Estimation of Growth Rates with package `growthrates`

Thomas Petzoldt

2015-08-10

Introduction

The growth rate of a population is a direct measure of fitness. Therefore, determination of growth rates is common in many disciplines of theoretical and applied biology, e.g. physiology, ecology, eco-toxicology or pharmacology.

This package aims to streamline estimation of growth rates from direct or indirect measures of population density (e.g. cell counts, optical density or fluorescence) determined in batch experiments or field observations. It should be applicable to different species of bacteria, archaea, protists, and metazoa, e.g. *E. coli*, *Cyanobacteria*, *Paramecium*, green algae or *Daphnia*.

The determination of growth rates from chemostat and semi-continuous cultures is currently not covered by the package, but we are open to include it, depending on your interest and the availability of data.

Methods

The package includes three types of methods:

- Nonlinear fitting of parametric growth models like the logistic or the Gompertz growth model. Parametric model fitting is done by using package `FME` (Flexible Modelling Environment) of Soetaert and Petzoldt (2010) . In addition to growth models given in closed form (i.e. empirical regression equations or analytical solution of differential equations) it is also possible to use numerically solved differential equation models. Such models are then solved with package 'deSolve' (Soetaert, Petzoldt, and Setzer 2010).
- Fitting of linear models to the period of exponential growth using the "growth rates made easy method" of Hall and Barlow (2013) ,
- Nonparametric growthrate estimation by using smoothers. R contains several very powerful methods for this, that can leveraged for this purpose. The currently implemented method uses function `smooth.spline`, similar to the package `grofit` (Kahm et al. 2010).

The package contains methods to fit single data sets or complete series of data sets organized in a data frame. It contains also functions for extracting results (e.g. `coef`, `summary`, `deviance`, `obs`, `residuals`, `rsquared` and `results`) and methods for plotting (`plot`, `lines`). The implementation follows an object oriented style, so that the functions above determine automatically which method is used for a given class of objects.

Data Set

The data set for demonstrating main features of the package was provided by Claudia Seiler from one of a series of plate reader experiments carried out at the Institute of Hydrobiology of TU Dresden. It describes growth of three different strains of bacteria (D=donor, R=recipient, T=transconjugant) in dependence of a gradient of the antibiotics Tetracycline.

```
library("growthrates")
```

Firstly, we load the data and inspect its structure with `str`:

```
data(bactgrowth)
str(bactgrowth)
```

```
## 'data.frame':    2232 obs. of  5 variables:
##  $ strain   : Factor w/ 3 levels "D","R","T": 3 3 3 3 3 3 3 3 3 3 ...
##  $ replicate: int  2 2 2 2 2 2 2 2 2 2 ...
##  $ conc     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ time     : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ value    : num  0.013 0.014 0.017 0.022 0.03 0.039 0.042 0.045 0.048 0.049 ...
```

And we can also inspect the full data set with `View(growthrates)` or look at the first few lines with `head`:

```
head(bactgrowth)
```

