Results

Description

In order to illustrate some mechanisms in rethomics, we provide a simple and reproducible example of analysis of circadian phenotype recorded with DAM2 monitors – a widely addopted paradigm.

TODO: * data source – The data was obtained from ... (citation) TODO. * description (genotypes etc) – exhaustive description in the `metadata.csv`

Data loading

First of all, the necessary `rethomics` packages are loaded.

```
library(damr)      # input DAM2 data
library(zeitgebr)  # periodogram computation
library(sleepr)    # sleep analysis
library(ggetho)    # behaviour visualisation
```

Then, the metadata file is read and linked to the `.txt` result files.

```
met <- damr::link_dam2_metadata("./metadata.csv", ".") # linking
dt <- load_dam2(met)                                   # loading
summary(dt)                                            # quick summary
```

```
## behavr table with:
##  58  individuals
##  8   metavariables
##  2   variables
##  1.58722e+05 measurements
##  1   key (id)
```

Preprocessing

We notice, from the metadata, that the two replicates do not have the same time in baseline. We would like to express the time relative to the important event: thetransition to `LL`. The best way is to sustract the `baseline_days` metavariable from the `t` variable. This gives us an opportunity to illustrate the use `xmv()` that maps metavariables as variables. In addition, we use the `data.table` syntax to create, in place, a `moving` variable. It is `TRUE` when and only when `activity` is greater than zero:

```
dt[,t := t - days(xmv(baseline_days))]    # baseline sustraction. not the use of xmv
dt[, moving :=  activity > 0]
```

To simplify visualisation, we create our own `label` metavariable, as combination of a number and `genotype`. In the restricted context of this analysis, this acts a unique identifyier. Importanly, we keep `id` , which is more rigourous and universal.

```
dt[, label := interaction(1:.N, genotype), meta=T]
# print(dt)
```

Curation

It is important to see an overview of how each individual and experiment behaved and, if necessary, alter the data accordingly. We save this figure as Fig 3A.

```
# make a ggplot object with label on the y and moving on the z axis
fig3A <- ggetho(dt, aes(y=label, z=moving)) +
  # show data as a tile plot. That is z is a pixel whose intensity maps moving
  stat_tile_etho() +
  # add layers to draw annotations to show L and D phases as white and black
  # the first layer is for the baseline (until t=0)
  stat_ld_annotations(x_limits = c(dt[,min(t)], 0)) +
  # in the 2nd one, we start at 0 and use grey instead of black as we work in LL
  stat_ld_annotations(x_limits = c(0, dt[,max(t)]), ld_colours = c("white", "grey"))
```

Dead or escaped animals are falsely scored as long series of zero-activity. Our `sleepr` packges offer a tool to detect and remove this artifactual data:

```
dt <- sleepr::curate_dead_animals(dt, moving)
# make a ggplot object with label on the y and moving on the z axis
fig3B <- ggetho(dt, aes(y=label, z=moving)) +
    stat_tile_etho() +
    stat_ld_annotations(x_limits = c(dt[,min(t)], 0)) +
    stat_ld_annotations(x_limits = c(0, dt[,max(t)]), ld_colours = c("white", "grey"))
```

The updated version can be visualised in Fig 3B.

For the purpose of this example, we keep only individuals that have at least five days in LL.

```
valid_dt <- dt[ , .(valid = max(t) > days(5)), by=id]
valid_ids <- valid_dt[valid == T, id]
dt <- dt[id %in% valid_ids]
summary(dt)
```

```
## behavr table with:
##  52  individuals
##  9   metavariables
##  3   variables
##  1.40609e+05 measurements
##  1   key (id)
```

Note that as a result, we now have 52 "valid" individuals.

Double plotted actograms

A common way of representing rythmicity in circadian experiments is to compute "double-plotted actograms". In Fig S1A, we show all double plotted actograms layed out in a grid.

```
figS1A <- ggetho(dt, aes(z = moving), multiplot = 2) +
            stat_bar_tile_etho() +
            facet_wrap( ~ label, ncol=4) +
            scale_y_discrete(name="Day")
```

Periodograms

For each indidual, we compute a χ^2 periodogram and we layout all of them in a grid (Fig S1B).

```
dt_ll <- dt[t > days(1)]
per_dt <- periodogram(moving,
                      dt_ll,
```

```
                        resample_rate = 1/mins(10),
                        FUN=chi_sq_periodogram)

per_dt <- find_peaks(per_dt)

figS1B <- ggperio(per_dt, aes(y = power, peak=peak)) +
      geom_line() +
      geom_line(aes(y=signif_threshold), colour="red") +
      geom_peak() +
      facet_wrap( ~ label, ncol=4)
```

```
## pdf
##   2
```

```
## pdf
##   2
```