1 Installation

`rethomics` is still under heavy development, so it cannot be uploaded to the CRAN (Comprehensive R Archive Network) – where most stable packages are – yet. Instead, we can install R using Hadley Wickham's popular `devtools`[1]. Once you have `devtools` installed, it should be straightforward to install `rethomics`

```
> library(devtools)
> install_github("gilestrolab/rethomics",subdir = "rethomics")
```

Check for error messages. Then, ensure the package is installed by loading it:

```
> library(rethomics)
```

2 Loading data

2.1 Data structure

In `rethomics` the goal is to store *all* the behavioural data in one single dataframe, which is standard for subsequent statistical analysis. In such a dataframe, every row correspond to a single measurement; that is he position of one animal at one time. Every column describes a statistical variable such as t, and X and Y positions, but they can also hold information about arbitrary conditions such as treatment, sex, age, genotype and so on.

By convention:

- t, the variable holding the time, is always *in seconds*
- X and Y, are *relative to the width of the region* they come from, and the *origin is top-left*

  For a single **experiment**, and when tracking a single animal (i.e. in a single **region**), your data table could look like:

```
> library(rethomics)
> data(multiple_iterative_y_mazes)
> single_animal <- multiple_iterative_y_mazes[
+               experiment_id == "female_FALSE_11.db"]
> print(single_animal)
```

```
           experiment_id region_id    sex sleep_deprived id               path
  1: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
  2: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
  3: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
  4: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
  5: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
 ---
```

[1] installation instructions are available here

```
4167: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
4168: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
4169: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
4170: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
4171: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
            t         x           y           w           h phi has_interacted
   1:  12.166 0.5505208 0.565104167 0.005729167 0.001562500   0          FALSE
   2:  12.200 0.5510417 0.564062500 0.008854167 0.003645833   0          FALSE
   3:  12.233 0.5520833 0.563020833 0.009375000 0.005729167   0          FALSE
   4:  12.266 0.5515625 0.562500000 0.011458333 0.006770833   0          FALSE
   5:  12.300 0.5510417 0.561979167 0.011458333 0.007812500   0          FALSE
  ---
4167: 151.633 0.6270833 0.002604167 0.009375000 0.003645833   0          FALSE
4168: 151.666 0.6265625 0.002604167 0.007812500 0.003645833   0          FALSE
4169: 151.833 0.6369792 0.001562500 0.003645833 0.001562500   0          FALSE
4170: 151.966 0.6328125 0.001562500 0.004687500 0.001562500   0          FALSE
4171: 152.100 0.6125000 0.003125000 0.004687500 0.004687500  90          FALSE
```

The first thing you may notice is that the first few columns seem unnecessary as they have constant values. This is because they describe variables that vary between individuals, and we have only one individual.

In order to understand the need for additional columns, we can load data from multiple experiments:

```
> library(rethomics)
> data(multiple_iterative_y_mazes)
> print(multiple_iterative_y_mazes)
```

```
            experiment_id region_id    sex sleep_deprived id                path
    1: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
    2: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
    3: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
    4: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
    5: female_FALSE_11.db         1 female          FALSE 11 female_FALSE_11.db
   ---
40426:   male_FALSE_25.db         1   male          FALSE 25   male_FALSE_25.db
40427:   male_FALSE_25.db         1   male          FALSE 25   male_FALSE_25.db
40428:   male_FALSE_25.db         1   male          FALSE 25   male_FALSE_25.db
40429:   male_FALSE_25.db         1   male          FALSE 25   male_FALSE_25.db
40430:   male_FALSE_25.db         1   male          FALSE 25   male_FALSE_25.db
             t         x           y           w           h phi has_interacted
    1:  12.166 0.5505208 0.565104167 0.005729167 0.001562500   0          FALSE
    2:  12.200 0.5510417 0.564062500 0.008854167 0.003645833   0          FALSE
    3:  12.233 0.5520833 0.563020833 0.009375000 0.005729167   0          FALSE
    4:  12.266 0.5515625 0.562500000 0.011458333 0.006770833   0          FALSE
    5:  12.300 0.5510417 0.561979167 0.011458333 0.007812500   0          FALSE
```

```
   ---
40426: 191.966 0.3661458 0.005208333 0.025520833 0.011458333  36          FALSE
40427: 192.000 0.3651042 0.004687500 0.024479167 0.011458333  37          FALSE
40428: 192.033 0.3645833 0.004166667 0.023958333 0.011979167  35          FALSE
40429: 192.066 0.3625000 0.006770833 0.021875000 0.011979167   0          FALSE
40430: 192.100 0.3625000 0.006250000 0.021875000 0.010937500   0          FALSE
```

