Supplementary Information

ShinyR-DAM: A program analyzing *Drosophila* activity, sleep, and circadian rhythms

Karol Cichewicz, kc3fb@virginia.edu; Jay Hirsh. University of Virginia, Charlottesville, VA.

Contents

The aim of this supplementary information

This supplementary information explains basics of the TriKinetics DAMSystem3 monitor file format, for users who would like to make other types of data compatible with ShinyR-DAM.

ShinyR-DAM allows download of data files in the CSV format, that can be used for further statistical analysis and plotting. In this document we explain the data stored in each CSV file and provide R code snippets that a user can use to recreate and edit plots using RStudio or other integrated development environments.

This document was written in the markup language R Markdown, that allows one to integrate R code into a text document. We provide an Rmd file and all CSV data files necessary to recreate this PDF in RStudio.

ShinyR-DAM input file format

DAMSystem3 monitor file format

ShinyR-DAM input files are TriKinetics (Waltham, MA) monitor files produced by the DAMSystem3 data acquisition software. The following description of DAMSystem3 monitor file format specification is adopted from TriKinetics User's Guide v3.0.pdf. Please refer to the original documentation for more details. These files are text files, and can be viewed or edited with any text editor. Text lines are terminated with CRLF on both Macintosh and Windows platforms for compatibility. Monitor files are named MonitorNNN.txt, with NNN representing monitor numbers from 1 to 120. Each monitor file contains 42 tab-delimited columns, with rows representing sequential data. Monitor files can also be produced by the DAMFileScan software, which can save validated and time-filtered versions of the raw monitor files.

The columns of a monitor file are as follows:

1.Index at reading (from 1 with each program restart)
2. Date of reading (9 Dec 18)
3. Time of reading (19:09:30) in format hh:mm:ss
4. Monitor status (1 = valid data received)
5. Extra readings included in this bin (DAMFileScan outputs only)
6. unused (0)
7. unused (0)
8. unused (0)

9. unused (0)
10. DAM2 Light Status (1 = On, 0 = Off)
11. Channel 1 - activity counts
12. Channel 2 - activity counts
...
...
42. Channel 32 - activity counts

All monitor file columns except 2 and 3 contain numerical data. Columns 5-10 are ignored by ShinyR-DAM, although they must be present in the input files.

Any text file meeting this 42-column, date-time specification is compatible with ShinyR-DAM.

An example of a monitor file:

```
Monitor_file <- read.table("Monitor61.txt", sep = "\t") #Reads a monitor file

knitr::kable(Monitor_file[1:6, 1:21]) #Displays the first 6 rows and 21 columns of a data frame
```

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21
95401	22 Jun 17	17:25:00	1	0	0	0	0	0	0	5	5	3	0	1	3	0	2	4	3	0
95402	22 Jun 17	17:26:00	1	0	0	0	0	0	0	4	2	2	0	3	3	0	2	2	3	3
95403	22 Jun 17	17:27:00	1	0	0	0	0	0	0	4	1	1	0	1	1	0	2	2	2	2
95404	22 Jun 17	17:28:00	1	0	0	0	0	0	0	5	2	3	0	8	2	0	2	2	2	3
95405	22 Jun 17	17:29:00	1	0	0	0	0	0	0	3	3	1	0	1	2	0	2	4	1	1
95406	22 Jun 17	17:30:00	1	0	0	0	0	0	0	8	3	3	0	1	1	0	2	3	0	1

Legacy DAMSystem2 channel file format

DAMSystem2 channel files are **NOT** compatible with ShinyR-DAM. However, for users who still use this file format we created a DAM2 to DAM3 file format converter. For the channel file format specification please refer to the TriKinetics User's Guide v3.0.pdf

R environment

Setting up the R environment

```
# Setting working directory. A user must set a directory where all input files for the script are located.
# Please make sure that input files are directly accessible in your working directory. They cannot be compressed.
# Instead of using the setwd command you can set your working directory in RStudio by going to
# Session > Set Working Directory > Choose Directory

setwd("C:/Users/Karol/Desktop/JH_lab/ShinyR-DAM program/ShinyR-DAM 2.1 Review woring copy 1/Code snippets")

# Loads R packages necessary to conduct computations and generate plots.
# If you don't have them already installed you can do it in RStudio by going to Tools > Install Packages
# or by executing install.packages("package_name") in a console.

library(knitr)
library(ggplot2)
library(gridExtra)
library(grid)
library(dplyr)
library(plyr)
library(zoo)
library(gtools)
library(scales)
library(gridExtra)
library(data.table)
library(Kmisc)
library(lubridate)
library(grid)
```

R session information

ShinyR-DAM and this Supplementary Information were developed in RStudio Version 1.1.414 with the following environemnt configuration:

R version 3.4.3 (2017-11-30)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows >= 8 x64 (build 9200)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.1252 LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252 LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
[1] grid stats graphics grDevices utils datasets methods base

other attached packages:
[1] bindrcpp_0.2 colourpicker_1.0 lubridate_1.7.1 Kmisc_0.5.0
[5] data.table_1.10.4-3 gridExtra_2.3 scales_0.5.0 gtools_3.5.0
[9] zoo_1.8-1 dplyr_0.7.4 ggplot2_2.2.1 plyr_1.8.4
[13] shiny_1.0.5

loaded via a namespace (and not attached):
[1] Rcpp_0.12.14 pillar_1.1.0 compiler_3.4.3 bindr_0.1 bitops_1.0-6
[6] tools_3.4.3 digest_0.6.14 evaluate_0.10.1 jsonlite_1.5 tibble_1.4.1
[11] gtable_0.2.0 lattice_0.20-35 pkgconfig_2.0.1 rlang_0.1.6 yaml_2.1.16
[16] stringr_1.2.0 knitr_1.18 htmlwidgets_0.9 rprojroot_1.3-2 glue_1.2.0
[21] R6_2.2.2 rmarkdown_1.8 RJSONIO_1.3-0 reshape2_1.4.3 magrittr_1.5
[26] backports_1.1.2 htmltools_0.3.6 rsconnect_0.8.5 assertthat_0.2.0 mime_0.5
[31] xtable_1.8-2 colorspace_1.3-2 httpuv_1.3.5 labeling_0.3 miniUI_0.1.1
[36] stringi_1.1.6 RCurl_1.95-4.10 lazyeval_0.2.1 munsell_0.4.3 markdown_0.8

We provide these details for compatibility in case users experience compatibility issues caused by different R or package versions.

Daily Locomotor Activity

Average Daily Locomotor Activity

```
Daily_locomotor_activity <- read.csv("Daily_locomotor_activity.csv") #Reads a csv file and saves it into an object

knitr::kable(head(Daily_locomotor_activity)) #Displays the first 6 rows of a data frame
```

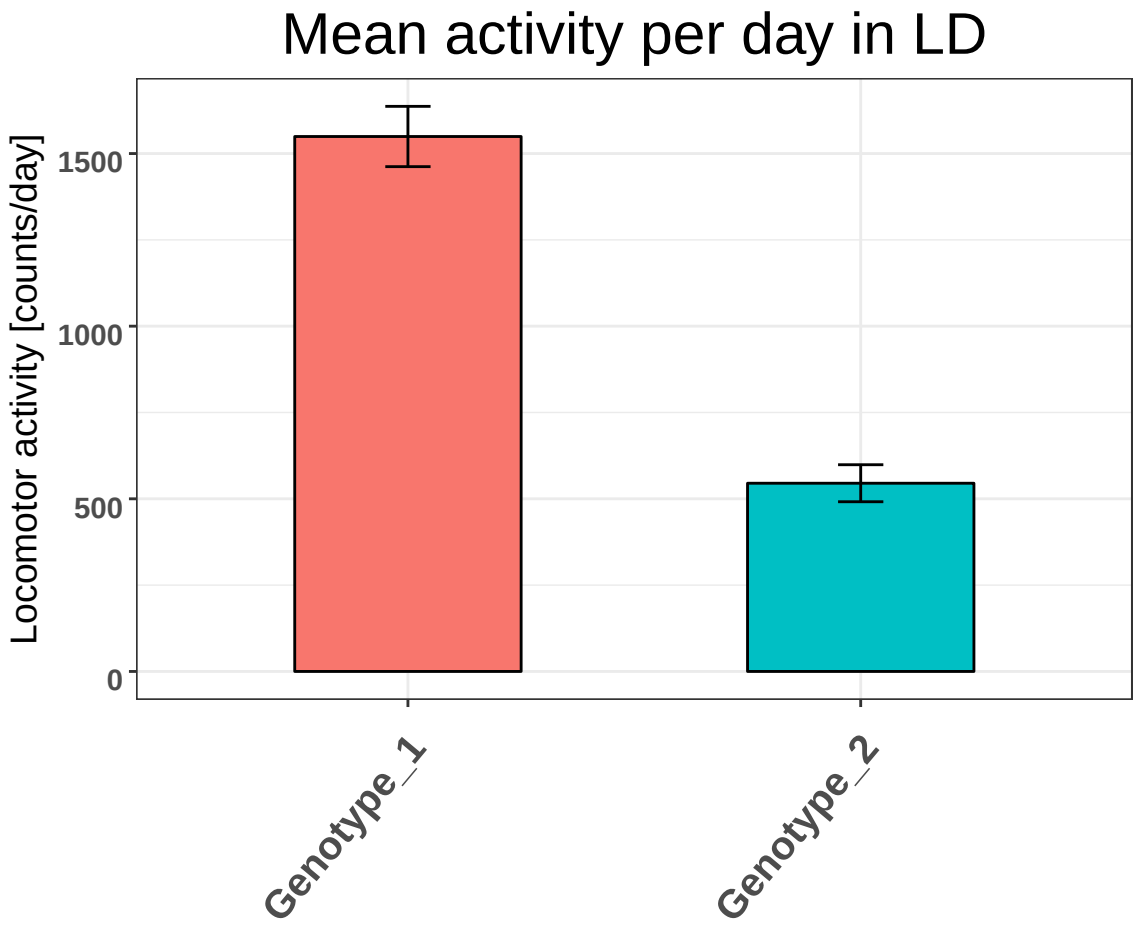
X	Condition	Light_cycle	Mean	SD	SEM	N_living_flies	Dead_flies	All_flies
1	Genotype_1	DD	1330.0500	436.3631	82.46488	28	4	32
2	Genotype_1	LD	1549.3500	461.8957	87.29008	28	4	32
3	Genotype_2	DD	401.7571	276.6109	52.27456	28	4	32
4	Genotype_2	LD	545.1357	283.6048	53.59628	28	4	32

Column description:

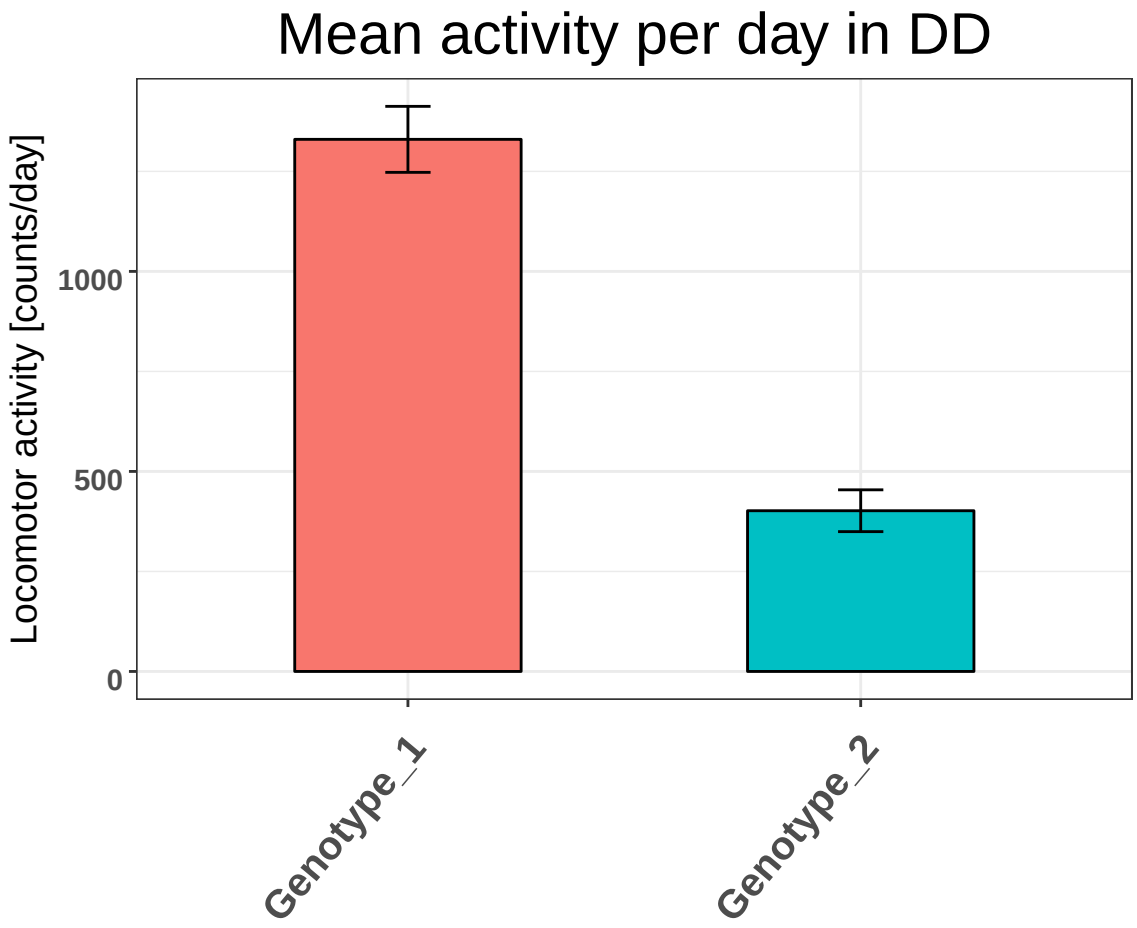
- X - row index
- Condition - a user-defined name of an experimental condition
- Light_cycle - LD or DD. LD - days with 12:12 light dark cycle, DD - constant darkness
- Mean - mean locomotor activity per day.
- SD - standard deviation between individuals. If an experiment is conducted over multiple days, first, the average values across multiple days are calculated for each individual, then an average across individuals of a condition is calculated. SD represents this individual-to-individual variance.
- SEM - standard error of the mean
- N_living_flies - number of living flies at the end of the experiment. Dead flies are excluded from calculating statistics, unless a user sets *Threshold of counts per day for identifying dead flies* to 0.
- Dead_flies - number of dead flies
- All_flies - number of all flies in a condition in the beginning of an experiment

```
# Filters LD data
Daily_locomotor_activity_LD <- filter(Daily_locomotor_activity, Light_cycle == 'LD')

# Plots a bar plot
  barplot_LD<- ggplot(Daily_locomotor_activity_LD, aes(x=Condition, y=Mean, fill=Condition, width=.5)) +
      geom_bar(stat = "summary", fun.y = "mean", colour="black") +          #plots bars, mean value
      labs(x="", y="Locomotor activity [counts/day]") +                     #adds/removes axis lables
```

```
    theme_bw()+
    geom_errorbar(data=Daily_locomotor_activity_LD, aes(ymax=Mean+SEM,ymin=Mean-SEM), width=0.1)+
    labs(title= "Mean activity per day in LD", size= rel(2))+
    theme(legend.text=element_text(size=16))+
    theme(legend.position="none")+
    theme(axis.text.x=element_text(angle=50, vjust=0.9, hjust=1, size=15, face="bold"))+
    theme(plot.title = element_text(size = rel(2), hjust=0.5))+
    theme(axis.text.y=element_text(vjust=0.9, hjust=1, size=11, face="bold"))+
    theme(axis.title.y = element_text(size=14))


barplot_LD
```


Mean activity per day in LD
Locomotor activity [counts/day]
1500
1000
500
0
Genotype_1
Genotype_2

```r
# Filters DD data
  Daily_locomotor_activity_DD <- filter(Daily_locomotor_activity, Light_cycle == 'DD')

  # Plots a bar plot
  barplot_DD<- ggplot(Daily_locomotor_activity_DD, aes(x=Condition, y=Mean, fill=Condition, width=.5)) +
      geom_bar(stat = "summary", fun.y = "mean", colour="black") +          #plots bars,mean value
      labs(x="", y="Locomotor activity [counts/day]") +                     #adds/removes axis lables
      theme_bw()+
      geom_errorbar(data=Daily_locomotor_activity_DD, aes(ymax=Mean+SEM,ymin=Mean-SEM), width=0.1)+
      labs(title= "Mean activity per day in DD", size= rel(2))+
      theme(legend.text=element_text(size=16))+
      theme(legend.position="none")+
      theme(axis.text.x=element_text(angle=50, vjust=0.9, hjust=1, size=15, face="bold"))+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      theme(axis.text.y=element_text(vjust=0.9, hjust=1, size=11, face="bold"))+
      theme(axis.title.y = element_text(size=14))


  barplot_DD
```


Mean activity per day in DD
Locomotor activity [counts/day]
1000
500
0
Genotype_1
Genotype_2