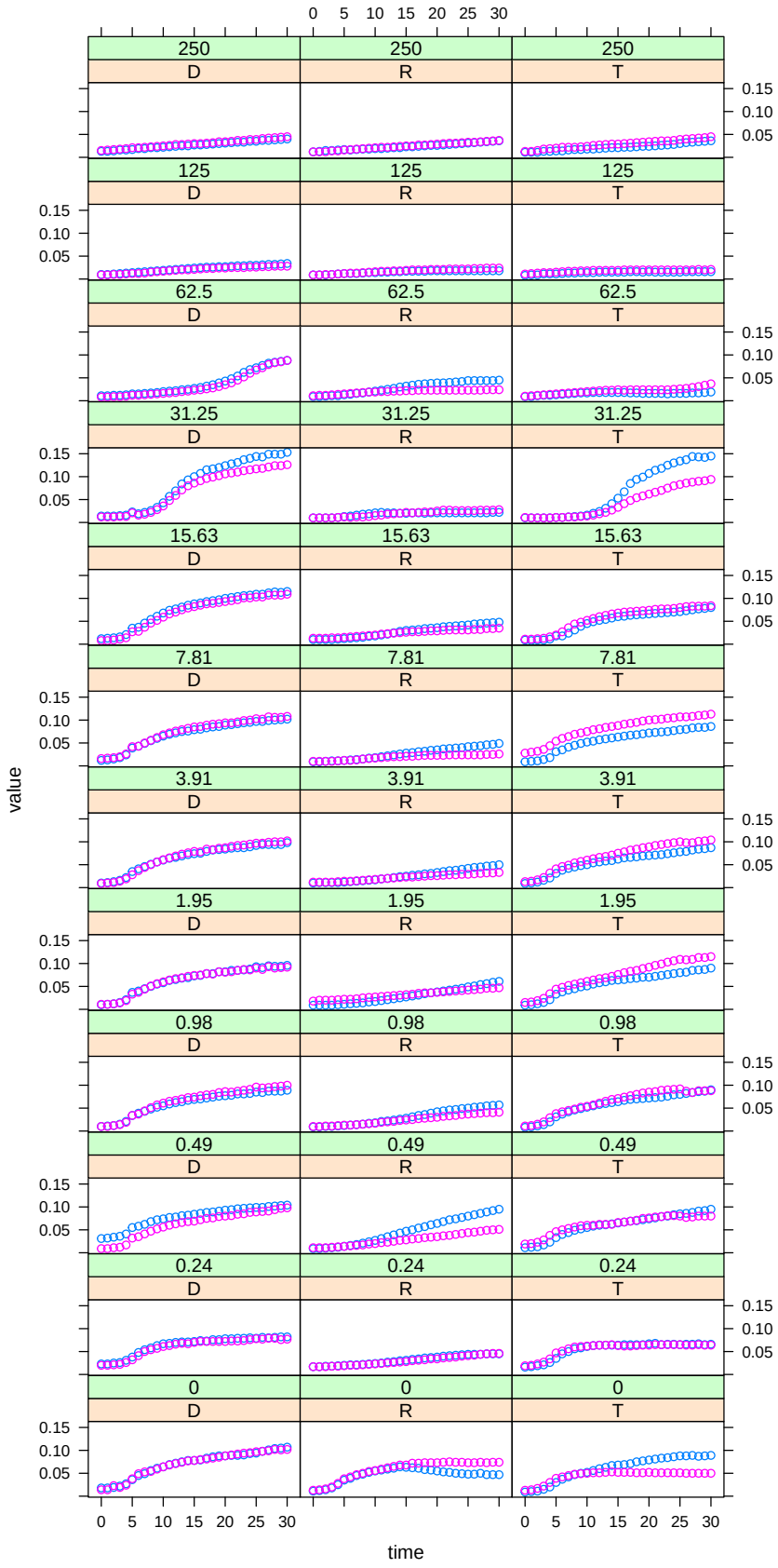
```
##   strain replicate conc time value
## 1      T         2    0    0 0.013
## 2      T         2    0    1 0.014
## 3      T         2    0    2 0.017
## 4      T         2    0    3 0.022
## 5      T         2    0    4 0.030
## 6      T         2    0    5 0.039
```

Examples

Inspect the Data

Plot raw data:

```
library(lattice)
data(bactgrowth)
xyplot(value ~ time|strain+as.factor(conc), data=bactgrowth, groups = replicate)
```


250
D
R
T
125
62.5
31.25
15.63
7.81
3.91
1.95
0.98
0.49
0.24
0
value
time
0 5 10 15 20 25 30
0.15
0.10
0.05

Fitting Models to Individual Data Sets

Single data sets can be analysed with functions `fit_easylinear`, `fit_growthmodels` or `fit_splines`. As a prerequisite, single data sets containing only one treatment have to be extracted from a complete experiment, which can be done with function 'multsiplit'. In the following example, the full data table is first split into a list of experiments according to a vector of criteria and then the first experiment is extracted:

Easy Linear Method

```
splitted.data <- multisplit(bactgrowth, c("strain", "conc", "replicate"))
dat <- splitted.data[[1]]
```

In the next step, model fitting is done, e.g. with the "easylinear" method:

```
fit <- fit_easylinear(dat$time, dat$value)
```

This method fits segments of linear models to the log-transformed data and tries to find the maximum growth rate. Several functions exist to inspect the outcome of the model fit, e.g.

```
summary(fit)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##        1        2        3        4        5        6
##  0.02113 -0.03716 -0.03727  0.04552  0.06376 -0.05598
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.39425    0.06429  -68.35 2.74e-07 ***
## x            0.20490    0.01336   15.34 0.000105 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05587 on 4 degrees of freedom
## Multiple R-squared:  0.9833, Adjusted R-squared:  0.9791
## F-statistic: 235.3 on 1 and 4 DF,  p-value: 0.0001053
```

```
coef(fit)      # exponential growth parameters
```

```
##        y0        mu
## 0.0123482 0.2048985
```

```
rsquared(fit)  # coefficient of determination (of log-transformed data)
```