The data will *always* have two columns: **experiment_id** and **region_id**. Together, these columns form a *key*, that is combinations of experiment *and* region represent unique animals. in other words, we can identify, unambiguously, any animal from its region and experiment identifier. Instead of using the legacy data.frames, `rethomic` takes advantage of Matt Doyle's powerful `data.table` package[2]. This makes it very easy and efficient to work with large amount of behavioural data. Common operation could involve filtering data and computing variable per condition or per individual. Let us go through several examples:

```
> library(rethomics)
> data(multiple_iterative_y_mazes)
> #We can simply call this data table `dt'
> dt <- multiple_iterative_y_mazes
> # keeping only females
> dt_female <- dt[sex == 'female',]
> # excluding any data point before 30 secondes (i.e. keeping >= 30s)
> dt_currated <- dt[t >= 30,]
> # Computing, per animal, the time spent in the experiment
> summary_dt =  dt[,.(time_spent = max(t) - min(t)),
+                 by=key(dt)]
> print(summary_dt)

        experiment_id region_id time_spent
1: female_FALSE_11.db         1    139.934
2:  female_FALSE_2.db         1    149.300
3:  female_TRUE_11.db         1    235.200
4:  female_TRUE_21.db         1    147.866
5:   female_TRUE_4.db         1    181.000
6:   female_TRUE_5.db         1    184.567
7:   female_TRUE_7.db         1    231.133
8:   male_FALSE_25.db         1    184.634
```

Much more can be achieved using `data.table`, so I would strongly recommend to, at least, read the package introduction[3].

[2] tutorial available at `http://user2014.stat.ucla.edu/files/tutorial_Matt.pdf`

[3] `http://cran.r-project.org/web/packages/data.table/vignettes/datatable-intro.pdf`

2.2 Loading one file

In this section, we will focus on loading data generated by pysolovideo[4] (i.e. `.db` files). In other words, we would like to load the data in the `.db` file as a data table structure such as the ones we described above. The function to load these data is called `loadPsvData`. It has many options, so it may be useful to read the documentation at some stage, but for now, we will just learn how to send a 'query' to obtain some data. I do not expect you to have already acquired any sort of data, so we will work with some of my samples, which I have enclosed within `rethomics`:

```
> # let us unpack these couple of files
> sample_files <- c("tube_monitor_validation_subset.db",
+                   "monitor_validation_subset.db")
> paths <- sapply(sample_files, loadSampleData)
> # the path should be unpacked in a temporary directory:
> print(paths)

                          tube_monitor_validation_subset.db
"/tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db"
                                monitor_validation_subset.db
     "/tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db"

> # The simplest case: we would like to load all the regions the first file:
> dt <- loadPsvData(paths[1],verbose=F)
> # We could also load all regions from all files:
> dt <- loadPsvData(paths,verbose=F)
```

Often, each file will correspond for a different condition (or combination of variables) that you would like to include in the resulting data table. In order to do that, we can pass a *query* to `loadPsvData`. a query is simply a table that has a column named `path`, and other columns for arbitrary conditions:

```
> query <- data.table(path=paths,
+                     treatment=c("control", "drug_A"))
> print(query)

                                                          path treatment
1: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
```

When working with a large number of files/conditions, it would be more common to load query from a spreadsheet/CSV file. One the query is prepared, we can simply load all the associated data:

[4] `https://github.com/gilestrolab/pySolo-Video`

```
> dt <- loadPsvData(query,verbose=F)
> print(dt)

                          experiment_id region_id
      1:      monitor_validation_subset.db         1
      2:      monitor_validation_subset.db         1
      3:      monitor_validation_subset.db         1
      4:      monitor_validation_subset.db         1
      5:      monitor_validation_subset.db         1
     ---
2402338: tube_monitor_validation_subset.db        20
2402339: tube_monitor_validation_subset.db        20
2402340: tube_monitor_validation_subset.db        20
2402341: tube_monitor_validation_subset.db        20
2402342: tube_monitor_validation_subset.db        20
                                                                path treatment
      1:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
      2:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
      3:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
      4:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
      5:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
     ---
2402338: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2402339: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2402340: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2402341: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2402342: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
               t         x          y          w          h phi mlog_L_x1000
      1:     2.0 0.4914966 0.02380952 0.02380952 0.02040816 146      -306000
      2:     2.5 0.5034014 0.02380952 0.03401361 0.02040816   0     -1858000
      3:     3.0 0.5476190 0.02551020 0.04931973 0.02380952   6     -1339000
      4:     3.5 0.5952381 0.02551020 0.04931973 0.02551020   0     -1308000
      5:     4.0 0.6003401 0.02891156 0.04761905 0.02551020   0     -2114000
     ---
2402338: 84224.5 0.7478705 0.04940375 0.03236797 0.02044293  10     -1753000
2402339: 84225.0 0.7563884 0.05110733 0.02214651 0.02044293  54     -1963000
2402340: 84225.5 0.7342419 0.04429302 0.04770017 0.02044293   3     -1601000
2402341: 84226.0 0.7529813 0.04940375 0.04770017 0.01873935   3     -1663000
2402342: 84226.5 0.7632027 0.04599659 0.04940375 0.01873935   8     -1953000
         xy_dist_log10x1000 xor_dist has_interacted
      1:               -306     1000          FALSE
      2:              -1858      580          FALSE
      3:              -1339     1000          FALSE
      4:              -1308      997          FALSE
      5:              -2114      346          FALSE
     ---
```

```
2402338:              -1753       529          FALSE
2402339:              -1963       420          FALSE
2402340:              -1601       761          FALSE
2402341:              -1663       547          FALSE
2402342:              -1953       402          FALSE
```

Importantly, there is now a column named 'treatment' in your data. This means you can then conveniently study differences explained by that variable. You could imagine a scenario where you have done 50 experiments where you have changed both sex and genotype, and want to explain behavioural changes with these variables.