Average Individual Daily Locomotor Activity

```
Average_individual_daily_locomotor_activity <- read.csv("Average_individual_daily_locomotor_activity.csv") #Reads a csv file

knitr::kable(head(Average_individual_daily_locomotor_activity)) #Displays the first 6 rows of a data frame
```

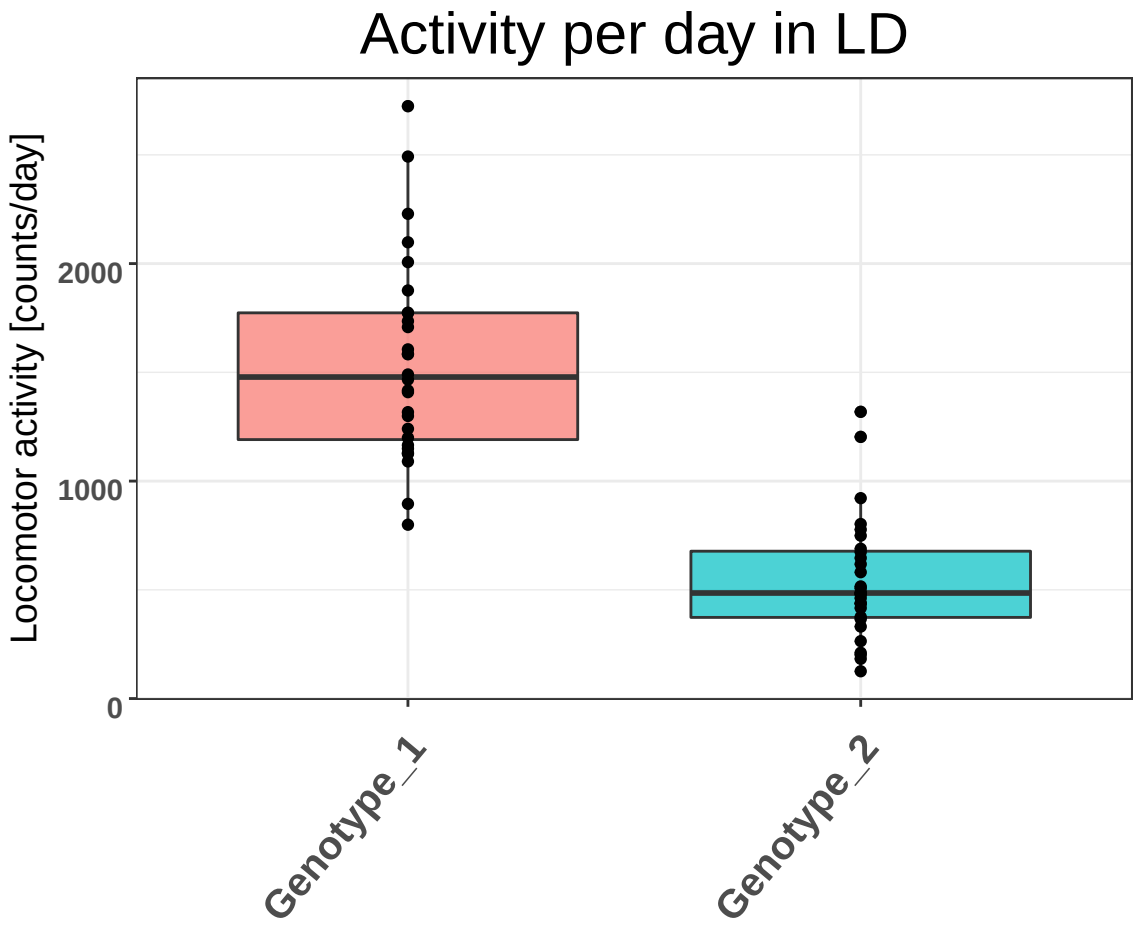
X	Condition	variable	Light_cycle	mean_value
1	Genotype_1	Monitor61_ch1	LD	1708.0
2	Genotype_1	Monitor61_ch1	DD	1445.8
3	Genotype_1	Monitor61_ch2	LD	895.2
4	Genotype_1	Monitor61_ch2	DD	552.4
5	Genotype_1	Monitor61_ch3	LD	1125.2
6	Genotype_1	Monitor61_ch3	DD	1293.6

Column description:

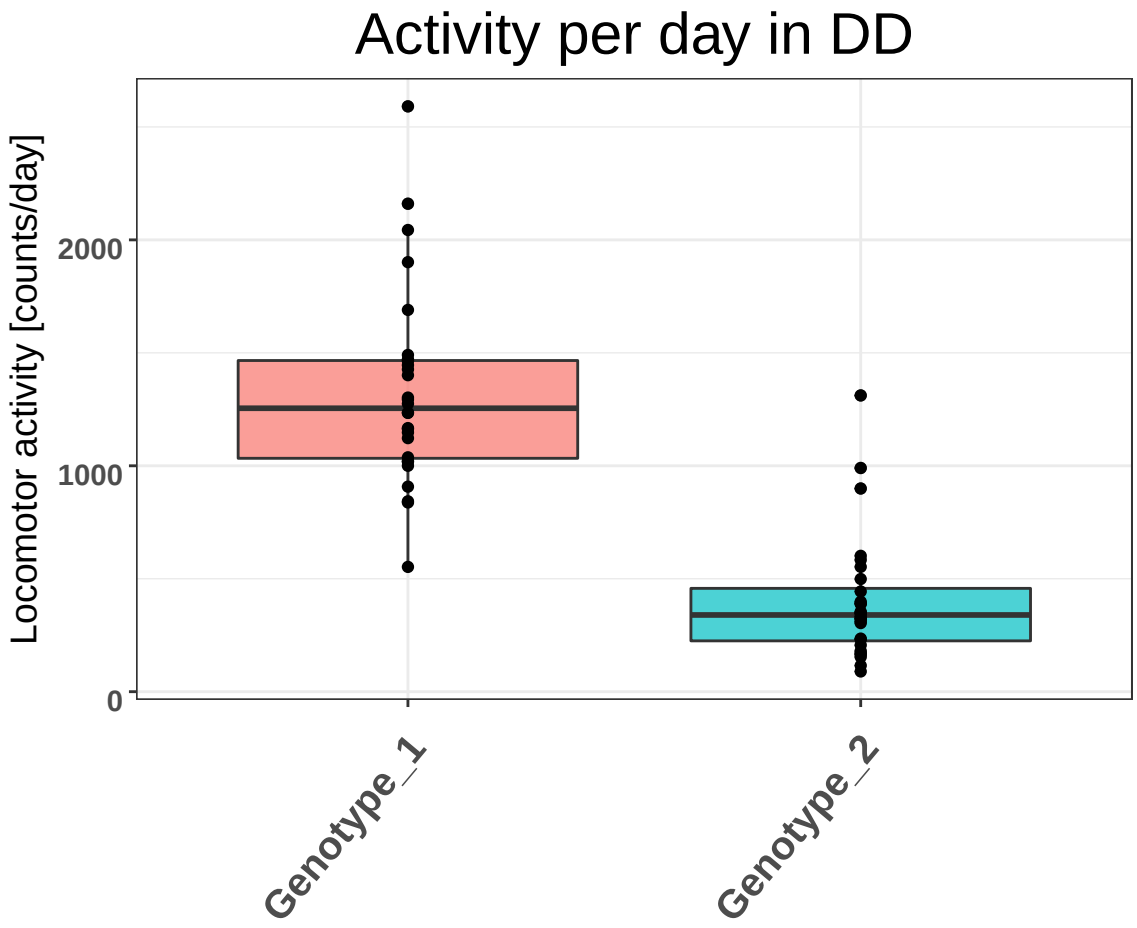
- X - row index
- Condition - a user-defined name of an experimental condition
- variable - DAM system channel
- Light_cycle - LD or DD. LD - days with 12:12 light dark cycle, DD - constant darkness
- Mean_value - mean locomotor activity [number of counts] per day

```
# Filters LD data
Average_individual_daily_locomotor_activity_LD <- filter(Average_individual_daily_locomotor_activity, Light_cycle == 'LD')

#Box plot of individual locomotor activity in LD
box_plot_LD<- ggplot(na.omit(Average_individual_daily_locomotor_activity_LD), aes(x=Condition, y=mean_value, fill=Condition)) +
      geom_boxplot(alpha=0.7)+
      geom_point()+
      labs(title= "Activity per day in LD")+
      labs(y="Locomotor activity [counts/day]", x="") +  #adds/removes axis lables
      theme_bw()+
      theme(legend.position="none")+
      theme(legend.text=element_text(size=16))+
      theme(legend.position="none")+
```

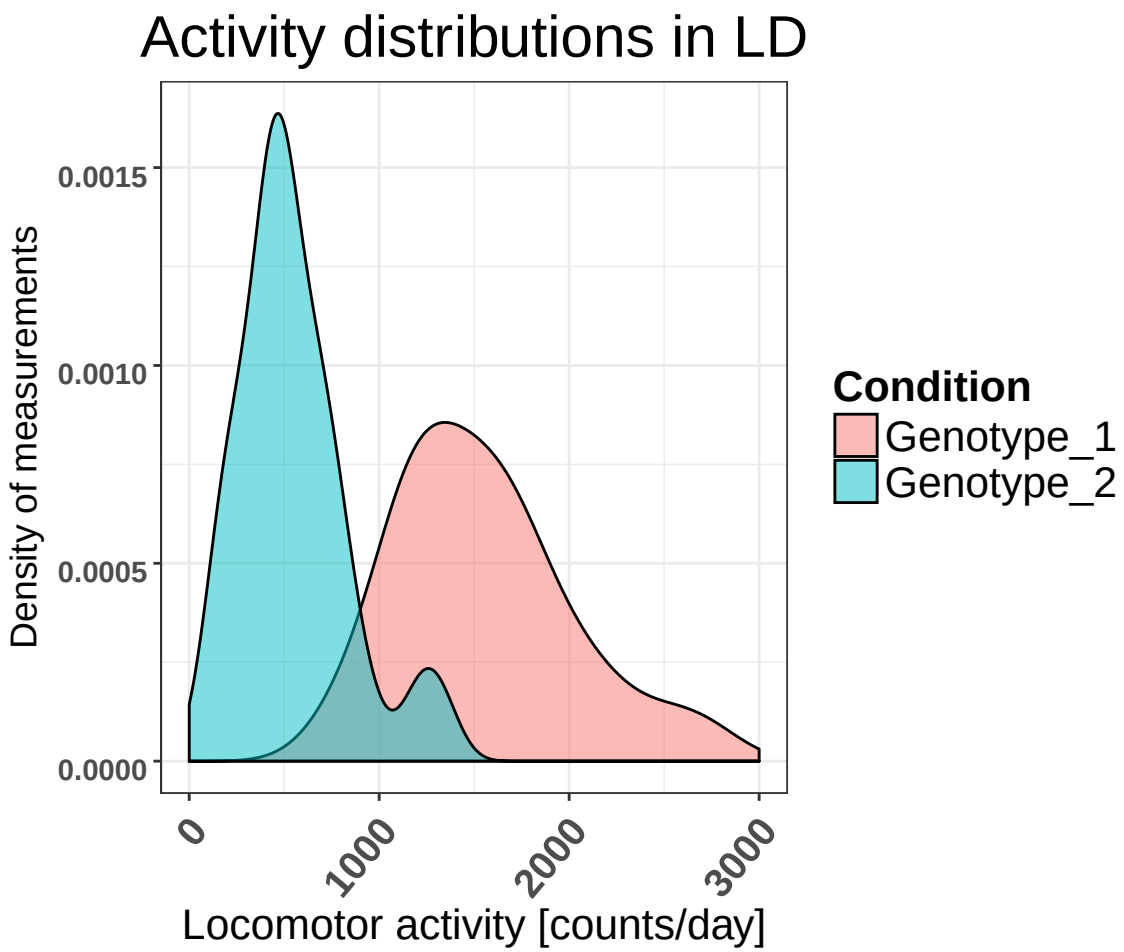
```
      theme(axis.text.x=element_text(angle=50, vjust=0.9, hjust=1, size=15, face="bold"))+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      theme(axis.text.y=element_text(vjust=0.9, hjust=1, size=11, face="bold"))+
      theme(axis.title.y = element_text(size=14))


box_plot_LD
```



```r
# Filters DD data
Average_individual_daily_locomotor_activity_DD <- filter(Average_individual_daily_locomotor_activity, Light_cycle == 'DD')

#Box plot of individual locomotor activity in DD
box_plot_DD<- ggplot(na.omit(Average_individual_daily_locomotor_activity_DD), aes(x=Condition, y=mean_value, fill=Condition)) +
      geom_boxplot(alpha=0.7)+
      geom_point()+
      labs(title= "Activity per day in DD")+
      labs(y="Locomotor activity [counts/day]", x="") +  #adds/removes axis lables
      theme_bw()+
      theme(legend.position="none")+
      theme(legend.text=element_text(size=16))+
      theme(legend.position="none")+
      theme(axis.text.x=element_text(angle=50, vjust=0.9, hjust=1, size=15, face="bold"))+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      theme(axis.text.y=element_text(vjust=0.9, hjust=1, size=11, face="bold"))+
      theme(axis.title.y = element_text(size=14))


box_plot_DD
```


Activity per day in DD
Locomotor activity [counts/day]
2000
1000
0
Genotype_1
Genotype_2

```r
# Density distribution plot in LD
density_plot_LD<- ggplot(na.omit(Average_individual_daily_locomotor_activity_LD), aes(x=mean_value, fill=Condition))+
      geom_histogram(aes(y=0.3*..density..),binwidth=100, alpha=0, position="identity") +
      geom_density(alpha=.5)+
      xlim(0,3000)+
      labs(title= "Activity distributions in LD")+
      labs(x="Locomotor activity [counts/day]", y="Density of measurements") +
      theme(legend.title=element_blank())+ #removes legend title
      theme_bw()+
      theme(legend.title=element_text(size=16, face="bold"))+
      theme(legend.text=element_text(size=16))+
      theme(axis.text.x=element_text(angle=50, vjust=0.9, hjust=1, size=15, face="bold"))+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      theme(axis.text.y=element_text(vjust=0.9, hjust=1, size=11, face="bold"))+
      theme(axis.title.y = element_text(size=14))+
      theme(axis.title.x = element_text(color="black", size=16))

density_plot_LD
```


Activity distributions in LD
Density of measurements
0.0015
0.0010
0.0005
0.0000
0
1000
2000
3000
Locomotor activity [counts/day]
Condition
Genotype_1
Genotype_2

Individual Daily Locomotor Activity

```
Individual_daily_locomotor_activity <- read.csv("Individual daily locomotor activity data.csv") #Reads a csv file

knitr::kable(head(Individual_daily_locomotor_activity)) #Displays the first 6 rows of a data frame
```

X	Condition	variable	Light_cycle	date	value
1	Genotype_1	Monitor61_ch1	LD	2017-06-23	1686
2	Genotype_1	Monitor61_ch1	LD	2017-06-24	1763
3	Genotype_1	Monitor61_ch1	LD	2017-06-25	1806
4	Genotype_1	Monitor61_ch1	LD	2017-06-26	1629
5	Genotype_1	Monitor61_ch1	LD	2017-06-27	1656
6	Genotype_1	Monitor61_ch1	DD	2017-06-28	1484

This data frame is not used for generating any plot in ShinyR-DAM, but it is provided for user convenience allowing further statistical analysis. Please not that dead flies are included in this output, allowing a user to validate the further automatic exclusion of dead flies.