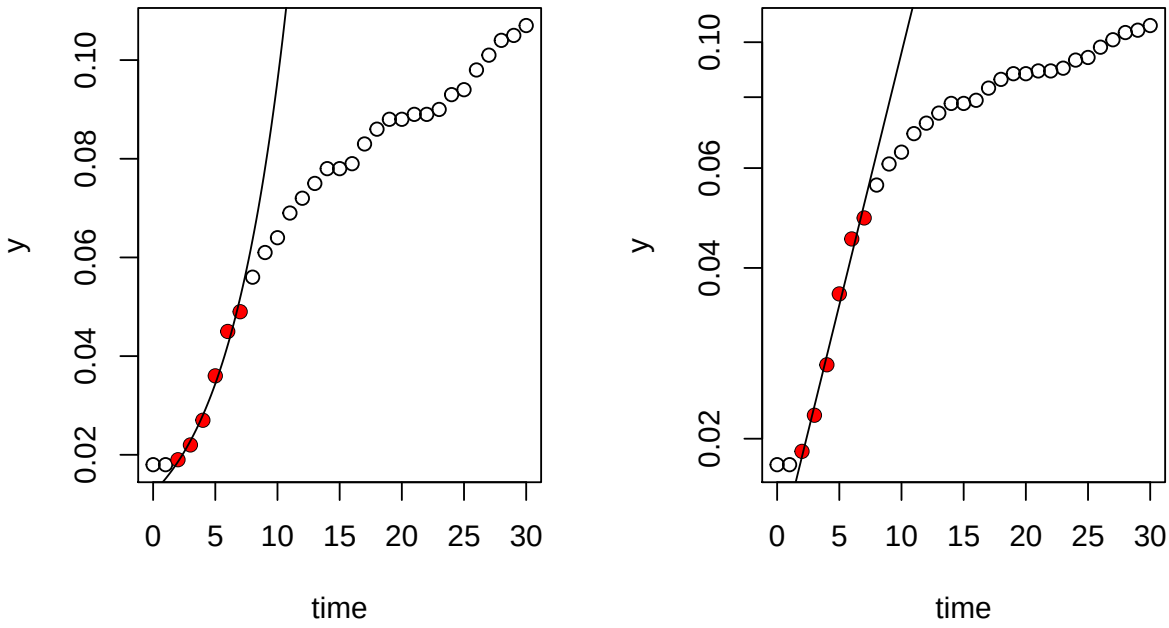
```
##        r2
## 0.9832876
```

```
deviance(fit)  # residual sum of squares of log-transformed data
```

```
## [1] 0.01248744
```

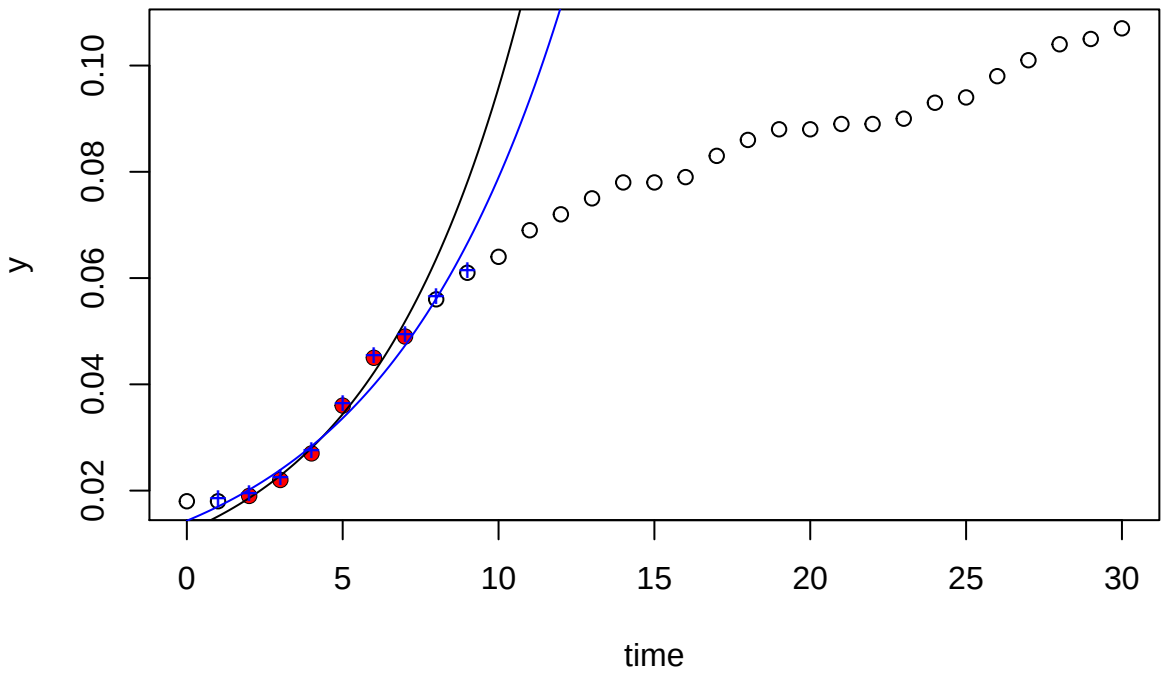
Plotting can then be done either in log-scale or after re-transformation

```
par(mfrow=c(1,2))
plot(fit)
plot(fit, log="y")
```



and in addition to Hall and Barlow (2013) it is also possible to modify the default settings of the algorithm:

```
fitx <- fit_easylinear(dat$time, dat$value, h=8, quota=0.95)
plot(fit)
lines(fitx, pch="+", col="blue")
```



