step/2,
                                 min.expression-expression.step/2,
                                 min.expression-expression.step,
                                 min.expression-expression.step),lwd=2,border="red",col="lightsalmon")

}


i <- 1
plot.sd.ll(gene.id = big.amplitude[i],gene.name = i, gene.expression)
i <- i + 1


sd.non.rhythmic.ll <- setdiff(complete.sd.rhythmic.genes,complete.sd.ll.rhythmic.genes)
plot.sd.ll(gene.id = sd.non.rhythmic.ll[i],gene.name = i, gene.expression)
i <- i + 1


plot.sd.ll(gene.id = rhythmic.genes.sd.12[i],gene.name = i, gene.expression)
i <- i + 1

i <- 1
plot.ld.ll(gene.id = complete.ld.ll.rhythmic.genes[i],gene.name = "Nitrogen regulatory PII-like, alpha/
i <- i + 1
i <- i - 1
```