Column description:

- X - row index
- Condition - a user-defined name of an experimental condition
- variable - DAM system channel
- Light_cycle - LD or DD. LD - days with 12:12 light dark cycle, DD - constant darkness
- date
- value - locomotor activity [number of counts] of an individual fly each day

Locomotor activity by day

```
Locomotor_activity_by_day <- read.csv("Locomotor_activity_by_day.csv") #Reads a csv file

knitr::kable(head(Locomotor_activity_by_day)) #Displays the first 6 rows of a data frame
```

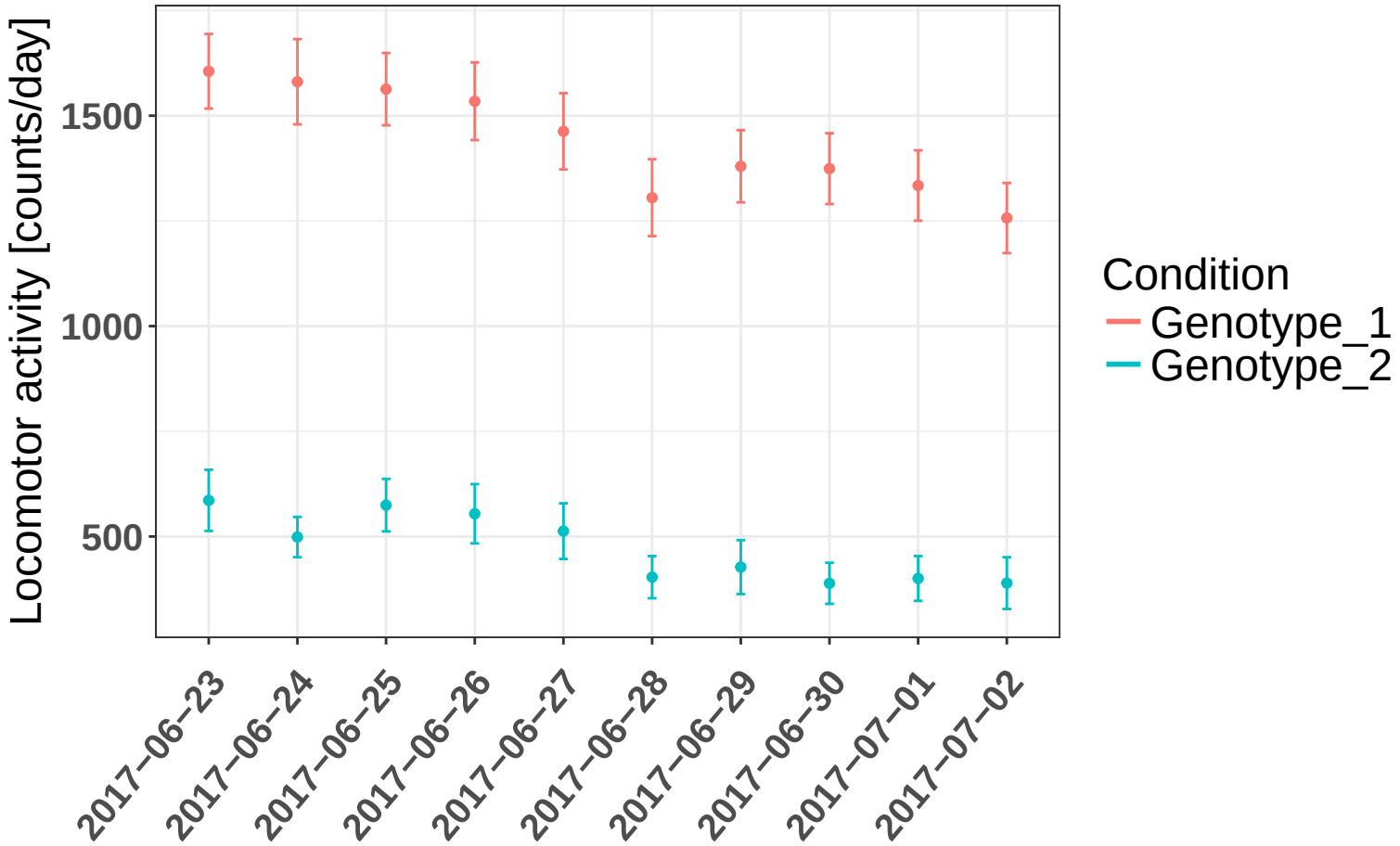
X	Condition	Light_cycle	date	mean	sum	sd	sem
1	Genotype_1	LD	2017-06-23	1605.6429	44958	468.8824	88.61044
2	Genotype_2	LD	2017-06-23	585.8571	16404	384.2547	72.61731
3	Genotype_1	LD	2017-06-24	1580.7500	44261	535.8609	101.26818
4	Genotype_2	LD	2017-06-24	498.6429	13962	253.1931	47.84900
5	Genotype_1	LD	2017-06-25	1563.1429	43768	454.7761	85.94461
6	Genotype_2	LD	2017-06-25	574.4286	16084	329.9482	62.35435

Column description:

- X - row index
- Condition - a user-defined name of an experimental condition
- Light_cycle - LD or DD. LD - days with 12:12 light dark cycle, DD - constant darkness
- date
- mean - mean locomotor activity of individuals in a condition each day
- sd - standard deviation of individuals
- sem - standard error of the mean

```
# Locomotor activity by day plot
 point_plot_by_date<- ggplot(Locomotor_activity_by_day, aes(x=date, y=mean, fill=Condition, color=Condition, width=.5))+
            geom_point(stat = "summary", fun.y = "mean", show.legend = FALSE) +
            labs(x="", y="Locomotor activity [counts/day]") +
            theme_bw()+
            geom_errorbar(data=Locomotor_activity_by_day, aes(ymax=mean+sem,ymin=mean-sem), width=0.1)+
            theme(strip.text = element_text(size=25)) +
            guides(size = guide_legend(order = 1))+
            theme(axis.title.x = element_text(color="black", size=16))+
            theme(axis.text.x=element_text(angle=50, vjust=0.9, hjust=1, size=15, face="bold"))+
            theme(axis.text.y=element_text(size=15, face="bold"))+
            theme(legend.text=element_text(size=18))+
```

```
        theme(legend.title = element_text(size=18))+
        theme(axis.title=element_text(size=18))+
        guides(colour = guide_legend(override.aes = list(size=1)))

point_plot_by_date
```



Daytime vs Nighttime activity

```
Daytime_nighttime_activity <- read.csv("Daytime_nighttime_activity_in_LD.csv") #Reads a csv file

knitr::kable(head(Daytime_nighttime_activity)) #Displays the first 6 rows of a data frame
```

X	Condition	Light_status	Mean	SEM
1	Genotype_1	Day	1144.1714	56.71807
2	Genotype_1	Night	405.1786	44.65318
3	Genotype_2	Day	400.8286	38.16972
4	Genotype_2	Night	144.3071	17.99452

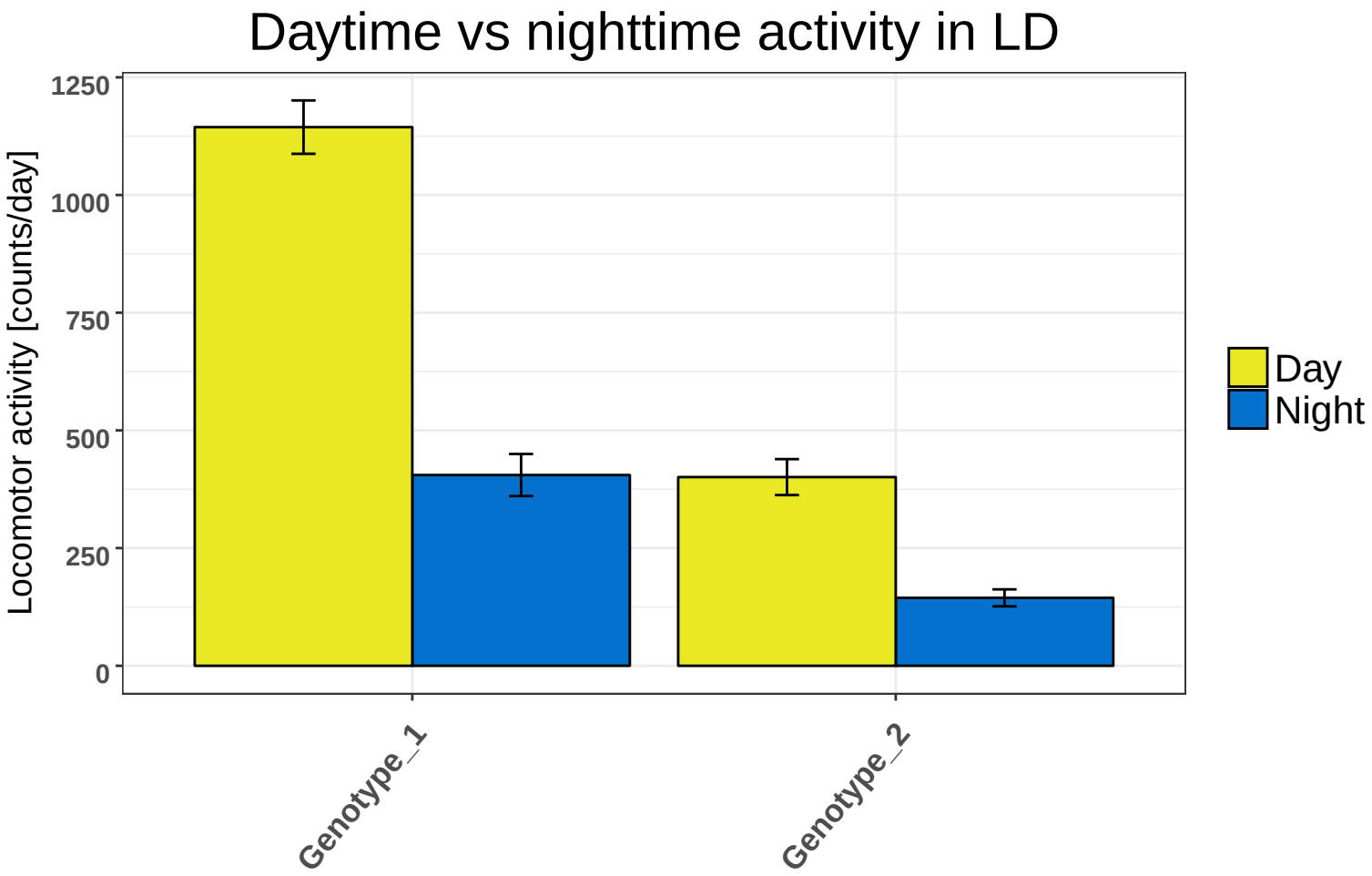
Column description:

- X - row index
- Condition - a user-defined name of an experimental condition
- Light_status - Day or Night
- Mean - mean locomotor activity of individuals in a condition during the day or night

- SEM - standard error of the mean

```
Daytime_nighttime_activity_plot <- ggplot(Daytime_nighttime_activity, aes(factor(Condition), Mean, fill = Light_status)) +
        geom_bar(stat="identity", position = "dodge", colour="black") +
        geom_errorbar(data=Daytime_nighttime_activity, aes(ymin=Mean - SEM, ymax=Mean + SEM),
                      position = position_dodge(0.9),  width=0.1) +
        scale_fill_manual(values=c("#E8E823", "#0472CC"))+
        theme_bw()+
        theme(legend.text=element_text(size=16))+
        theme(axis.text.x=element_text(angle=50, vjust=0.9, hjust=1, size=13, face="bold"))+
        labs(title= "Daytime vs nighttime activity in LD", size= 14)+
        theme(plot.title = element_text(size = rel(2), hjust=0.5))+
        theme(axis.text.y=element_text(vjust=0.9, hjust=1, size=11, face="bold"))+
        theme(axis.title.y = element_text(size=14))+
        theme(legend.title=element_blank())+
        theme(axis.title.x =  element_blank())+
        labs(x="", y="Locomotor activity [counts/day]")
```

```
Daytime_nighttime_activity_plot
```



Activity Profiles

```
Daily_activity_profiles_data <- read.csv("Daily_activity_profiles.csv") #Reads a csv file

knitr::kable(head(Daily_activity_profiles_data)) #Displays the first 6 rows of a data frame
```

X	Dec_time	Condition	date	Dec_ZT_time	median	mean	sem
1	0	Genotype_1	2017-06-23	1080	0	0.1428571	0.0991270
2	1	Genotype_1	2017-06-23	1081	0	0.1785714	0.1035555
3	2	Genotype_1	2017-06-23	1082	0	0.5714286	0.3504667
4	3	Genotype_1	2017-06-23	1083	0	0.3928571	0.2319647
5	4	Genotype_1	2017-06-23	1084	0	0.1428571	0.0991270
6	5	Genotype_1	2017-06-23	1085	0	0.0714286	0.0495635

Column description:

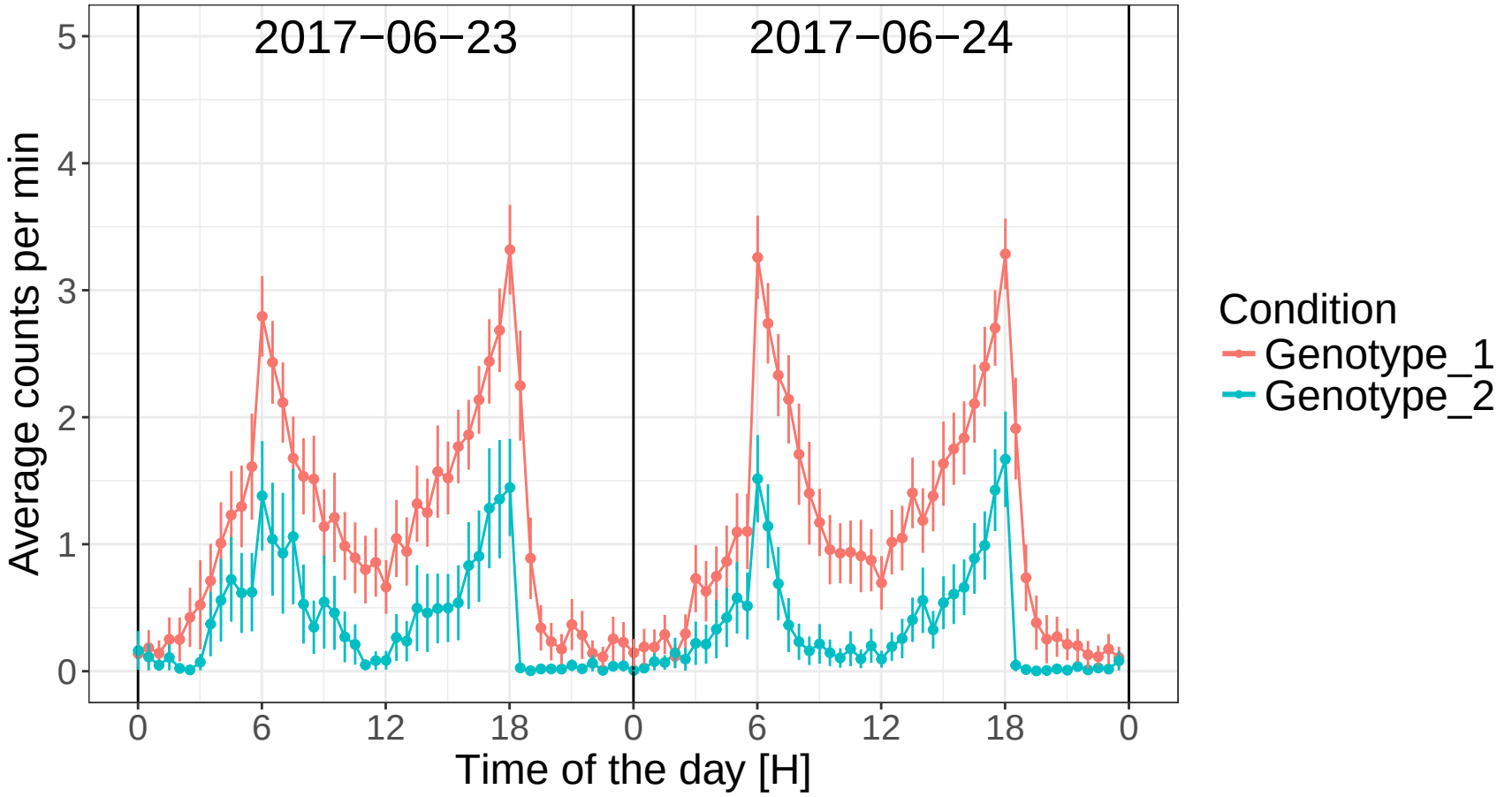
- X - row index
- Dec_time - time in decimal units encoded as a minute of a day, 0 = midnight, 720 = noon
- Condition - a user-defined name of an experimental condition
- date
- Dec_ZT_time - Dec_timetime in Zeitgeber coordinates, 0 = the beginning of the day (light onset time)
- median - median locomotor activity
- mean - mean locomotor activity [number of counts] of individuals in a condition
- sem - SEM of the mean