Parametric Nonlinear Growth Models

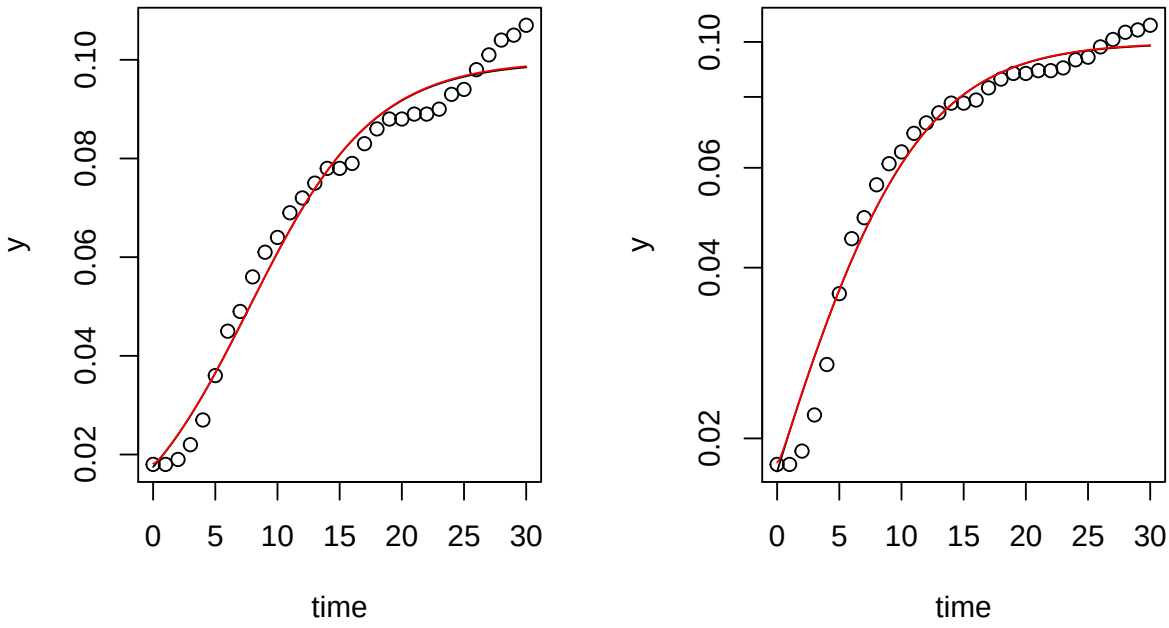
```
p     <- c(y0=0.01, mu=0.2, K=0.1)
lower <- c(y0=1e-6, mu=0,   K=0)
upper <- c(y0=0.05, mu=5,   K=0.5)

fit1 <- fit_growthmodel(FUN=grow_logistic, p=p, dat$time, dat$value,
                        lower=lower, upper=upper)


p     <- c(yi=0.02, ya=0.001, kw=0.1,   mu=0.2, K=0.1)
lower <- c(yi=1e-6, ya=1e-6, kw=0,    mu=0,   K=0)
upper <- c(yi=0.05, ya=0.05, kw=10,   mu=5,   K=0.5)

fit2 <- fit_growthmodel(FUN=grow_twostep, p=p, time=dat$time, y=dat$value,
                        lower=lower, upper=upper)


coef(fit1)
```

```
##         y0         mu          K
## 0.01748268 0.20006908 0.09962612
```

```
coef(fit2)
```

```
##          yi          ya          kw          mu           K
## 0.014224009 0.003910263 3.884232782 0.199236020 0.099856849
```

```
par(mfrow=c(1,2))
plot(fit1)
lines(fit2, col="red")


plot(fit1, log="y")
lines(fit2, col="red")
```



Differential equation models

In the example above, growth is described as a two-step process of adaption of inactive cells y_i and logistic growth of active cells y_a:

$$\frac{dy_i}{dt} = -k_w \cdot y_i$$

$$\frac{dy_a}{dt} = k_w \cdot y_i + \mu \cdot y_a \cdot \left(1 - \frac{K}{y_a + y_i}\right)$$

with amount of total organisms $y = y_i + y_a$, the parameters adaption rate k_w, intrinsic growth rate μ, carrying capacity K. The initial abundance (normally y_0) is splitted in two separate values, $y_{i,0}$ and $y_{a,0}$ that are by default also fitted.

