Analysing growth curves and other user-defined plates in opm

Lea A.I. Vaas
Leibniz Institute DSMZ

Markus Göker
Leibniz Institute DSMZ

Abstract

This is tutorial about the analysis of growth curves and other user defined kinetics with the **opm** package in the version of December 16, 2013. It is explained how any kinds of growth or respiration measurements can be input into **opm**. We also show how Phenotype Microarray (PM) data with user-defined plate types can be analysed. Analysing such data visually and statistically requires in some cases adaptations of function arguments whose defaults are targeting PM data. All these practically relevant issues are explained in detail.

Keywords: Growth Kinetics.

1. Introduction

A detailed description of the OmniLog® Phenotype Microarray (PM) system, its measuring procedure and data characteristics are found in the vignette "**opm**: An R Package for Analysing OmniLog® Phenotype Microarray Data" (called "main tutorial" in the following). How substrate information stored within **opm** can be accessed and used for advanced visual and statistical analyses is explained in the vignette "Working with substrate information in **opm**" (called "substrate tutorial" in the following). The description of the methods below presupposes that the user is familiar with the usage of **opm** and has studied the main tutorial, the substrate tutorial as well as the entries of the **opm** manual relevant to her or his research. Especially the concepts behind, and the methods available for, the different classes of **opm** objects should be known before starting with this tutorial.

In addition to visual inspection or statistical comparative analyses of PM data, as described in the main tutorial and the substrate tutorial, users might be interested in analysing data other than PM data, or analysing PM with user-defined plate types. To work with user-defined PM plates only requires registering these plates, i.e. storing a mapping from well coordinates to substrate names, and optionally also a full, descriptive name for the plate. The analysis of data other than PM data, such as growth curves, additionally requires inputting these data and converting them to OPMX objects. Moreover, some defaults of the plotting functions are only suitable for PM data. Hence, the functions should be called slightly distinctly.

Besides these slight restrictions, which are illustrated with examples below, non-PM data can be analysed with **opm** almost like PM data.

2. Preparation

As usual, **opm** must be loaded before any analysis can begin:

```
R> if ("package:opm" %in% search())
     detach("package:opm", unload = TRUE)
R> library("opm")
```

3. Growth-curve data input

3.1. User-entered data frames

In the following we will use the growth-measurements data set from Vaas, Marheine, Sikorski, Göker, and Schumacher (2013) as exemplar. These data have been entered by hand and then input into R with one of the functions for reading Comma-Separated Values (CSV), yielding a data frame, which comes with **opm**:

```
R> data("potato")
R> head(potato)
```

```
  Genotype  Treatment Replicate Time  FM DM
1  07-08-1 0.16M NaCl         1    2 597 44
2  07-08-1 0.16M NaCl         2    2 550 40
3  07-08-1 0.16M NaCl         3    2 633 48
4  07-08-1 0.16M NaCl         4    2 490 31
5  07-08-1 0.16M NaCl         5    2 617 47
6  07-08-1 0.16M NaCl         1    4 585 55
```

For details on this data set, enter `?potato` at the R prompt. The measurements are in "long" format and must be reshaped using the eponymous function into "wide" format. We do this separately for the dry-mass and fresh-mass measurements within the data set:

```
R> potato.fm <- reshape(potato, v.names = "FM", drop = "DM", direction = "wide",
     idvar = c("Genotype", "Treatment", "Replicate"), timevar = "Time")
R> potato.dm <- reshape(potato, v.names = "DM", drop = "FM", direction = "wide",
     idvar = c("Genotype", "Treatment", "Replicate"), timevar = "Time")
```

The main function for converting user-defined data frames to OPMX or MOPMX objects is `opmx`, which can directly be applied to the objects created in the last step. This works because the "horizontal" input format of `opmx` corresponds to the "wide" format of `reshape`.

```
R> potato.fm <- opmx(potato.fm, position = c("Genotype", "Replicate"),
     well = "Treatment", prefix = "FM.",
     full.name = c(fm = "Growth experiment, fresh mass"))
R> potato.dm <- opmx(potato.dm, position = c("Genotype", "Replicate"),
     well = "Treatment", prefix = "DM.",
     full.name = c(dm = "Growth experiment, dry mass"))
```