Daily Locomotor Activity Profiles

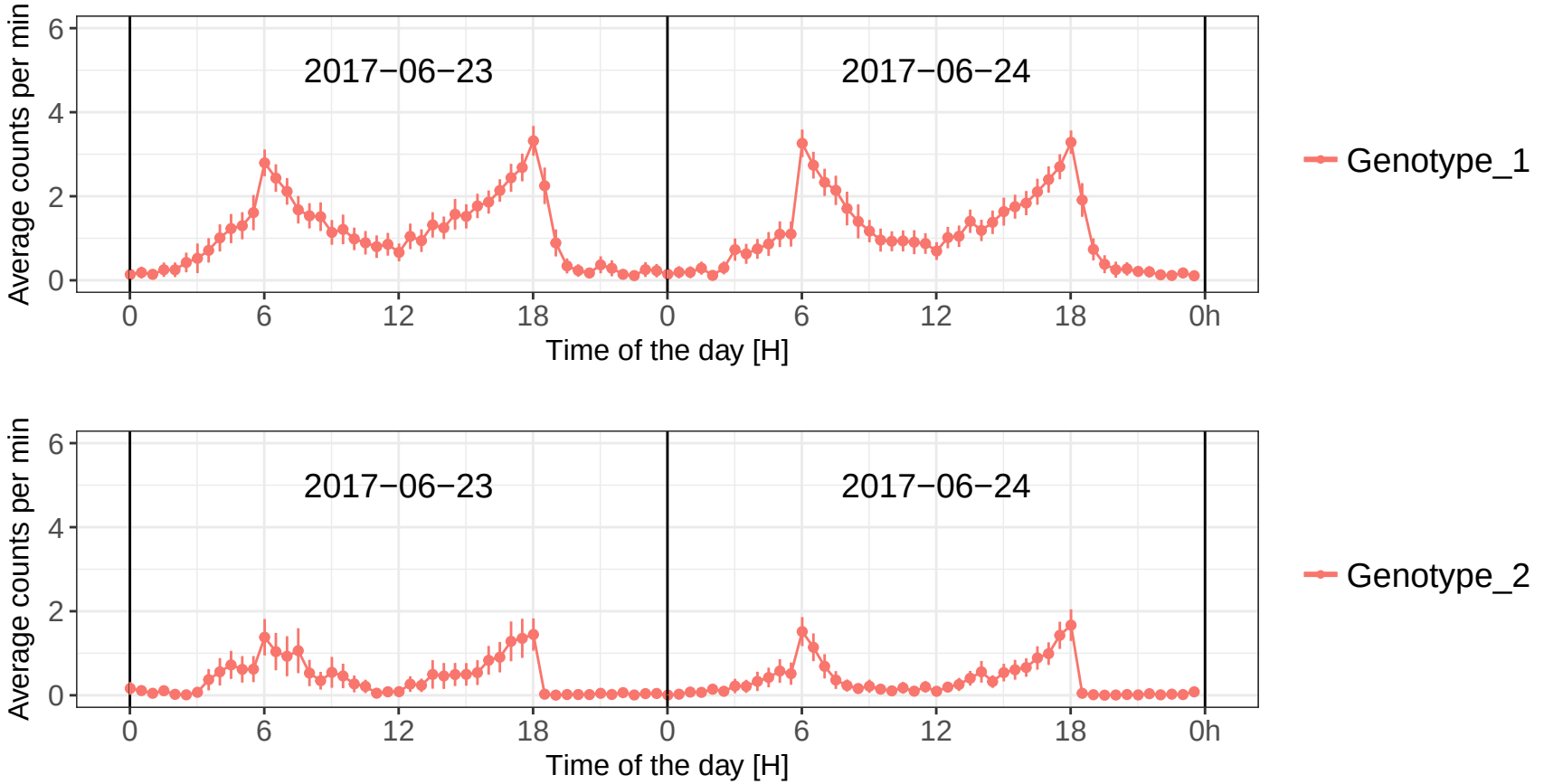
```
# Daily_activity_profiles.csv file a user downloads has a default time resolution of a detaset.
# Using the code below a user can bin the data to any desired resolution, or even edit the binning function parameters.

# Sets parameters for the binning function and the plot
act_profile_window <- 30   # data will be binned into bins of 30 min
data_recording_frequency <- 1 # dataset recording frequency is 1 min
binning_value <- act_profile_window / data_recording_frequency # value for the binning window depends on the...
```

```r
                                                    # ...desired binning length and the data resolution
ac_profile_y_lim <- 5 # sets the X axis limit for the plot


# Filters only the first two days of the data to make a smaller demo plot
    df_mm <- filter(Daily_activity_profiles_data, date %in% as.character(unique(Daily_activity_profiles_data$date)[1:2]))

#  Binning process
    df_mm$Order_column <- rep(c(1:nrow(filter(df_mm, Condition == df_mm$Condition[1]))),
                              length(unique(df_mm$Condition)))


    df_mm$binned_mean <- rep(rollapply(df_mm$mean, width = binning_value, by = binning_value,
                                       FUN = mean, align = "left"), each=binning_value)
    df_mm$binned_sem <- rep(rollapply(df_mm$sem, width = binning_value, by = binning_value,
                                      FUN = mean, align = "left"), each=binning_value)

    df_mm$Condition <- factor(df_mm$Condition, levels = unique(df_mm$Condition))
    df_mm <- arrange(df_mm, Condition)


    df_mm2 <- df_mm[seq(1, nrow(df_mm), by = binning_value),] #Reduces the number of rows after binning

    df_mm2$Condition <- factor(df_mm2$Condition, levels = unique(df_mm$Condition))
    df_mm2 <- arrange(df_mm2, Condition)

# Generates vectors of brake points for plot annotation
    b1 <- seq(0,length(unique(df_mm$Order_column)), (1440/4)/data_recording_frequency)
    b2 <-  seq(0, length(unique(df_mm$Order_column)), b1[3])
    b2 <- as.vector(b2)
    b2 <- b2[1:length(unique(df_mm$date))]
    b2 <- b2*2
    b2 <- b2 + b1[3]

# Plotting
    act_profile_by_day <- ggplot(df_mm2, aes(x=Order_column, y=binned_mean, ymax=3, ymin=0, colour = Condition)) +
      geom_line()+
      geom_point()+
```

```
    theme_bw()+
    scale_x_continuous(breaks = b1,
                       labels=c(rep(c("0", "6", "12", "18"), times=length(unique(df_mm$date))), "0"))+ # X axis labels
    geom_vline(xintercept = seq(0,length(unique(df_mm$Order_column)), 1440/data_recording_frequency))+
        annotate("text", x= b2, y=ac_profile_y_lim, label= unique(df_mm$date), size = 7)+
    # displays day annotations
    labs(title= "", x= "Time of the day [H]", y = "Average counts per min")+
    coord_cartesian(ylim=c(0,as.numeric(5)))+
    theme(legend.text=element_text(size=18))+
    theme(legend.title = element_text(size=18))+
    theme(axis.text.x=element_text(hjust=0.5, size=15))+
    theme(axis.text.y=element_text(size=15))+
    theme(axis.title=element_text(size=18))+
    guides(colour = guide_legend(override.aes = list(size=1))) +
    geom_errorbar(aes(ymax=binned_mean+binned_sem ,ymin=binned_mean-binned_sem), width=0.3)


act_profile_by_day
```


2017−06−23
2017−06−24
Average counts per min
0
1
2
3
4
5
0
6
12
18
0
6
12
18
0
Time of the day [H]
Condition
Genotype_1
Genotype_2

```r
# Daily activity profiles split into separate plots of each condition

# Please notice that a plot is specified as a function that is called for each unique condition via lapply function
  plots_activity_x <- lapply(unique(df_mm$Condition), function(x)
      ggplot(filter(df_mm2, Condition==x), aes(x=Order_column, y=binned_mean, ymax=3, ymin=0, colour= Condition)) +
        theme_bw()+
        geom_line()+
        geom_point()+
        scale_x_continuous(breaks = b1,
                           labels=c(rep(c("0", "6", "12", "18"), times=length(unique(df_mm$date))), "0h"))+
        annotate("text", x= b2, y=ac_profile_y_lim, label= unique(df_mm$date), size = 5)+
        labs(title= "", x= "Time of the day [H]", y = "Average counts per min")+
        coord_cartesian(ylim=c(0,as.numeric(6)))+
        theme(legend.text=element_text(size=14))+
        theme(legend.title = element_text(size=0))+
        theme(axis.text.x=element_text(hjust=0.5, size=12))+
        theme(axis.text.y=element_text(size=12))+
        theme(axis.title=element_text(size=12))+
        guides(colour = guide_legend(override.aes = list(size=1)))+ #edits the point size in a legend
        geom_vline(xintercept = seq(0,length(unique(df_mm$Order_column)), 1440/data_recording_frequency)) +
            geom_errorbar(aes(ymax=binned_mean+binned_sem ,ymin=binned_mean-binned_sem), width=0.3)
          )

# Plots are assembled into one panel with marrangeGrob
    marrangeGrob(plots_activity_x, ncol=1, nrow = length(unique(df_mm$Condition)), top ="")
```


Average counts per min
6
4
2
0
2017−06−23
2017−06−24
0
6
12
18
0
6
12
18
0h
Time of the day [H]
Genotype_1
Average counts per min
6
4
2
0
2017−06−23
2017−06−24
0
6
12
18
0
6
12
18
0h
Time of the day [H]
Genotype_2

Average Activity Profiles in LD

```
Average_activity_profiles_in_LD_data <- read.csv("Average_activity_profiles_in_LD.csv") #Reads a csv file

knitr::kable(head(Average_activity_profiles_in_LD_data)) #Displays the first 6 rows of a data frame
```

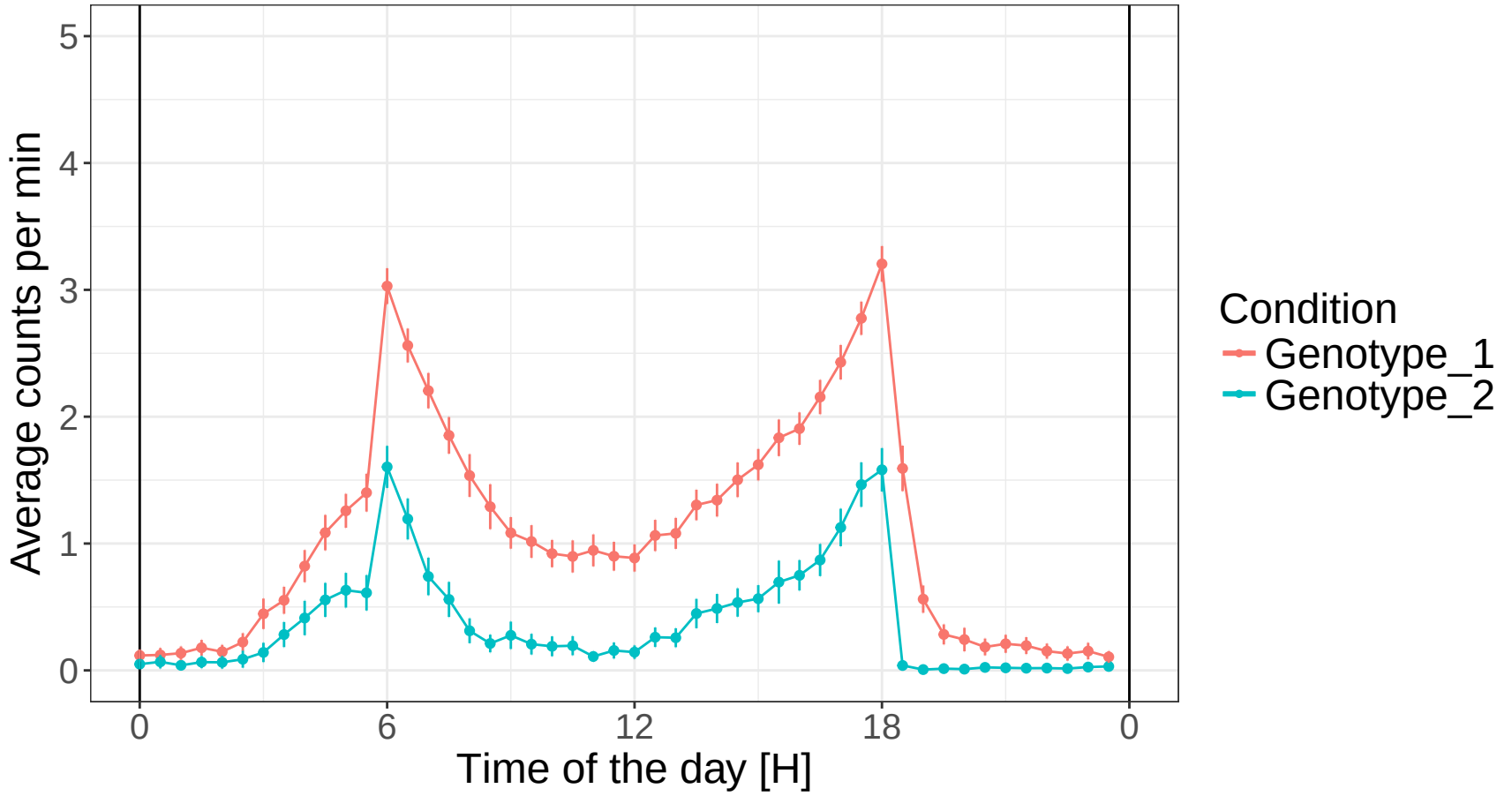
X	Dec_time	Dec_ZT_time	Condition	binned_mean	binned_sem
1	0	1080	Genotype_1	0.1176190	0.0475512
31	30	1110	Genotype_1	0.1216667	0.0551927
61	60	1140	Genotype_1	0.1357143	0.0531906
91	90	1170	Genotype_1	0.1783333	0.0621312
121	120	1200	Genotype_1	0.1469048	0.0548230
151	150	1230	Genotype_1	0.2228571	0.0708161

Column description:

- X - row index
- Dec_time - time in decimal units encoded as a minute of a day, 0 = midnight, 720 = noon
- Dec_ZT_time - Dec_timetime in Zeitgeber coordinates, 0 = the beginning of the day (light onset time)
- Condition - a user-defined name of an experimental condition
- date
- binned_mean - average locomotor activity of individuals in a bin length
- binned_sem - SEM of the mean

```
# Average_activity_profiles_in_LD.csv are already binned

    plots_average_act_profile <-  ggplot(Average_activity_profiles_in_LD_data,
                                         aes(get("Dec_time"), y=binned_mean, ymax=3, ymin=0, colour=Condition)) +
      geom_point()+
      geom_line()+
      labs(title= "", x= "Time of the day [H]", y = "Average counts per min")+
      coord_cartesian(ylim=c(0, 5))+
      scale_x_continuous(breaks = c(0, 360, 720, 1080, 1440), labels=c("0", "6", "12", "18", "0"))+
      geom_vline(xintercept = c(0, 1440))+
      theme_bw()+
      theme(legend.text=element_text(size=18))+
```

```
    theme(legend.title = element_text(size=18))+
    theme(axis.text.x=element_text(hjust=0.5, size=15))+
    theme(axis.text.y=element_text(size=15))+
    theme(axis.title=element_text(size=18))+
    guides(colour = guide_legend(override.aes = list(size=1)))+
    geom_errorbar(aes(ymax=binned_mean+binned_sem,ymin=binned_mean-binned_sem), width=0.3)


plots_average_act_profile
```


Average counts per min
0
1
2
3
4
5
0
6
12
18
0
Time of the day [H]
Condition
Genotype_1
Genotype_2

Sleep Analysis

Daily sleep profiles

```
Sleep_profiles_data <- read.csv("Daily_sleep_profiles.csv") #Reads a csv file

knitr::kable(head(Sleep_profiles_data)) #Displays the first 6 rows of a data frame
```

X	Dec_time	Condition	date	Dec_ZT_time	mean_of_sleep_counts	sem
1	0	Genotype_1	2017-06-23	1080	0.8571429	0.0673435
2	1	Genotype_1	2017-06-23	1081	0.8571429	0.0673435
3	2	Genotype_1	2017-06-23	1082	0.8571429	0.0673435
4	3	Genotype_1	2017-06-23	1083	0.8928571	0.0595238
5	4	Genotype_1	2017-06-23	1084	0.8928571	0.0595238
6	5	Genotype_1	2017-06-23	1085	0.8571429	0.0673435

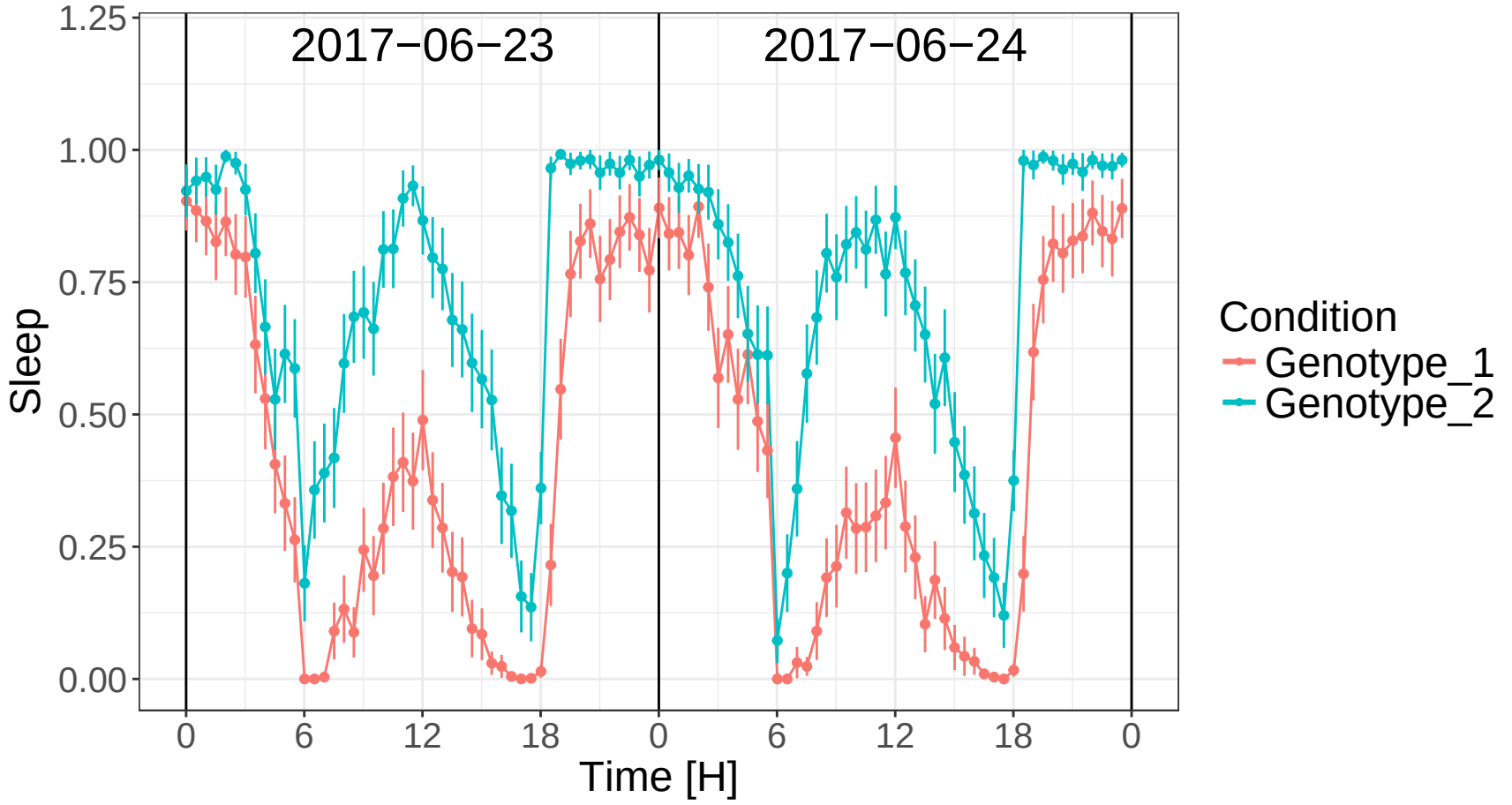
Column description:

- X - row index
- Dec_time - time in decimal units encoded as a minute of a day, 0 = midnight, 720 = noon
- Condition - a user-defined name of an experimental condition
- date
- Dec_ZT_time - Dec_time in Zeitgeber coordinates, 0 = the beginning of the day (light onset time)
- mean_of_sleep_counts - average value of sleep in all living individuals in a condition. 1 = sleep, 0 = activity
- sem - SEM of the mean

```
# Daily_sleep_profiles.csv file has a default time resolution of a detaset.
# Using the code below a user can bin the data to any desired resolution, or even edit the binning function parameters.

# Sets parameters for the binning function and the plot
sleep_profile_window <- 30
data_recording_frequency <- 1

ac_profile_y_lim <- 1.2
slp_profile_max_y <- 1.2
```

```r
binning_value <- sleep_profile_window / data_recording_frequency

unique_conditions <- unique(Sleep_profiles_data$Condition)


#Filters only the first two days
    df_sl <- filter(Sleep_profiles_data, date %in% as.character(unique(Sleep_profiles_data$date)[1:2]))

# Binning sleep
    z <- lapply(unique(df_sl$Condition), function(x) {
      pp <- filter(df_sl, Condition == x)
      pp$Order_column <- c(1:nrow(pp))
      binned_sleep <- rep(rollapply(pp$mean_of_sleep_counts, width = binning_value, by = binning_value,
                                    FUN = mean, align = "left"), each=binning_value)
      binned_sem <-   rep(rollapply(pp$sem, width = binning_value, by = binning_value,
                                    FUN = mean, align = "left"), each=binning_value)

      p1 <- pp[1:length(binned_sleep),]
      p1$binned_sleep <- binned_sleep
      p1$binned_sem <- binned_sem
      p1
    }
    )

    binned_sleep <- rbindlist(z)

    ss <- seq(1, as.numeric(nrow(binned_sleep)), by = binning_value)

    df_sl2 <- binned_sleep[ss,] #Reduces the number of rows after binning

    df_sl2$Condition <- factor(df_sl2$Condition, levels = unique_conditions)
    df_sl2 <- arrange(df_sl2, Condition)


# Generates vectors of brake points for plot annotation
    b1 <- seq(0, 1440 *length(unique(df_sl2$date)), 360)

    b2 <-  seq(0, length(unique(binned_sleep$Order_column)), b1[3])
```

```
  b2 <- as.vector(b2)
  b2 <- b2[1:length(unique(binned_sleep$date))]
  b2 <- b2*2
  b2 <- b2 + b1[3]



# Plotting
  sleep_profile_by_day <- ggplot(df_sl2, aes(x=Order_column, y=binned_sleep, ymax=3, ymin=0, colour = Condition)) +
    geom_line()+
    geom_point()+
    theme_bw()+
    geom_vline(xintercept = c(seq(0,(df_sl2$Order_column)[length(unique(df_sl2$Order_column))], 1440), b1[length(b1)])) +
    annotate("text", x=b2,
             y=ac_profile_y_lim, label= unique(df_sl2$date), size = 7)+
    scale_x_continuous(breaks = b1,
                       labels=c(rep(c("0", "6", "12", "18"), times=length(unique(df_sl2$date))), "0"))+
    labs(title= "", x= "Time [H]", y = "Sleep")+
    coord_cartesian(ylim=c(0,as.numeric(slp_profile_max_y)))+
    theme(legend.text=element_text(size=18))+
    theme(legend.title = element_text(size=18))+
    theme(axis.text.x=element_text(hjust=0.5, size=15))+
    theme(axis.text.y=element_text(size=15))+
    theme(axis.title=element_text(size=18))+
    guides(colour = guide_legend(override.aes = list(size=1))) +
    geom_errorbar(aes(ymax=binned_sleep+binned_sem ,ymin=binned_sleep-binned_sem), width=0.3)


  sleep_profile_by_day
```


2017−06−23
2017−06−24
Sleep
1.25
1.00
0.75
0.50
0.25
0.00
0
6
12
18
0
6
12
18
0
Time [H]
Condition
Genotype_1
Genotype_2

Average sleep profiles in LD

```
Average_sleep_profiles_data <- read.csv("Average_sleep_profiles_in_LD.csv") #Reads a csv file

knitr::kable(head(Average_sleep_profiles_data)) #Displays the first 6 rows of a data frame
```

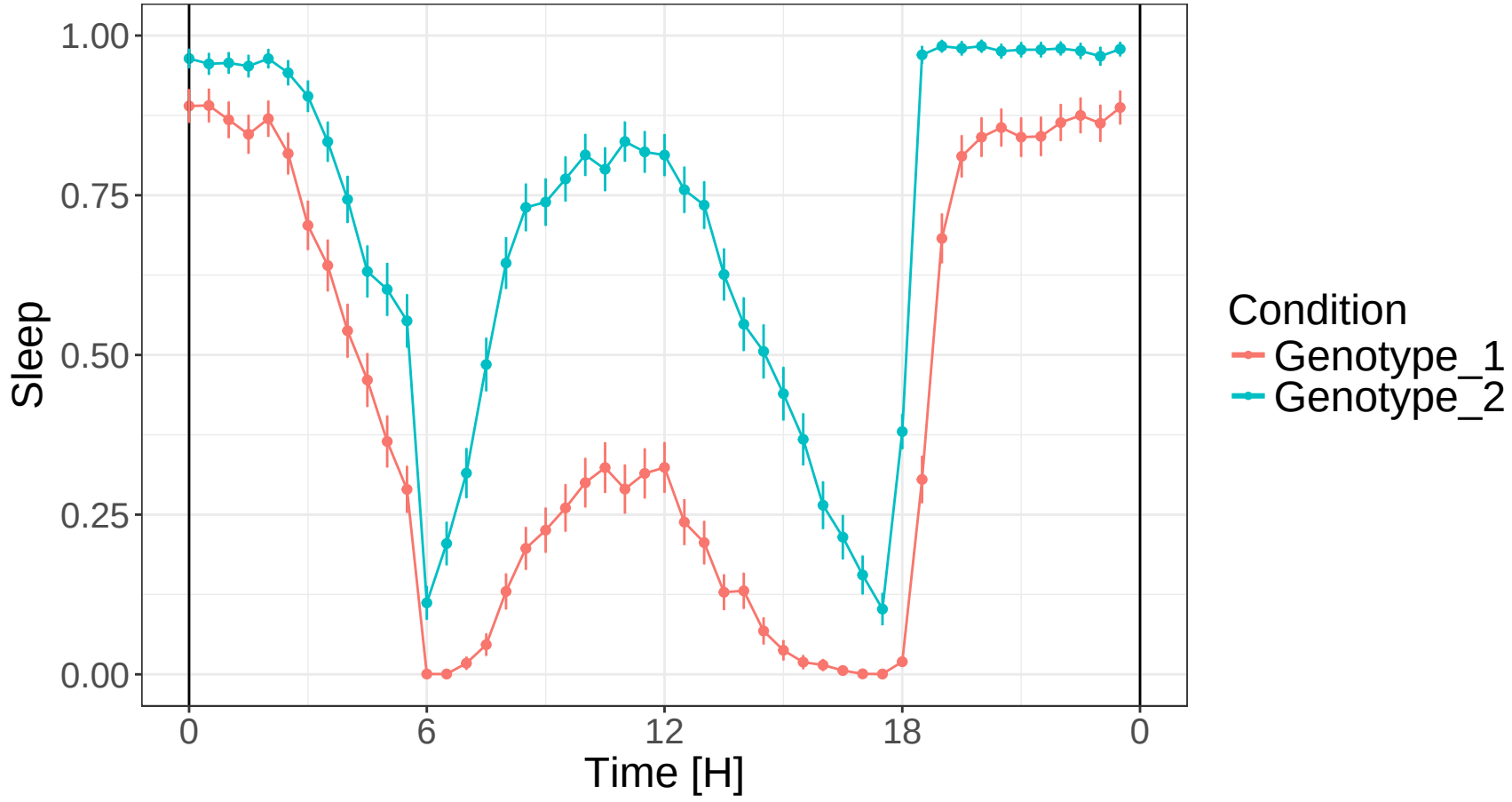
X	Dec_time	Dec_ZT_time	Condition	mean_binned_sleep	sem_binned_sleep
1	0	1080	Genotype_1	0.8897619	0.0264346
2	30	1110	Genotype_1	0.8904762	0.0263946
3	60	1140	Genotype_1	0.8680952	0.0286685
4	90	1170	Genotype_1	0.8454762	0.0305661
5	120	1200	Genotype_1	0.8697619	0.0284495
6	150	1230	Genotype_1	0.8152381	0.0327393

Column description:

- X - row index
- Dec_time - time in decimal units encoded as a minute of a day, 0 = midnight, 720 = noon
- Dec_ZT_time - Dec_timetime in Zeitgeber coordinates, 0 = the beginning of the day (light onset time)
- Condition - a user-defined name of an experimental condition
- mean_binned_sleep - mean daily sleep across all individuals, averaged over a selected bin length
- sem_binned_sleep - SEM of the mean_binned_sleep

```
average_sleep_profile <-    ggplot(Average_sleep_profiles_data, aes(Dec_time, y=mean_binned_sleep, ymax=1,
                                                                    ymin=0, colour=Condition)) +
                            geom_point() +
                            geom_line()+
                            theme_bw()+
        scale_x_continuous(breaks = c(0, 360, 720, 1080, 1440), labels=c("0", "6", "12", "18", "0"))+
        geom_vline(xintercept = c(0, 1440))+
        labs(title= "", x= "Time [H]", y = "Sleep")+
        theme(legend.text=element_text(size=18))+
        theme(legend.title = element_text(size=18))+
        theme(axis.text.x=element_text(hjust=0.5, size=15))+
        theme(axis.text.y=element_text(size=15))+
        theme(axis.title=element_text(size=18))+
```

```
        guides(colour = guide_legend(override.aes = list(size=1))) +
        geom_errorbar(aes(ymax=mean_binned_sleep-sem_binned_sleep,ymin=mean_binned_sleep+sem_binned_sleep), width=0.3)


average_sleep_profile
```



Individual day night sleep in LD

Sleep of individual flies in LD is divided into daytime and nighttime sleep

```
Individual_day_night_sleep <- read.csv("Individual_day_night_sleep.csv") #Reads a csv file

knitr::kable(head(Individual_day_night_sleep)) #Displays the first 6 rows of a data frame
```

X	Channel	Condition	Light_status	mean_sleep_per_ind	sem
1	Monitor61_ch1	Genotype_1	Day	0.0077778	0.0014643
2	Monitor61_ch1	Genotype_1	Night	0.8183333	0.0064271
3	Monitor61_ch10	Genotype_1	Day	0.0883333	0.0047303
4	Monitor61_ch10	Genotype_1	Night	0.8097222	0.0065429
5	Monitor61_ch11	Genotype_1	Day	0.0358333	0.0030983
6	Monitor61_ch11	Genotype_1	Night	0.4591667	0.0083067

Column description:

- X - row index
- Channel - DAM system channel
- Condition - a user-defined name of an experimental condition
- Light_status - Day or Night
- mean_sleep_per_ind - average value of sleep of an individual during the day or night . 1 = sleep, 0 = activity
- sem - sem of the mean_sleep_per_ind. Represents day-to-day variance

Individual sleep and activity bout data in LD

The Individual_sleep_activity_bout_data.csv file contains individual fly data of sleep and activity bout numbers and lengths.

```
Individual_bout_data <- read.csv("Individual_sleep_activity_bout_data.csv") #Reads a csv file

knitr::kable(head(Individual_bout_data)) #Displays the first 6 rows of a data frame
```

X	Channel	Condition	Light_cycle	date	time	ZT_time	Dec_ZT_time	value	sleep_counts	bout	bout_length
1	Monitor61_ch1	Genotype_1	LD	2017-06-23	00:00:00	18:00:00	1080	0	1	1	85
86	Monitor61_ch1	Genotype_1	LD	2017-06-23	01:25:00	19:25:00	1165	3	0	1	5
91	Monitor61_ch1	Genotype_1	LD	2017-06-23	01:30:00	19:30:00	1170	0	1	1	189
280	Monitor61_ch1	Genotype_1	LD	2017-06-23	04:39:00	22:39:00	1359	1	0	1	5
285	Monitor61_ch1	Genotype_1	LD	2017-06-23	04:44:00	22:44:00	1364	0	1	1	3
288	Monitor61_ch1	Genotype_1	LD	2017-06-23	04:47:00	22:47:00	1367	2	0	1	15

Column description:

- X - row index
- Channel - DAM system channel
- Condition - a user-defined name of an experimental condition
- Light_cycle - LD. This analysis is only conducted for LD days
- date
- time - time entries in this file were filtered to only include the starting times of new activity/sleep bouts
- Dec_time - time in decimal units, 0 = midnight, 720 = noon
- Dec_time - time in decimal units, 0 = midnight, 720 = noon
- value - the first raw count value in a bout. Sleep bouts have value = 0. Activity bouts have values > 0.
- sleep_counts - takes values of 1 for sleep, and values of 0 for activity bouts
- bout - this column contains only 1s as it counts all starting bouts. It is useful for counting the number of sleep and activity bouts
- bout_length - bout length in minutes

Actograms

Mean and median actograms

```
Mean_and_median_actogram_data <- read.csv("Mean_and_median_actogram_data.csv") #Reads a csv file

knitr::kable(head(Mean_and_median_actogram_data)) #Displays the first 6 rows of a data frame
```

X	Dec_time	Condition	date	Dec_ZT_time	median	mean	sem
1	0	Genotype_1	2017-06-23	1080	0	0.1428571	0.0991270
2	1	Genotype_1	2017-06-23	1081	0	0.1785714	0.1035555
3	2	Genotype_1	2017-06-23	1082	0	0.5714286	0.3504667
4	3	Genotype_1	2017-06-23	1083	0	0.3928571	0.2319647
5	4	Genotype_1	2017-06-23	1084	0	0.1428571	0.0991270
6	5	Genotype_1	2017-06-23	1085	0	0.0714286	0.0495635

Column description:

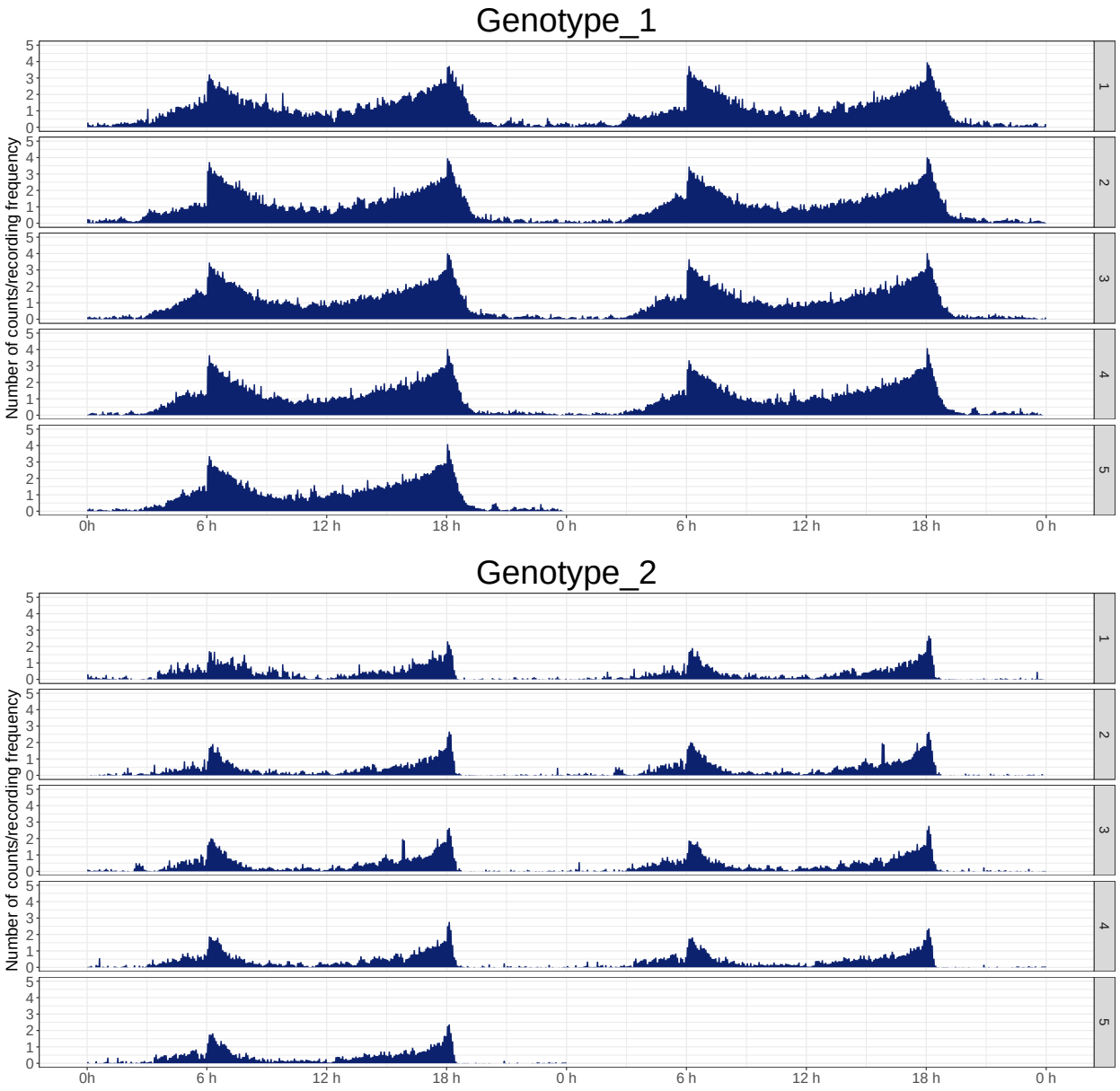
- X - row index
- Dec_time - time in decimal units, 0 = midnight, 720 = noon
- Condition - a user-defined name of an experimental condition
- date
- Dec_ZT_time - Dec_time in Zeitgeber coordinates, 0 = the beginning of the day (light onset time)
- median - median value of locomotor activity in a time point
- mean - mean value of locomotor activity in a time point
- sem - SEM of the mean locomotor activity, represents variance between individuals

```
# Setting plot settings as variables allows to easily edit the plot without searching for parameters deep in the function

actogram_bin <- 5             # Plot bin size [min]
data_recording_frequency <- 1 # Data acquisition frequency [min]
Double_Single <- 'DP'         # Select 'SP' for single plotted actograms, and 'DP' for double plotted
ac_max_counts <- 5            # Max value of counts displayed on an actogram
data_freq <- 1440             # Number of data records in a day, Set to 1440/5 if your DAM system saves counts every 5 min.
mean_or_median_column <- "binned_mean"        # Set to "binned_median" for median actogram
```

```r
# Filters the first two conditions in a dataset to generate a smaller demo plot
mmf <- filter(Mean_and_median_actogram_data, Condition %in% unique(Mean_and_median_actogram_data$Condition)[1:2])

# Function defining mean actogram plot
mean_actogram <- ({function(x){

# Binning actogram mean values - deccreases data resolution for plotting
  y <- filter(mmf, Condition==x) # Filteres a condition
  binning_value <- actogram_bin / data_recording_frequency #Generated a value for binning
  y$binned_mean <- rep(rollapply(y$mean, width = binning_value, by = binning_value,
                                 FUN = mean, align = "left"), each=binning_value)
  y$binned_median <- rep(rollapply(y$median, width = binning_value, by = binning_value,
                                   FUN = mean, align = "left"), each=binning_value)

# Uses a plotting function for a single plotted actogram if Double_Single <- 'SP' is set by user

  if (Double_Single == 'SP') {

      ggplot(y, aes(Dec_time, y=get(mean_or_median_column), ymax=get(mean_or_median_column),
                    ymin=min(get(mean_or_median_column)))) +
        geom_ribbon() +
        facet_grid(date ~ .)+
        geom_ribbon(fill="#0D226E") +
        labs(title= x, x= "", y = "Counts/recording frequency")+
        theme_bw()+
        scale_x_continuous(breaks = c(0, 360,720, 1080, (1440-data_recording_frequency)),
                           labels=c("0 h", "6 h", "12 h", "18 h", "0 h")) +
        theme(axis.text=element_text(size=14))+
        theme(text = element_text(size=16))+
        theme(plot.title = element_text(size = rel(2), hjust=0.5))+
        coord_cartesian(ylim=c(0,as.numeric(ac_max_counts)))

  } else {

# Uses a plotting function for a double plotted actogram if Double_Single <- 'DP' is set by the user
# Well, actually any value other than 'SP' would do it

  # Thsese 3 lines double the actogram data
```

```
    b <- arrange(y, date, Dec_time)
    qqq <- lapply(unique(b$date), function(w) filter(b, date==w))
    zr1 <- do.call("rbind", replicate(2, qqq, simplify = T))

  # Generates a doubled dec time X scale
    zr1$Dec_time_double <- c(rep(1:data_freq, length(unique(zr1$date))),
                             rep((data_freq+1):(data_freq*2), length(unique(zr1$date))))*data_recording_frequency

    zr2 <- arrange(zr1, date, Dec_time_double)
    zr3 <- zr2[(data_freq+1):nrow(zr2),]     #drops the first repeated day
    zr3 <- arrange(zr3, date, Dec_time_double)

  # Generates another doubled dec time X scale after dropping the 1st day
    zr3$Dec_time_double2 <- ((c(rep(1:(data_freq*2), length(unique(zr3$date))))[1:nrow(zr3)])* data_recording_frequency)

  # Generates double days for faceting the actograms
    zr3$date2 <- (c(rep(c(1:length(unique(zr3$date))), each=(data_freq*2)))[1:nrow(zr3)])

  # Plotting
      ggplot(zr3, aes(Dec_time_double2, y=get(mean_or_median_column), ymax=get(mean_or_median_column),
                      ymin=min(get(mean_or_median_column)))) +
        theme_bw()+
        facet_grid(date2 ~ .)+
        geom_ribbon(fill="#0D226E") +
        labs(title= x, x= "", y = "Number of counts/recording frequency")+
        scale_x_continuous(breaks = c(1, 360,720, 1080, 1440, 1800, 2160, 2520, 2880),
                           labels=c("0h", "6 h", "12 h", "18 h", "0 h", "6 h", "12 h", "18 h", "0 h"))+
        coord_cartesian(xlim=c(1,2880))+
        theme(plot.title = element_text(size = rel(2), hjust=0.5))+
        theme(axis.text=element_text(size=14))+
        theme(text = element_text(size=16))+
        coord_cartesian(ylim=c(0,as.numeric(ac_max_counts)))
    }
}
})

# Assembles mean actograms using marrangeGrob
```

```
marrangeGrob(lapply(unique(mmf$Condition), function(x) FUN=mean_actogram(x)),
             ncol=1, nrow = length(unique(mmf$Condition)), top ="")
```


Genotype_1
Genotype_2
Number of counts/recording frequency
0
1
2
3
4
5
0h
6 h
12 h
18 h
0 h
6 h
12 h
18 h
0 h

Individual actograms

```
Individual_actogram_data <- read.csv("Individual_actogram_data.csv") #Reads a csv file

knitr::kable(head(Individual_actogram_data)) #Displays the first 6 rows of a data frame
```

X	day	month	year	time	date	Dec_time	ZT_time	Dec_ZT_time	Light_cycle	variable	value	Condition
1	23	Jun	17	00:00:00	2017-06-23	0	18:00:00	1080	LD	Monitor61_ch1	0	Genotype_1
2	23	Jun	17	00:01:00	2017-06-23	1	18:01:00	1081	LD	Monitor61_ch1	0	Genotype_1
3	23	Jun	17	00:02:00	2017-06-23	2	18:02:00	1082	LD	Monitor61_ch1	0	Genotype_1
4	23	Jun	17	00:03:00	2017-06-23	3	18:03:00	1083	LD	Monitor61_ch1	0	Genotype_1
5	23	Jun	17	00:04:00	2017-06-23	4	18:04:00	1084	LD	Monitor61_ch1	0	Genotype_1
6	23	Jun	17	00:05:00	2017-06-23	5	18:05:00	1085	LD	Monitor61_ch1	0	Genotype_1

Column description:

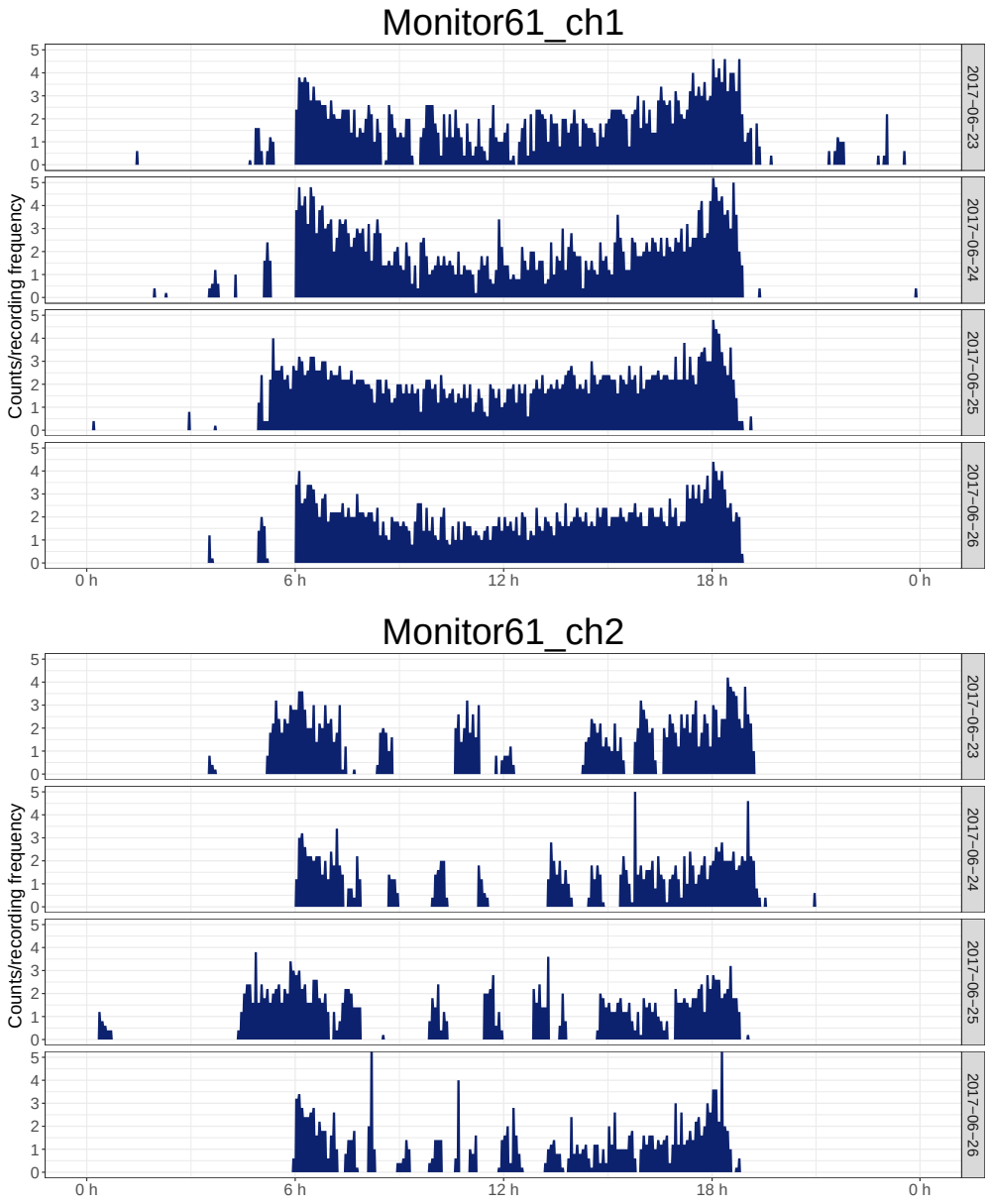
- X - row index
- day - day in a month
- month
- year
- time
- date
- Dec_time - time in decimal units, 0 = midnight, 720 = noon
- ZT_time - Zeitgeber time, 00:00:00 = the beginning of the day (light onset time)
- Dec_ZT_time - Dec_time in Zeitgeber coordinates, 0 = the beginning of the day (light onset time)
- Light_cycle - LD or DD. LD - days with 12:12 light dark cycle, DD - constant darkness
- variable - DAM system channel
- value - raw number of counts
- Condition - a user-defined name of an experimental condition

```
# Setting plot settings as variables allows to easily edit the plot without searching for parameters deep in the function

actogram_bin <- 5             # Plot bin size [min]
data_recording_frequency <- 1 # Data acquisition frequency [min]
Double_Single <- 'SP'         # Select 'SP' for single plotted actograms, and 'DP' for double plotted
ac_max_counts <- 5            # Max value of counts displayed on the actogram
```

```
data_freq <- 1440             # Number of data records in a day, Set to 1440/5 if your DAM system saves counts every 5 min.

binning_value <- actogram_bin / data_recording_frequency #Generated a value for binning

List_of_channels <- read.csv("List_of_alive_flies.csv") # reads a list of DAM system channels of alive flies

# Selects only two names from the list of channels to make a small demo plot
d <- as.character(List_of_channels[1:2,2])

# Function defining infividual actograms
ind_act <- function(x){

 y <- filter(Individual_actogram_data, variable==x) # Filteres an individual DAM channel


# Binning actogram values
  y$binned_value <- rep(rollapply(y$value, width = binning_value, by = binning_value,
                                  FUN = mean, align = "left"), each=binning_value)

  if (Double_Single == 'SP') {

      ggplot(y, aes(Dec_time, y=as.numeric(binned_value), ymax=as.numeric(binned_value),
                    ymin=min(as.numeric(binned_value)))) +
        geom_ribbon(fill="#0D226E")+
        facet_grid(date ~ .)+
        labs(title= x, x= "", y = "Counts/recording frequency")+
        theme_bw()+
        scale_x_continuous(breaks = c(0, 360,720, 1080, (1440-data_recording_frequency)),
                           labels=c("0 h", "6 h", "12 h", "18 h", "0 h")) +
        theme(axis.text=element_text(size=14))+
        theme(text = element_text(size=16))+
        theme(plot.title = element_text(size = rel(2), hjust=0.5))+
        coord_cartesian(ylim=c(0,as.numeric(ac_max_counts)))

  } else {

# Thsese 3 lines double the actogram data
    b <- arrange(y, date, Dec_time)
```

```
    qqq <- lapply(unique(b$date), function(w) filter(b, date==w))
    zr1 <- do.call("rbind", replicate(2, qqq, simplify = T))

    # Generates a doubled dec time X scale
    zr1$Dec_time_double <- c(rep(1:data_freq, length(unique(zr1$date))),
                             rep((data_freq+1):(data_freq*2), length(unique(zr1$date))))*data_recording_frequency

    zr2 <- arrange(zr1, date, Dec_time_double)
    zr3 <- zr2[(data_freq+1):nrow(zr2),] #drops the first repeated day
    zr3 <- arrange(zr3, date, Dec_time_double)

# Generates another doubled dec time X scale after dropping the 1st day
    zr3$Dec_time_double2 <- ((c(rep(1:(data_freq*2),
                                    length(unique(zr3$date))))[1:nrow(zr3)])*data_recording_frequency)

# Generates double days for faceting the actograms
    zr3$date2 <- (c(rep(c(1:length(unique(zr3$date))), each=(data_freq*2)))[1:nrow(zr3)])


      ggplot(zr3, aes(Dec_time_double2, y=binned_value, ymax=binned_value, ymin=min(binned_value))) +
        geom_ribbon(fill="#0D226E")+
        theme_bw()+
        facet_grid(date2 ~ .)+
        labs(title= x, x= "", y = "Number of counts/recording frequency")+
        scale_x_continuous(breaks = c(1, 360,720, 1080, 1440, 1800, 2160, 2520, 2880),
                           labels=c("0 h", "6 h", "12 h", "18 h", "0 h", "6 h", "12 h", "18 h", "0 h"))+
        coord_cartesian(xlim=c(1,2880))+
        theme(plot.title = element_text(size = rel(2), hjust=0.5))+
        theme(axis.text=element_text(size=14))+
        theme(text = element_text(size=16))+
        coord_cartesian(ylim=c(0,as.numeric(ac_max_counts)))

  }
}


# Assembles individual actograms using marrangeGrob
```

```
marrangeGrob(lapply(d, function(x) FUN=ind_act(x)), ncol=1, nrow = length(d), top ="")
```