The underlying ordinary differential equation (ODE) model has no simple analytical solution and is therefore solved numerically using a differential solver from package **deSolve**. Here both, the model and the solver are running in compiled code (C resp. Fortran), but it is of course also possible to define user-specified models in R code.

Selective parameter fitting

Despite the fact that the above model is solved as a differential equation, the relatively high number of parameters may need special care, too. In such cases, package growthrates allows to fit subsets of parameters while setting the others to fixed values. In the following, this is done by specifying a subset without initial abundances y_a and y_i in `which`:

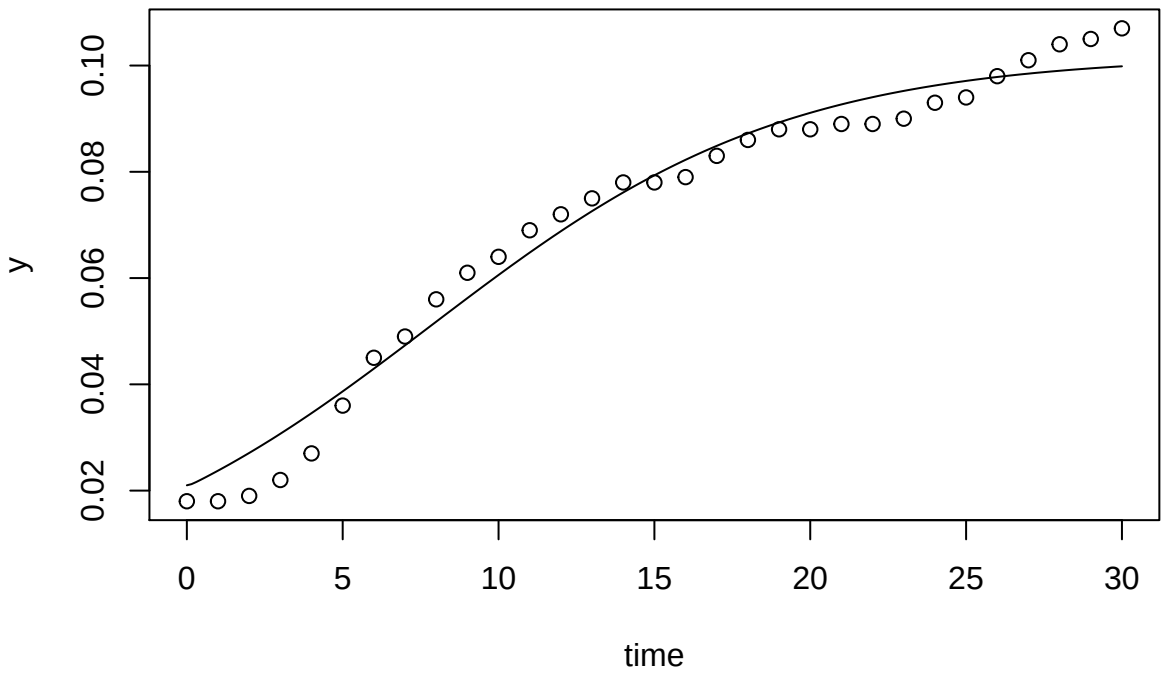
```
fit3 <- fit_growthmodel(FUN=grow_twostep, p=p, time=dat$time, y=dat$value,
                        lower=lower, upper=upper, which=c("kw", "mu", "K"))

summary(fit3)
```

```
##
## Parameters:
##     Estimate Std. Error t value Pr(>|t|)
## kw  9.791445  66.893493   0.146    0.885
## mu  0.174802   0.015625  11.188 7.64e-12 ***
## K   0.101923   0.002429  41.958  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.00452 on 28 degrees of freedom
##
## Parameter correlation:
##         kw      mu       K
## kw  1.0000 -0.8893  0.5405
## mu -0.8893  1.0000 -0.7735
## K   0.5405 -0.7735  1.0000
```

```
coef(fit3)
```

```
##        yi        ya        kw        mu         K
## 0.0200000 0.0010000 9.7914448 0.1748016 0.1019230
```

```
plot(fit3)
```