2.3 Fancier queries

In some cases, you may want to load only specific regions, and even add different conditions per region, within the same experiment. If you add a column in your query called `region_id`, only specified regions will be loaded (as opposed to all regions in the previous example).

```
> query <- data.table(path=paths,
+                     treatment=c("control", "drug_A"),
+                     region_id=rep(1:20,each= 2))
> # We make a dummy query where  we imagine that
> # every other reagion contains a female individual
> query[, sex := ifelse(region_id %% 2, "male", "female" )]
```

```
                                                        path treatment
 1: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
 2:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
 3: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
 4:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
 5: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
 6:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
 7: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
 8:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
 9: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
10:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
11: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
12:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
13: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
14:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
15: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
16:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
17: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
18:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
19: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
```

```
20:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
21: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
22:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
23: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
24:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
25: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
26:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
27: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
28:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
29: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
30:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
31: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
32:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
33: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
34:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
35: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
36:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
37: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
38:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
39: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
40:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
                                                          path treatment
    region_id    sex
 1:         1   male
 2:         1   male
 3:         2 female
 4:         2 female
 5:         3   male
 6:         3   male
 7:         4 female
 8:         4 female
 9:         5   male
10:         5   male
11:         6 female
12:         6 female
13:         7   male
14:         7   male
15:         8 female
16:         8 female
17:         9   male
18:         9   male
19:        10 female
20:        10 female
21:        11   male
22:        11   male
23:        12 female
```

```
24:        12 female
25:        13   male
26:        13   male
27:        14 female
28:        14 female
29:        15   male
30:        15   male
31:        16 female
32:        16 female
33:        17   male
34:        17   male
35:        18 female
36:        18 female
37:        19   male
38:        19   male
39:        20 female
40:        20 female
    region_id    sex

> # Note that we load only regions lower than twenty
>
> # Now we can use our query
> dt <- loadPsvData(query,verbose=F)
> print(dt)

                                                                path treatment
      1:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
      2:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
      3:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
      4:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
      5:      /tmp/RtmpRbDHlQ/db_files/monitor_validation_subset.db    drug_A
     ---
2095923: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2095924: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2095925: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2095926: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
2095927: /tmp/RtmpRbDHlQ/db_files/tube_monitor_validation_subset.db   control
         region_id    sex                     experiment_id       t         x
      1:         1   male      monitor_validation_subset.db     2.0 0.4914966
      2:         1   male      monitor_validation_subset.db     2.5 0.5034014
      3:         1   male      monitor_validation_subset.db     3.0 0.5476190
      4:         1   male      monitor_validation_subset.db     3.5 0.5952381
      5:         1   male      monitor_validation_subset.db     4.0 0.6003401
     ---
2095923:        20 female tube_monitor_validation_subset.db 84224.5 0.7478705
2095924:        20 female tube_monitor_validation_subset.db 84225.0 0.7563884
```

```
2095925:        20 female tube_monitor_validation_subset.db 84225.5 0.7342419
2095926:        20 female tube_monitor_validation_subset.db 84226.0 0.7529813
2095927:        20 female tube_monitor_validation_subset.db 84226.5 0.7632027
                  y          w          h phi mlog_L_x1000 xy_dist_log10x1000
      1: 0.02380952 0.02380952 0.02040816 146      -306000               -306
      2: 0.02380952 0.03401361 0.02040816   0     -1858000              -1858
      3: 0.02551020 0.04931973 0.02380952   6     -1339000              -1339
      4: 0.02551020 0.04931973 0.02551020   0     -1308000              -1308
      5: 0.02891156 0.04761905 0.02551020   0     -2114000              -2114
     ---
2095923: 0.04940375 0.03236797 0.02044293  10     -1753000              -1753
2095924: 0.05110733 0.02214651 0.02044293  54     -1963000              -1963
2095925: 0.04429302 0.04770017 0.02044293   3     -1601000              -1601
2095926: 0.04940375 0.04770017 0.01873935   3     -1663000              -1663
2095927: 0.04599659 0.04940375 0.01873935   8     -1953000              -1953
         xor_dist has_interacted
      1:     1000          FALSE
      2:      580          FALSE
      3:     1000          FALSE
      4:      997          FALSE
      5:      346          FALSE
     ---
2095923:      529          FALSE
2095924:      420          FALSE
2095925:      761          FALSE
2095926:      547          FALSE
2095927:      402          FALSE
```

Note that there were some warnings, this is because some of the requested regions are not in the example data. Indeed, I only provided a small subset of available regions in order to reduce the size of the samples. Also, as before, our data table(`dt`) has additional columns for the conditions we added.

As a heads-up, lets see how one would use data table syntax to compute median x position per sex and treatment:

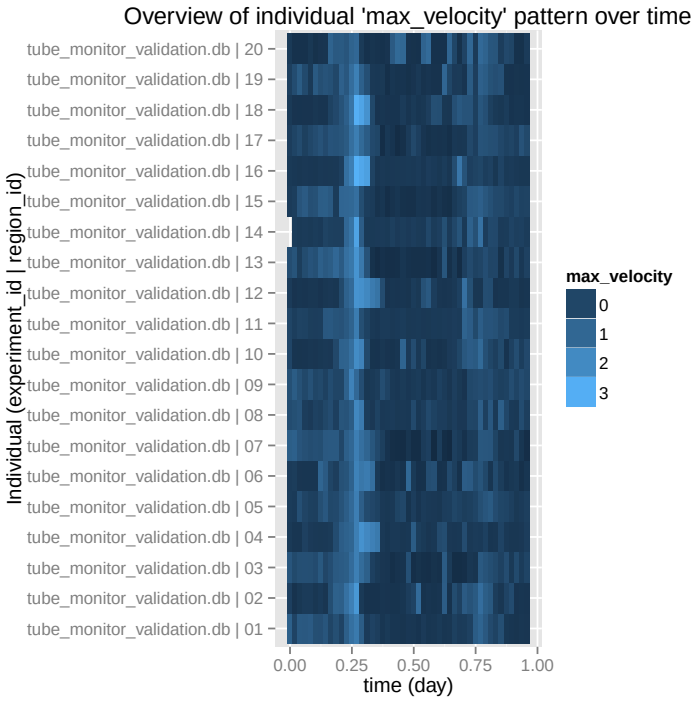
```
> summary <- dt[, .(median_x = median(x)),
+                 by=c(key(dt),"sex")]
> print(summary)

                       experiment_id region_id    sex   median_x
 1:      monitor_validation_subset.db         1   male 0.05612245
 2:      monitor_validation_subset.db         2 female 0.14795918
 3:      monitor_validation_subset.db         3   male 0.46088435
 4:      monitor_validation_subset.db         8 female 0.52721088
 5:      monitor_validation_subset.db        13   male 0.51788756
```

```
 6:      monitor_validation_subset.db        19   male 0.39693356
 7: tube_monitor_validation_subset.db         1   male 0.20136519
 8: tube_monitor_validation_subset.db         6 female 0.39249147
 9: tube_monitor_validation_subset.db         9   male 0.19112628
10: tube_monitor_validation_subset.db        11   male 0.70136519
11: tube_monitor_validation_subset.db        17   male 0.39931741
12: tube_monitor_validation_subset.db        19   male 0.66382253
13: tube_monitor_validation_subset.db        20 female 0.64054514
```