The data frame contains all substrate information (in the "Treatment" column). Hence, **opm** registers the mapping from well coordinates to substrate names on the fly. The plate type must be provided, however. As it is not within the data frame, the short name of the plate type is taken from the `full.name` argument, whose main purpose is to enter the full, descriptive name of the plate type. "Genotype" and "Replicate" go to the metadata of the resulting object and together identify each plate. In the case of PM data, this is done using the position of the plate within the OmniLog® reader. Thus the relevant argument here is `position`, which must be supplied unless there is a column of that name. The `prefix` argument helps identifying the columns with measurements over time.

The registered plate type can be queried as follows:

```
R> dim(potato.fm)

[1] 15  9  4

R> dim(potato.dm)

[1] 15  9  4

R> plate_type(TRUE) # shows all user-defined plates

[1] "CUSTOM:DM" "CUSTOM:FM"

R> listing(wells(plate = c("CUSTOM:FM", "CUSTOM:DM")))

CUSTOM:FM:
- Growth experiment, fresh mass
- A01: 0.16M NaCl
  A02: 0.32M NaCl
  A03: 0.5M Sorbitol
  A04: Control
CUSTOM:DM:
- Growth experiment, dry mass
- A01: 0.16M NaCl
  A02: 0.32M NaCl
  A03: 0.5M Sorbitol
  A04: Control
```

Note the prefix "CUSTOM:", which is used to distinguish user-defined plate type from those that come with **opm**. The object resulting from `listing` can be output with `to_yaml` or `saveRDS` for externally storing plate types in files.

3.2. Direct registration of plate types

TODO.

3.3. Input of TECAN data

Here we will use an exemplar that comes with **opm** as input data file:

```
R> tecan.file <- opm_files("growth")
R> tecan.file <- grep("tecan", tecan.file, ignore.case = TRUE, value = TRUE)
R> tecan <- read.table(tecan.file)
R> head(tecan)

  V1    V2    V3    V4    V5    V6    V7
1 <> 1.000 2.000 3.000 4.000 5.000 6.000
2  A 0.087 0.088 0.087 0.088 0.085 0.084
3  B 0.087 0.088 0.087 0.086 0.087 0.085
4  C 0.083 0.082 0.081 0.083 0.079 0.077
5  D 0.083 0.083 0.081 0.082 0.080 0.079
6 <> 1.000 2.000 3.000 4.000 5.000 6.000
```

This format is not particularly useful within R but can be converted using the "rectangular" mode of `opmx`.

TODO.

4. Visualisation of growth curves

TODO.

```
R> print(xy_plot(potato.fm, theor.max = FALSE, include = "Genotype",
     main = list(in.parens = FALSE), neg.ctrl = FALSE, ylab = "Mass [g]"))
R> print(xy_plot(potato.dm, theor.max = FALSE, include = "Genotype",
     main = list(in.parens = FALSE), neg.ctrl = FALSE, ylab = "Mass [g]"))
```

5. Estimating parameters from growth curves

TODO.

6. Statistical analysis of growth curves

TODO.

7. Acknowledgements

We are grateful to Victoria Michael (Deutsche Sammlung von Mikroorganismen und Zellkulturen (DSMZ)) for providing growth curves measured with a TECAN instrument.

References

Vaas LAI, Marheine M, Sikorski J, Göker M, Schumacher HM (2013). "Impacts of pr-10a Overexpression at the Molecular and the Phenotypic Level." *International Journal of Molecular Sciences*, **14**, 15141–15166. doi:10.3390/ijms140715141. URL http://www.mdpi.com/1422-0067/14/7/15141.

Affiliation:

Markus Göker
Leibniz Institute DSMZ – German Collection of Microorganisms and Cell Cultures
Braunschweig

Telephone: +49/531-2616-272
Fax: +49/531-2616-237
E-mail: markus.goeker@dsmz.de
URL: www.dsmz.de