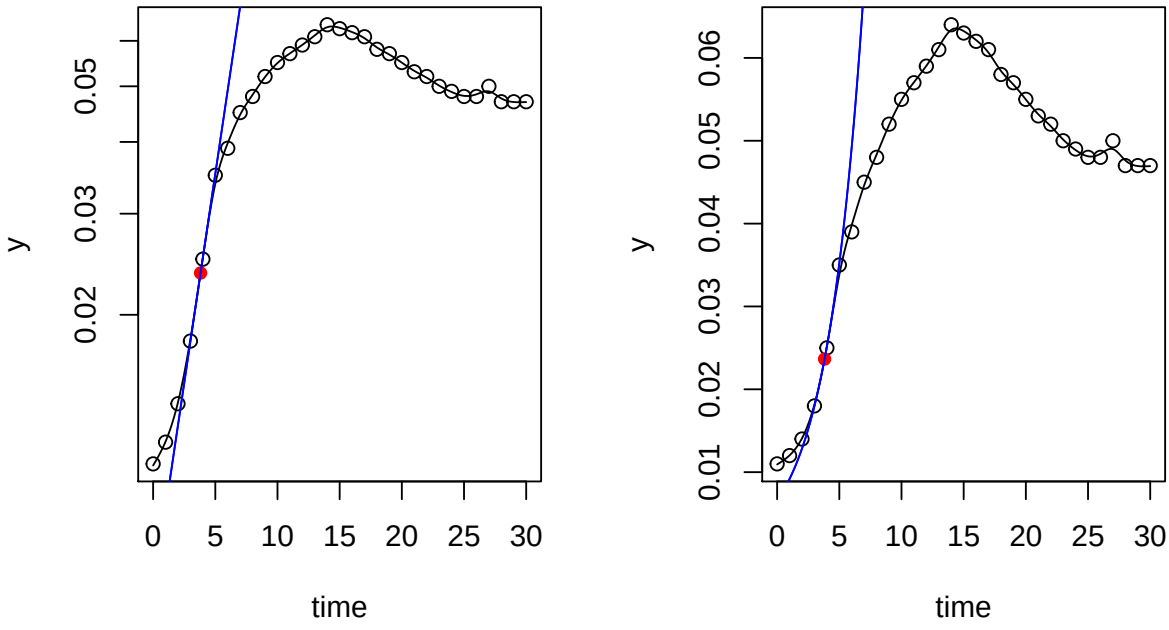


We see that `summary` shows only the fitted parameters whereas `coef` contains the full set.

Both, start values for the fitted parameters and fixed values of the other parameters need to be available in the start parameter vector `p`, while `upper` and `lower` bounds for the fixed parameters can be omitted.

Nonparametric Smoothing Splines

```
dat <- splitted.data[[2]]
time <- dat$time
y    <- dat$value

## automatic smoothing with cv
res <- fit_spline(time, y)

par(mfrow=c(1,2))
plot(res, log="y")
plot(res)
```




```
coef(res)
```

```
##          y0          mu
## 0.006562443 0.335991063
```

Fiting Multiple Data Sets

Fiting multiple data sets at once is possible with functions `all_easyliner`, `all_growthrates` and `alls_splines`. Usage is similar for all methods, whereas the parameters are analogous to the single-fit methods. Both, the easy growth rates and the smooting splines method are quite robust. In contrast to this, parametric fits with function `all_growthrates` need more care and a little bit more computational power.

Special emphasis should be given to the selection of good starting points. In addition, it is possible to select an alternative optimization algorithm, to enable additional output (`trace`) or to fine-tune their optimization control parameters. Nevertheless, it should be noted that parametric models have more explanatory power and may therefore be advantageous for basic research.