Monitor61_ch1
Counts/recording frequency
2017-06-23
2017-06-24
2017-06-25
2017-06-26
0 h
6 h
12 h
18 h
0 h
Monitor61_ch2
Counts/recording frequency
2017-06-23
2017-06-24
2017-06-25
2017-06-26
0 h
6 h
12 h
18 h
0 h

Circadian Period Analysis

Mean periodograms

```
Mean_period_by_condition_rhythmic <- read.csv("Mean_periodograms_data.csv") #Reads a csv file

knitr::kable(head(Mean_period_by_condition_rhythmic)) #Displays the first 6 rows of a data frame
```

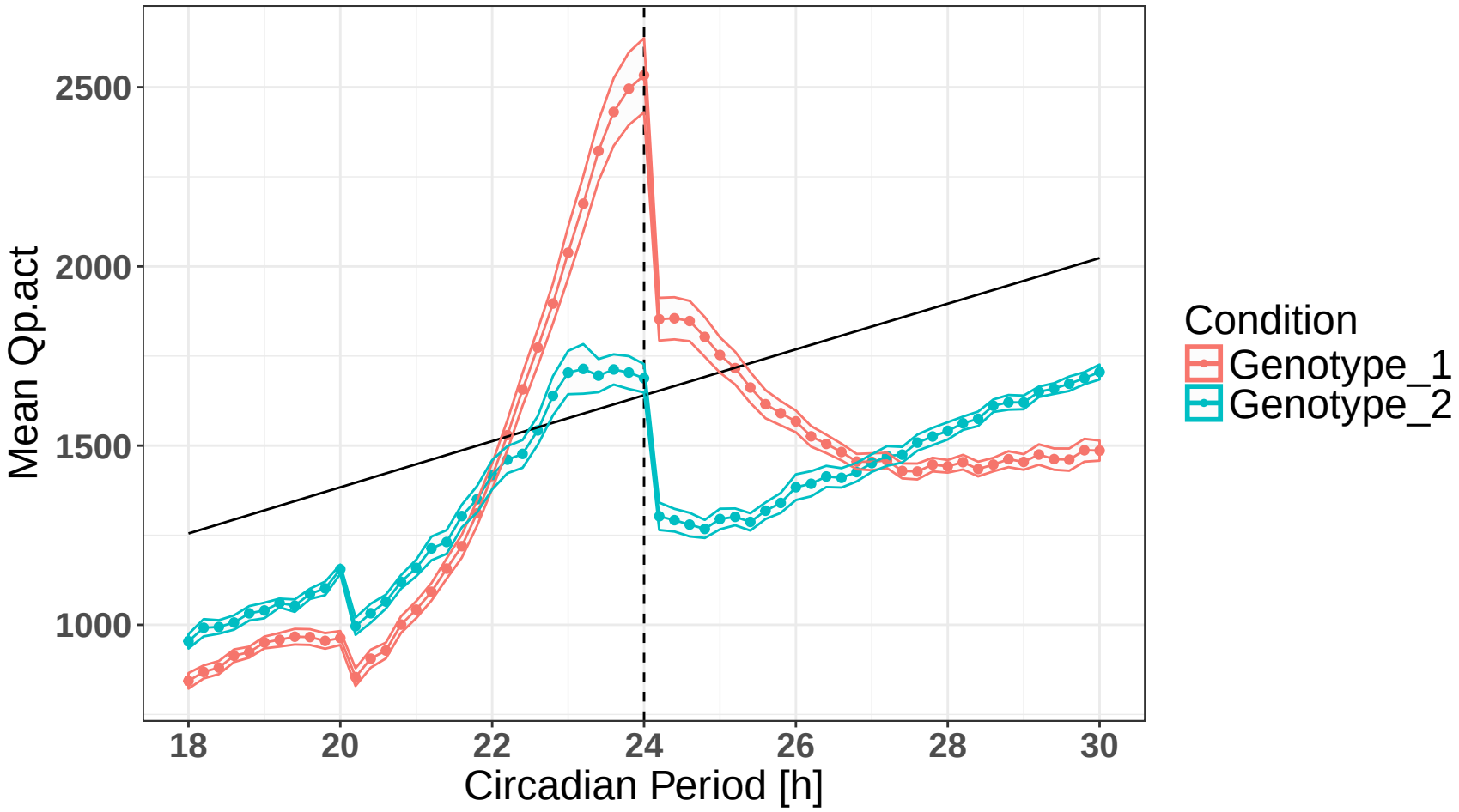
X	Condition	Period	Mean_Qp.act	SEM_Qp.act	N_of_rhythmic_flies	Mean_Qp.sig	Mean_Qp.act_Qp.sig_ratio	SEM_Qp.act_Qp.sig_ratio
1	Genotype_1	18.0	843.8173	21.75037	28	1254.870	0.6724343	0.0173328
2	Genotype_1	18.2	868.6801	18.20022	28	1267.794	0.6851903	0.0143558
3	Genotype_1	18.4	880.8313	18.28159	28	1280.713	0.6877662	0.0142745
4	Genotype_1	18.6	913.7706	17.55817	28	1293.628	0.7063630	0.0135728
5	Genotype_1	18.8	923.7290	15.35431	28	1306.537	0.7070056	0.0117519
6	Genotype_1	19.0	950.7352	16.43003	28	1319.442	0.7205588	0.0124523

Column description:

- X - row index
- Condition - a user-defined name of an experimental condition
- Period - period tested by the Chi-Square algorithm [hours]
- Mean_Qp.act - Mean Chi-Square Qp.act value across all rhythmic individuals in a condition
- SEM_Qp.act - SEM of the Mean_Qp.act
- N_of_rhythmic_flies - number of rhythmic flies passing the Qp.act/Qp.sig threshold
- Mean_Qp.sig - Chi-Square Qp.sig period significance threshold
- Mean_Qp.act_Qp.sig_ratio - Ratio of Mean_Qp.act / Mean_Qp.sig. Values used for calling the strongest peak.
- SEM_Qp.act_Qp.sig_ratio - SEM of the Mean_Qp.act_Qp.sig_ratio

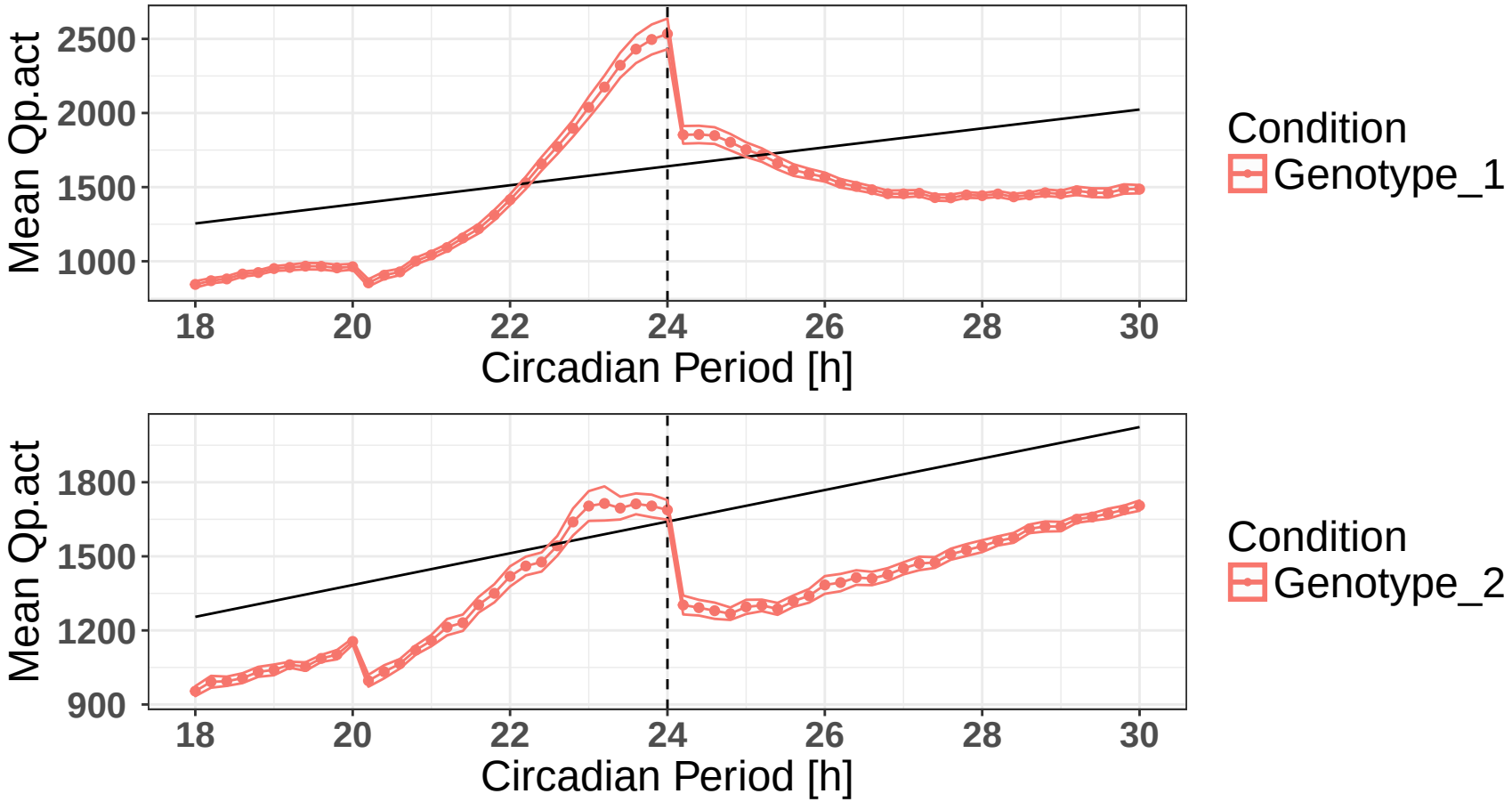
```
#Overlapping mean periodograms

 Mean_periodograms_plot_overlapping <- ggplot(Mean_period_by_condition_rhythmic,
                                              aes(x=Period, y=Mean_Qp.act, colour = Condition)) +
      geom_point()+
      geom_line()+
      geom_line(data=Mean_period_by_condition_rhythmic, aes(x=Period, y = Mean_Qp.sig), size=0.5, colour="black")+
      theme_bw()+
```

```
      labs(x="Circadian Period [h]", y="Mean Qp.act")+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      scale_x_continuous(breaks = seq(4, 56, 2)) +
      geom_vline(xintercept = 24, linetype = 2 ) +
      theme(legend.text=element_text(size=18))+
      theme(legend.title = element_text(size=18))+
      theme(axis.text.x=element_text(hjust=0.5, size=15, face="bold"))+
      theme(axis.text.y=element_text(hjust=0.5, size=15, face="bold"))+
      theme(axis.title=element_text(size=18))+
      guides(colour = guide_legend(override.aes = list(size=1))) +
      geom_ribbon(aes(ymin=Mean_Qp.act-SEM_Qp.act, ymax=Mean_Qp.act+SEM_Qp.act), linetype=1, alpha=0.01)

      #scale_colour_manual(values=c("blue", "red")) # Add this line to customize colors


Mean_periodograms_plot_overlapping
```




```
#Mean periodogram of individual conditions

unique_conditions <- unique(Mean_period_by_condition_rhythmic$Condition) # stors names of all unique experimental conditions

#Defines a plot function executed for all unique conditions
plot_x <- lapply(unique_conditions, function(x)
      ggplot(filter(Mean_period_by_condition_rhythmic, Condition==x), aes(x=Period, y=Mean_Qp.act, colour = Condition)) +
      geom_point()+
```

```
      geom_line()+
      geom_line(data=Mean_period_by_condition_rhythmic, aes(x=Period, y = Mean_Qp.sig), size=0.5, colour="black")+
      theme_bw()+
      labs(x="Circadian Period [h]", y="Mean Qp.act")+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      scale_x_continuous(breaks = seq(4, 56, 2)) +
      geom_vline(xintercept = 24, linetype = 2 ) +
      theme(legend.text=element_text(size=18))+
      theme(legend.title = element_text(size=18))+
      theme(axis.text.x=element_text(hjust=0.5, size=15, face="bold"))+
      theme(axis.text.y=element_text(hjust=0.5, size=15, face="bold"))+
      theme(axis.title=element_text(size=18))+
      guides(colour = guide_legend(override.aes = list(size=1)))+
      geom_ribbon(aes(ymin=Mean_Qp.act-SEM_Qp.act, ymax=Mean_Qp.act+SEM_Qp.act), linetype=1, alpha=0.01)
                  )
#Executes the ploting function and grupes plots into a figure using marrangeGrob
  Mean_periodograms_plot_split <- marrangeGrob(plot_x, ncol=1, nrow = length(unique_conditions), top ="")

  Mean_periodograms_plot_split
```


Mean Qp.act
2500
2000
1500
1000
18
20
22
24
26
28
30
Circadian Period [h]
Condition
Genotype_1
Mean Qp.act
1800
1500
1200
900
18
20
22
24
26
28
30
Circadian Period [h]
Condition
Genotype_2

Individual periodograms

```
Indivdual_periodograms_data <- read.csv("Individual_periodograms_rhythmic_alive.csv")

knitr::kable(head(Indivdual_periodograms_data))
```

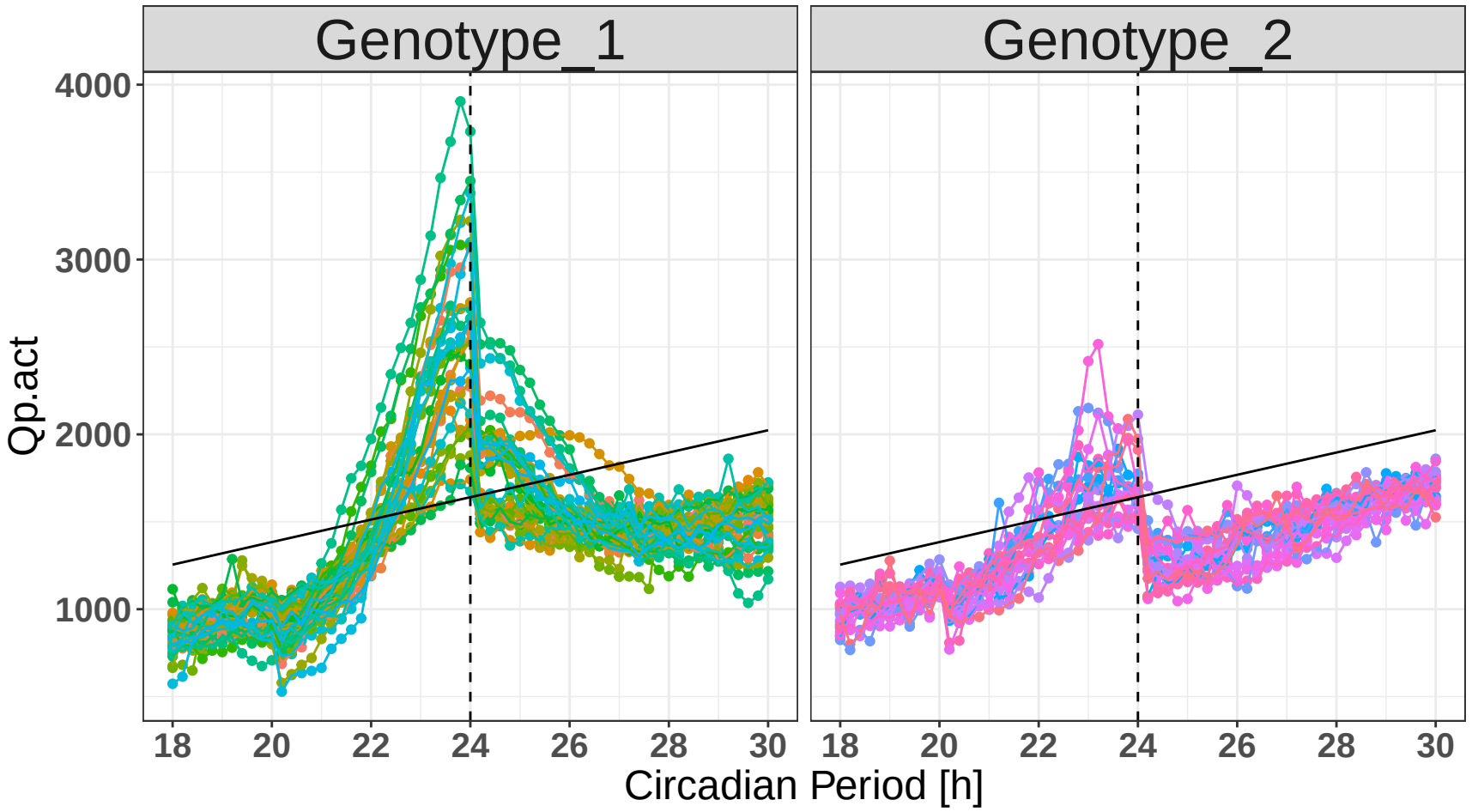
X	Condition	channel	Period	Qp.act	Qp.sig	Act_Sig_ratio	Condition_channel
1	Genotype_1	Monitor61_ch1	18.0	827.1517	1254.870	0.6591535	Genotype_1_Monitor61_ch1
2	Genotype_1	Monitor61_ch1	18.2	845.1306	1267.794	0.6666151	Genotype_1_Monitor61_ch1
3	Genotype_1	Monitor61_ch1	18.4	934.0897	1280.713	0.7293511	Genotype_1_Monitor61_ch1
4	Genotype_1	Monitor61_ch1	18.6	978.6343	1293.628	0.7565039	Genotype_1_Monitor61_ch1
5	Genotype_1	Monitor61_ch1	18.8	934.8861	1306.537	0.7155451	Genotype_1_Monitor61_ch1
6	Genotype_1	Monitor61_ch1	19.0	995.7385	1319.442	0.7546666	Genotype_1_Monitor61_ch1