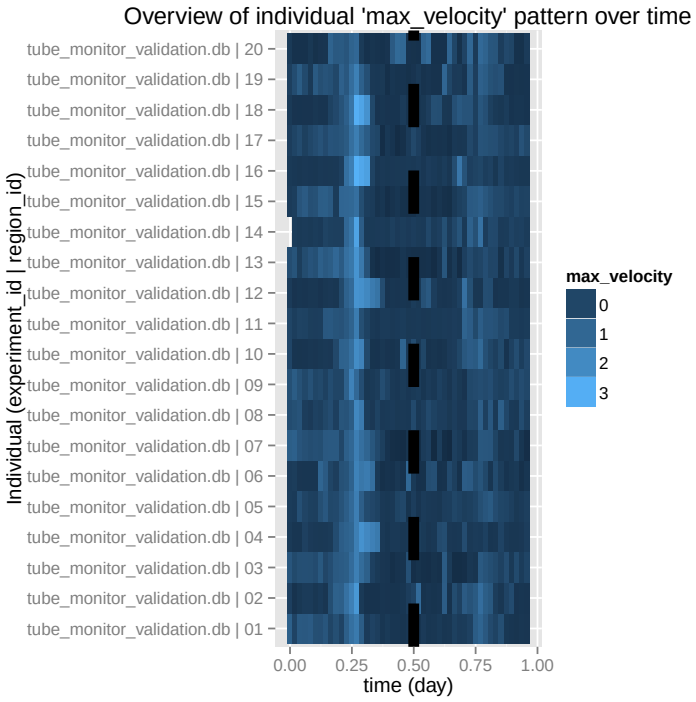
2.4 Visualisation

This is the exciting part! Now we know how to load data, I will use preloaded data in the package. The `tube_monitor_validation` data set is one experiement with 20 regions. We would like to have an overview of the activity in each region as a graphical quality control:

```
> data(tube_monitor_validation)
> # we make a sleep annotation, for ach animal. this will
> # also compute maximal velocity for every chunk of 10s
> my_data <- tube_monitor_validation[,sleepAnnotation(.SD),
+                                    by=key(tube_monitor_validation)]
> pl<- overviewPlot(max_velocity,my_data)
> print(pl)
```




```
> pl <- pl + geom_vline(aes(xintercept=c(0.5)),linetype=2,size=3)
> print(pl)
```


Overview of individual 'max_velocity' pattern over time
Individual (experiment_id | region_id)
tube_monitor_validation.db | 20
tube_monitor_validation.db | 19
tube_monitor_validation.db | 18
tube_monitor_validation.db | 17
tube_monitor_validation.db | 16
tube_monitor_validation.db | 15
tube_monitor_validation.db | 14
tube_monitor_validation.db | 13
tube_monitor_validation.db | 12
tube_monitor_validation.db | 11
tube_monitor_validation.db | 10
tube_monitor_validation.db | 09
tube_monitor_validation.db | 08
tube_monitor_validation.db | 07
tube_monitor_validation.db | 06
tube_monitor_validation.db | 05
tube_monitor_validation.db | 04
tube_monitor_validation.db | 03
tube_monitor_validation.db | 02
tube_monitor_validation.db | 01
0.00
0.25
0.50
0.75
1.00
time (day)
max_velocity
0
1
2
3