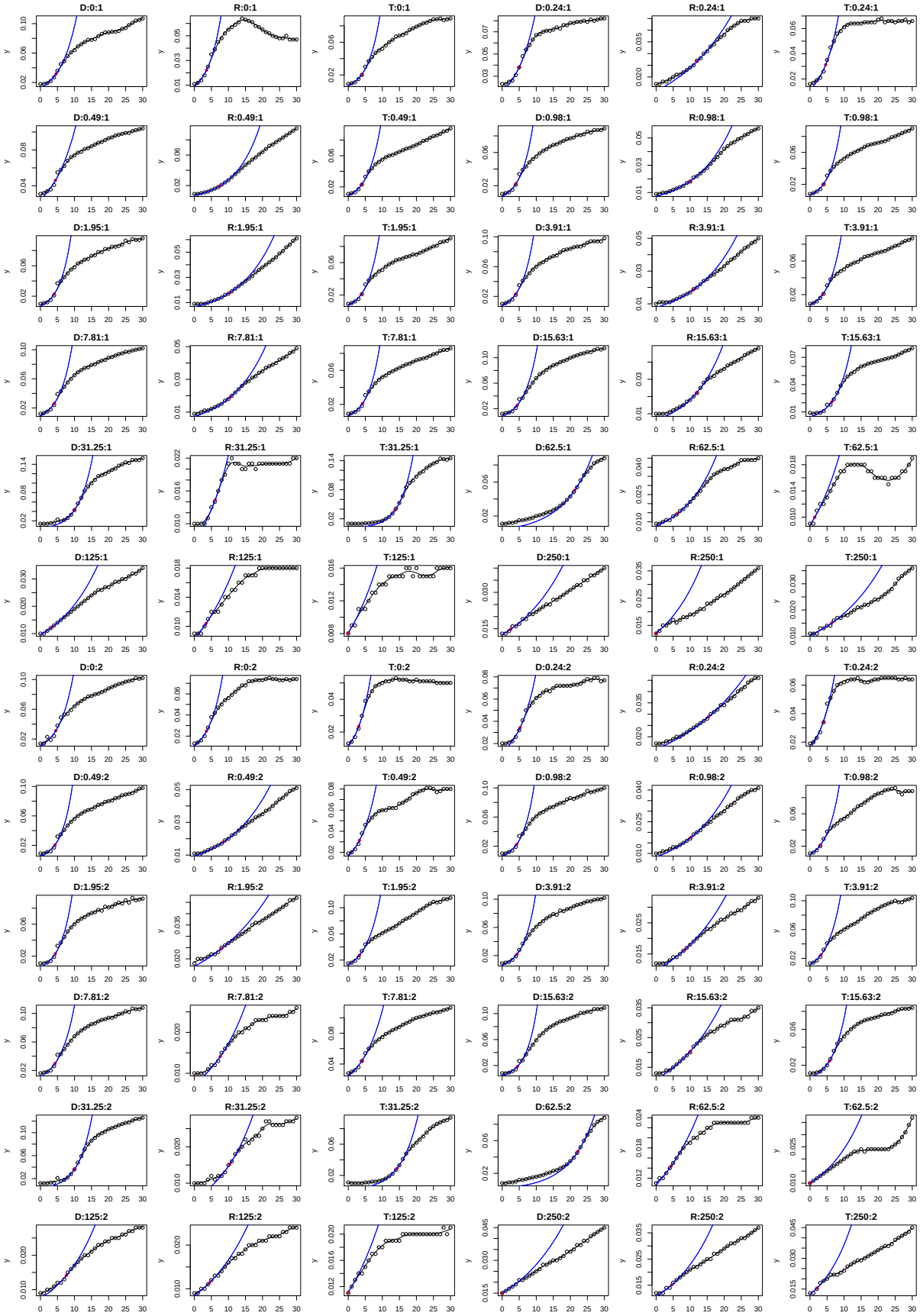
Nonlinear optimization is done with parallelized code, so multi-core computers can speed up computation.

It can be a good idea, to use a nonparametric approach like the smoothing spline method to get a first impression and, potentially, to derive start parameters for a parametric model.

In the following, we show an example with the smoothing spline method:

```
many_fits <- all_splines(bactgrowth,
                  criteria=c("strain", "conc", "replicate"), spar=0.5)
```

```r
par(mfrow=c(12,6))
par(mar=c(2.5,4,2,1))
plot(many_fits)
```