Column description:

- X - row index
- Condition - a user-defined name of an experimental condition
- channel - DAM system channel
- Period - period tested by the Chi-Square algorithm [hours]
- Qp.act - Chi-Square algorithm Qp.act value
- Qp.sig - Chi-Square algorithm Qp.sig value
- Act_Sig_ratio - Qp.act / Qp.sig ratio. Values are used for determining most significant peak
- Condition_channel - Condition and channel columns spliced together. These names are used by the plotting function for labeling

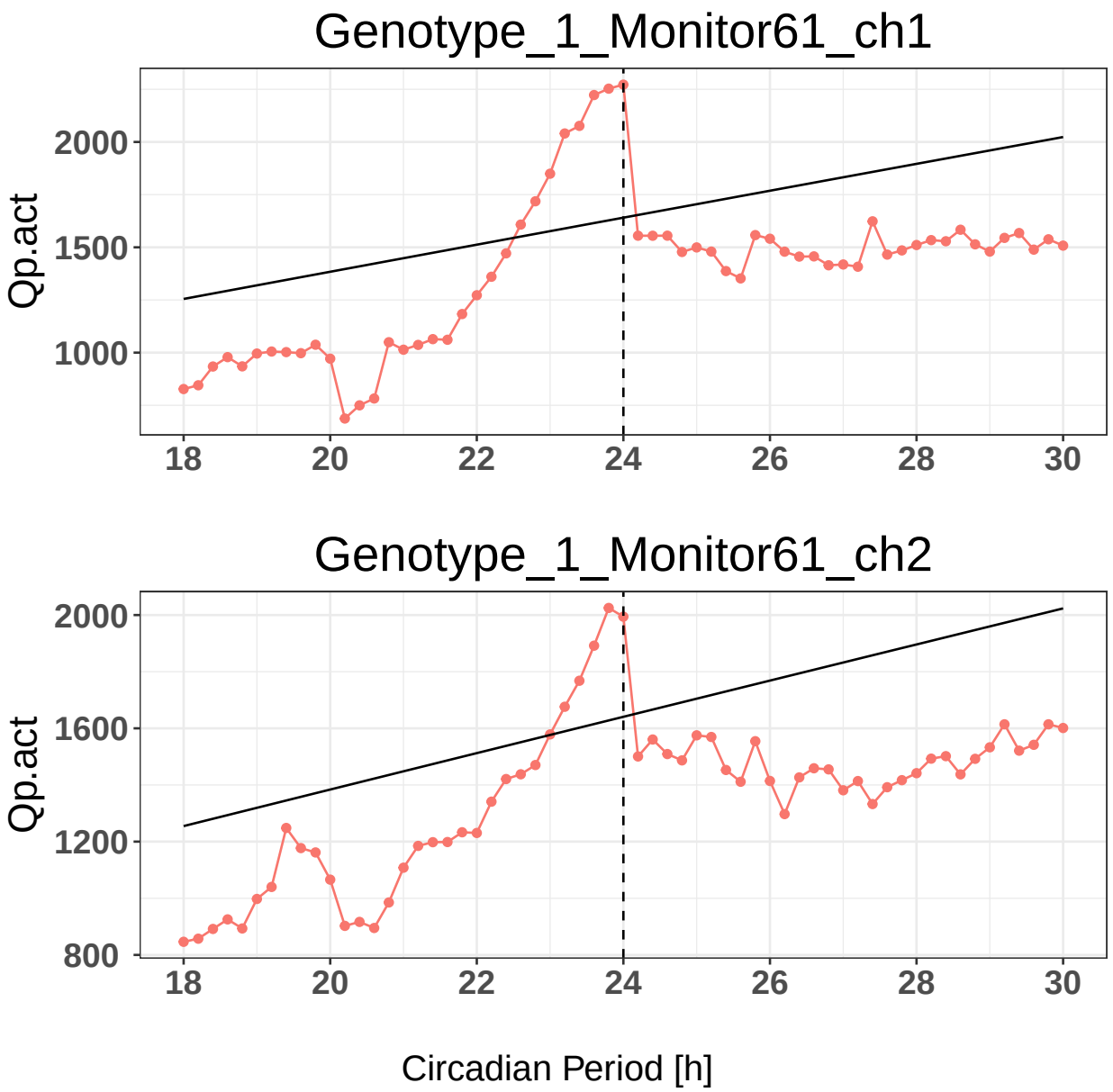
```
#Indivdual periodograms gruped by condition

 Indivdual_periodograms_gruped_by_condition_plot <-  ggplot(Indivdual_periodograms_data,
                                                           aes(x=Period, y=Qp.act, colour = channel)) +
      geom_point()+
      geom_line()+
      geom_line(data= Indivdual_periodograms_data, aes(x=Period, y = Qp.sig), size=0.5, colour="black")+
      theme_bw()+
      labs(x="Circadian Period [h]", y="Qp.act")+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      scale_x_continuous(breaks = seq(4, 56, 2)) +
```

```
    geom_vline(xintercept = 24, linetype = 2 ) +
    theme(axis.text.x=element_text( hjust=0.5, size=15, face="bold"))+
    theme(axis.text.y=element_text( hjust=0.5, size=15, face="bold"))+
    theme(axis.title = element_text(size=18))+
    theme(legend.title = element_blank())+
    theme(legend.position="none")+
    theme(strip.text = element_text(size=25))+
    facet_wrap(~ Condition)

Indivdual_periodograms_gruped_by_condition_plot
```




```
#Indivdual periodograms

  #Defines a function plotting individual periodograms
    ind_periodogram <- function(x){
      y <- filter(Indivdual_periodograms_data, Condition_channel==x) # Filteres a condition

      ggplot(y, aes(x=Period, y=Qp.act, colour = channel)) +
        geom_point()+
```

```
        geom_line()+
        geom_line(data= y, aes(x=Period, y = Qp.sig), size=0.5, colour="black")+
        theme_bw()+
        labs(title= x, x="", y="Qp.act")+
        theme(plot.title = element_text(size = rel(2), hjust=0.5))+
        scale_x_continuous(breaks = seq(4, 56, 2)) +
        geom_vline(xintercept = 24, linetype = 2 ) +
        theme(axis.text.x=element_text( hjust=0.5, size=15, face="bold"))+
        theme(axis.text.y=element_text( hjust=0.5, size=15, face="bold"))+
        theme(axis.title = element_text(size=18))+
        theme(legend.title = element_blank())+
        scale_fill_manual(values=Plot_colors())+
        theme(legend.position="none")+
        theme(strip.text = element_text(size=25))
    }

    # Defines a list of unique channel.
    #To increase the clarity only the first 2 individuals in that list are selected using the [1:2] operator
    list_of_chan <- unique(Indivdual_periodograms_data$Condition_channel)[1:2]

#Executes the ploting function and grupes plots into a figure
marrangeGrob(lapply(list_of_chan, function(x) FUN=ind_periodogram(x)), ncol=1, nrow = length(list_of_chan),
             top ="",         bottom=textGrob("Circadian Period [h]", gp=gpar(fontsize=16)))
```


Genotype_1_Monitor61_ch1
Qp.act
2000
1500
1000
18
20
22
24
26
28
30
Genotype_1_Monitor61_ch2
Qp.act
2000
1600
1200
800
18
20
22
24
26
28
30
Circadian Period [h]

Period peaks

```
Indivdual_fly_period_peaks_data <- read.csv("Individual_fly_period_peaks.csv")

knitr::kable(head(Indivdual_fly_period_peaks_data))
```

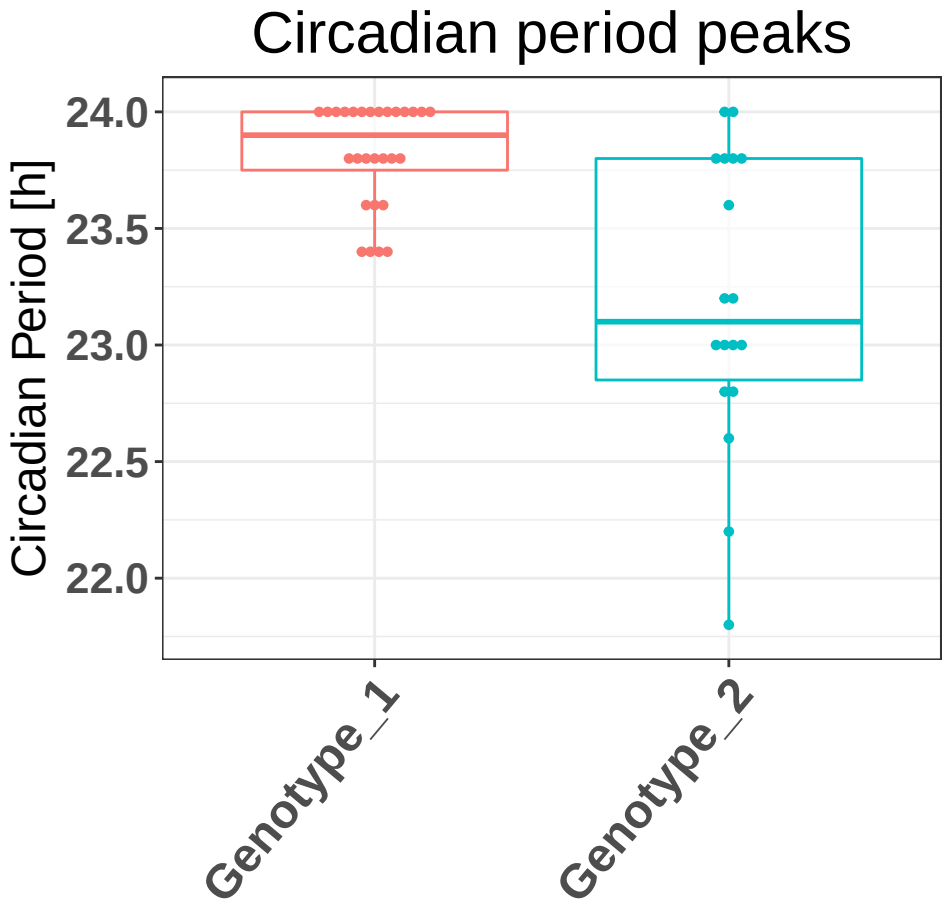
X	Condition	channel	Period	Qp.act	Qp.sig	Act_Sig_ratio
1	Genotype_1	Monitor61_ch1	24.0	2272.101	1640.682	1.384852
2	Genotype_1	Monitor61_ch2	23.8	2024.989	1627.877	1.243945
3	Genotype_1	Monitor61_ch3	23.8	2717.039	1627.877	1.669069
4	Genotype_1	Monitor61_ch5	24.0	3379.208	1640.682	2.059637
5	Genotype_1	Monitor61_ch6	24.0	3098.915	1640.682	1.888797
6	Genotype_1	Monitor61_ch8	24.0	2659.998	1640.682	1.621276

Column description:

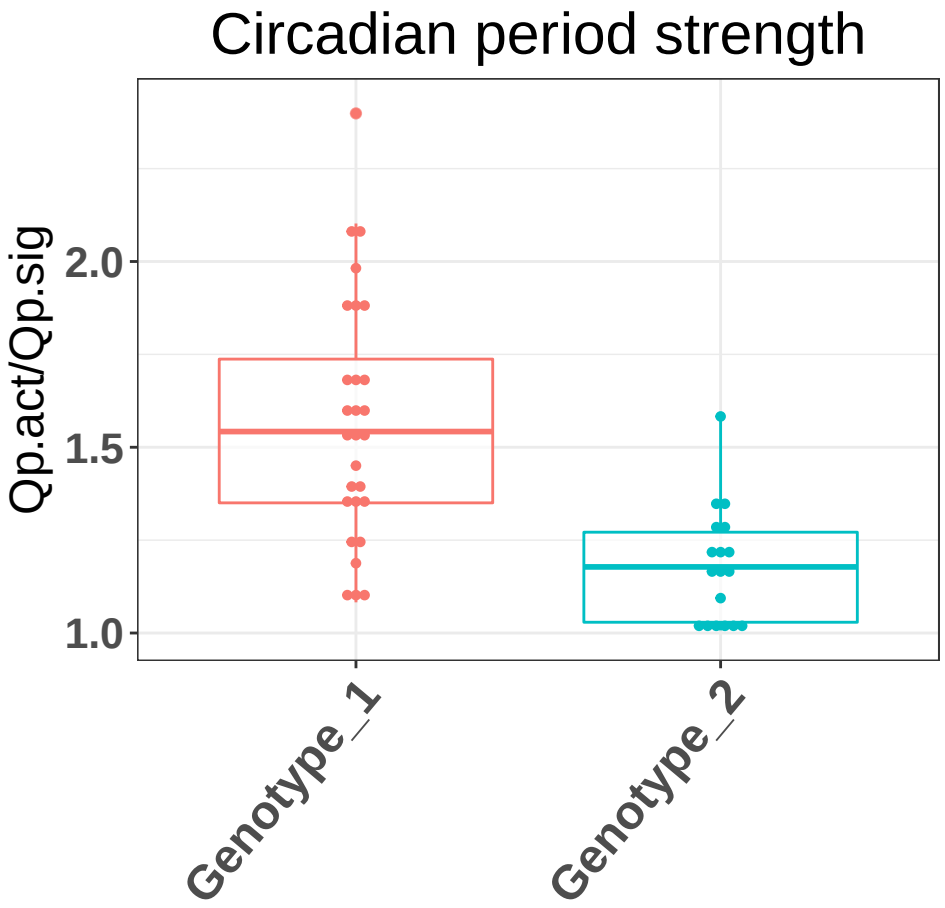
- X - row index
- Condition - a user-defined name of an experimental condition
- channel - DAM system channel
- Period - Chi-Square Period peak [hours] of an individual
- Qp.act - Chi-Square Qp.act period peak value of an individual
- Qp.sig - Chi-Square Qp.sig period peak significance threshold
- Act_Sig_ratio - Qp.act/Qp.sig ratio. Provides information about the period strength

```
# Period peaks boxplot

 ind_periods<- ggplot(na.omit(Indivdual_fly_period_peaks_data), aes(x=Condition, y=Period, colour = Condition)) +
      geom_boxplot(alpha=0.7)+
      geom_dotplot(binaxis = "y", stackdir = "center", dotsize = 0.5, aes(fill=Condition))+
      labs(y="Circadian Period [h]", x="") +  #adds/removes axis lables
      theme(legend.title=element_blank())+ #removes legend title
      theme_bw()+
      theme(axis.text.x=element_text(angle=50, hjust=1, size=18, face="bold"))+
      theme(axis.text.y=element_text(hjust=1, size=16, face="bold"))+
      theme(axis.title.y = element_text(color="black", size=18))+         #axis title
      theme(legend.title = element_blank())+
```

```
      theme(legend.text = element_text(size=18))+
      labs(title= "Circadian period peaks")+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      theme(legend.position="none")

ind_periods
```



```r
# Box plot of individual Period strengths

    ind_strength<- ggplot(na.omit(Indivdual_fly_period_peaks_data), aes(x=Condition, y=Act_Sig_ratio, colour = Condition)) +
      geom_boxplot(alpha=0.7)+
      geom_dotplot(binaxis = "y", stackdir = "center", dotsize = 0.5, aes(fill=Condition))+
      labs(y="Qp.act/Qp.sig", x="") +  #adds/removes axis lables
      theme(legend.title=element_blank())+ #removes legend title
      theme_bw()+
      theme(axis.text.x=element_text(angle=50, hjust=1, size=18, face="bold"))+
      theme(axis.text.y=element_text(hjust=1, size=16, face="bold"))+
      theme(axis.title.y = element_text(color="black", size=18))+          #axis title
      theme(legend.title = element_blank())+
      theme(legend.text = element_text(size=18))+
      labs(title= "Circadian period strength")+
      theme(plot.title = element_text(size = rel(2), hjust=0.5))+
      theme(legend.position="none")

    ind_strength
```


Circadian period strength
Qp.act/Qp.sig
2.0
1.5
1.0
Genotype_1
Genotype_2