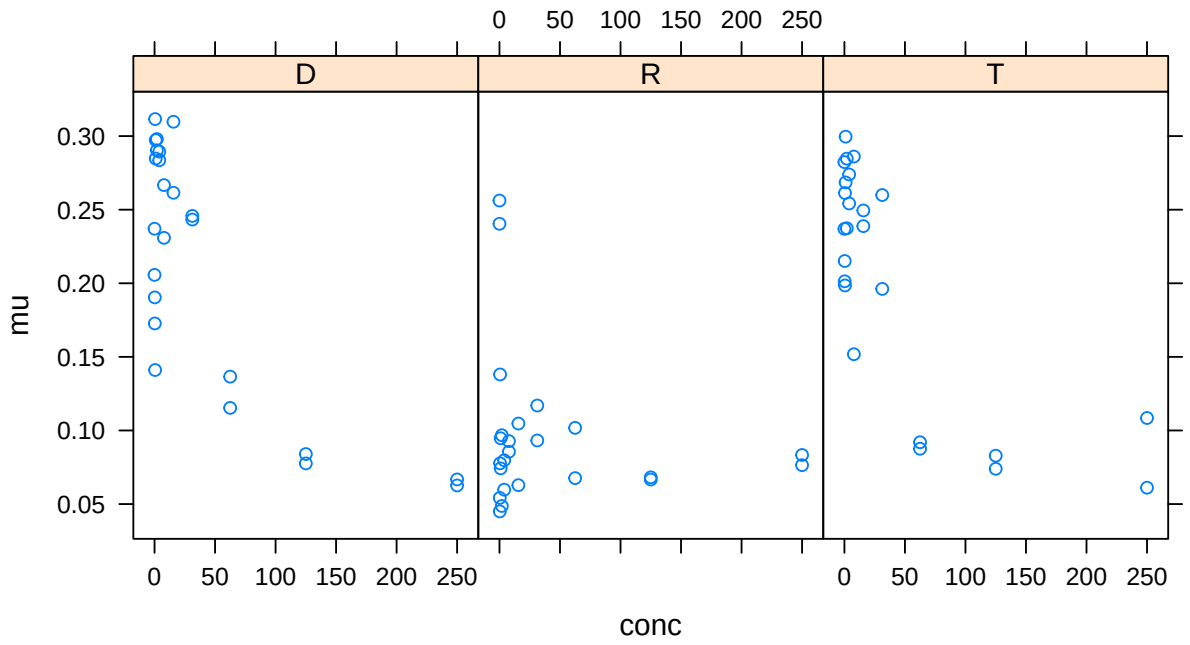

D:0:1
R:0:1
T:0:1
D:0.24:1
R:0.24:1
T:0.24:1
D:0.49:1
R:0.49:1
T:0.49:1
D:0.98:1
R:0.98:1
T:0.98:1
D:1.95:1
R:1.95:1
T:1.95:1
D:3.91:1
R:3.91:1
T:3.91:1
D:7.81:1
R:7.81:1
T:7.81:1
D:15.63:1
R:15.63:1
T:15.63:1
D:31.25:1
R:31.25:1
T:31.25:1
D:62.5:1
R:62.5:1
T:62.5:1
D:125:1
R:125:1
T:125:1
D:250:1
R:250:1
T:250:1
D:0:2
R:0:2
T:0:2
D:0.24:2
R:0.24:2
T:0.24:2
D:0.49:2
R:0.49:2
T:0.49:2
D:0.98:2
R:0.98:2
T:0.98:2
D:1.95:2
R:1.95:2
T:1.95:2
D:3.91:2
R:3.91:2
T:3.91:2
D:7.81:2
R:7.81:2
T:7.81:2
D:15.63:2
R:15.63:2
T:15.63:2
D:31.25:2
R:31.25:2
T:31.25:2
D:62.5:2
R:62.5:2
T:62.5:2
D:125:2
R:125:2
T:125:2
D:250:2
R:250:2
T:250:2

In this example, smoothness is set to an average value (`spar = 0.5`). Other values between zero and one will result in different degrees of smoothing. If `spar` is omitted, leave-one-out cross-validation is used to determine smoothness automatically.

Finally

Dependency of growth rate on antibiotic concentration for the three strains:

```
many_res <- results(many_fits)
xyplot(mu ~ conc|strain, data=many_res)
```



Describing the observed dependency can again be described with parametric or nonparametric a functional response curves, which may then be done directly in R or by using a specialized package for dose-response curves, for example package **drc** (Ritz and Streibig 2005).

Acknowledgments

Many thanks to Claudia Seiler for the data set. Many thanks to the R Core Team (R Core Team 2015) for developing and maintaining **R**. This documentation was written using **knitr** (Xie 2014) and **rmarkdown** (Allaire et al. 2015).

References

Allaire, JJ, Joe Cheng, Yihui Xie, Jonathan McPherson, Winston Chang, Jeff Allen, Hadley Wickham, and Rob Hyndman. 2015. *Rmarkdown: Dynamic Documents for R*. http://CRAN.R-project.org/package=rmarkdown.

Hall, Acar, B. G., and M. Barlow. 2013. "Growth Rates Made Easy." *Mol. Biol. Evol.* 31: 232–38. doi:10.1093/molbev/mst197.

Kahm, Matthias, Guido Hasenbrink, Hella Lichtenberg-Frate, Jost Ludwig, and Maik Kschischo. 2010. "Grofit: Fitting Biological Growth Curves with R." *Journal of Statistical Software* 33 (7): 1–21. http://www.jstatsoft.org/v33/i07.

R Core Team. 2015. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. http://www.R-project.org/.

Ritz, C., and J. C. Streibig. 2005. "Bioassay Analysis Using R." *Journal of Statistical Software* 12 (5). http://www.bioassay.dk.

Soetaert, Karline, and Thomas Petzoldt. 2010. "Inverse Modelling, Sensitivity and Monte Carlo Analysis in R Using Package FME." *Journal of Statistical Software* 33 (3): 1–28. http://www.jstatsoft.org/v33/i03/.

Soetaert, Karline, Thomas Petzoldt, and R. Woodrow Setzer. 2010. "Solving Differential Equations in R: Package deSolve." *Journal of Statistical Software* 33 (9): 1–25. http://www.jstatsoft.org/v33/i09.

Xie, Yihui. 2014. "Knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC. http://www.crcpress.com/product/isbn/9781466561595.