A Toolbox for Nonlinear Regression in R: The Package **nlstools**

Florent Baty* Christian Ritz† Sandrine Charles‡ Martin Brutsche§
Jean-Pierre Flandrois¶ Marie-Laure Delignette-Muller‖

Abstract

Nonlinear regression models are applied in a broad variety of scientific fields. Various R functions are already dedicated to fitting such models, among which the function `nls()` has a prominent position. Unlike linear regression fitting of nonlinear models relies on non-trivial assumptions and therefore users are required to carefully ensure and validate the entire modelling. Parameter estimation is carried out using some variant of the least-squares criterion involving an iterative process that ideally leads to the determination of the optimal parameter estimates. Therefore, users need to have a clear understanding of the model and its parameterization in the context of the application and data considered, an *a priori* idea about plausible values for parameter estimates, knowledge of model diagnostics procedures available for checking crucial assumptions, and, finally, an understanding of the limitations in the validity of the underlying hypotheses of the fitted model and its implication for the precision of parameter estimates. Current nonlinear regression modules lack dedicated diagnostic functionality. So there is a need to provide users with an extended toolbox of functions enabling a careful evaluation of nonlinear regression fits. To this end, we introduce a unified diagnostic framework with the R package **nlstools**. In this paper, the various features of the package are presented and exemplified using a worked example from pulmonary medicine.

Keywords: confidence regions, residuals, diagnostic tools, resampling techniques, starting values, 6-minute walk test, nonlinear regression, diagnostic tools, R

*Cantonal Hospital St. Gallen, Lung Center, 9007 St. Gallen, Switzerland, florent.baty@kssg.ch
†University of Copenhagen
‡University of Lyon
§Cantonal Hospital St. Gallen
¶University of Lyon
‖University of Lyon

1 Introduction

Nonlinear regression is used routinely in a wide range of biological disciplines including pharmacology, toxicology, biochemistry, ecology, microbiology and medicine [e.g., Bates and Watts, 1988, Seber and Wild, 1989, Huet et al., 2003, Ritz and Streibig, 2008]. However, statistical software programs do not always include user-friendly routines or modules for fitting nonlinear regression models; this means that researchers often choose to use inappropriate approaches such as polynomial regression or data segmentation (with arbitrary trimming of data), i.e., approaches easily carried out in any statistical software by means of linear regression. On the other hand, specialized commercial programs are available, but they are not always sufficiently flexible or intuitive [Motulsky and Ransnas, 1987].

In addition to limitations in software availability, several other difficulties arise when using nonlinear regression. Like in linear regression, nonlinear regression provides parameter estimates based on the least-squares criterion. However, unlike linear regression, no explicit mathematical solution is available and specific algorithms are needed to solve the minimization problem, involving iterative numerical approximations. Unfortunately, minimization, or optimization in general, is not always a straightforward exercise for such models due to the nonlinear relationships between parameters [Nash and Varadhan, 2011]. Consequently, obtaining useful nonlinear regression model fits may often require careful specification of model details, critical appraisal of the resulting output, and perhaps also use of summary statistics that do not rely too heavily on the model assumptions.

Therefore nonlinear regression may appear to be more daunting than linear regression. It requires a higher degree of user interaction, both for initializing the estimation procedure and for interpreting the results. The package **nlstools** offers tools for addressing these steps when fitting nonlinear regression models using `nls()` (function implemented in the R package **stats**). **nlstools** is available on the Comprehensive R Archive Network at http://CRAN.R-project.org/package=nlstools.

Specifically, there are three key issues that are often causing problems when using nonlinear regression in practice:

1. The iterative estimation procedure requires initial values of the model parameters. These so-called starting values need to be relatively close to the unknown parameter estimates in order to avoid convergence problems where the iterative procedure fails to approach the optimal parameter values. Consequently, a clear understanding of the model features and, in particular, the meaning or interpretation of its parameters would be desirable for ensuring uncomplicated model fitting. In practice researchers may find it difficult to convert such knowledge into an operational format suitable for statistical software programs. Within the statistical environment R [R Core Team, 2013], a number of extension packages provide ways to get around having to come up with starting values. The package **nls2** provides a number of ways to do grid search among candidate starting values and the resulting object may be fed directly into `nls()` [Grothendieck, 2013]. Although the use of a grid search gives the user some flexibility in the definition of the starting values, a range for each model parameter still has to be provided and this may still be a challenge when balancing against the computational burden of an exhaustive search. Specifically for dose-response and growth curve modeling, the packages **drc** [Ritz and Streibig, 2005], **drfit** [Ranke, 2006], and **grofit** [Kahm et al., 2010] among others offer infrastructure for automatically providing data-driven, informed starting values so that the user need not think about providing suitable starting values. The idea behind such self starter routines is explained by Watkins and Venables [2006]. To our knowledge this idea has not been implemented in any other statistical software. However, in many cases no self starter routines are available. This means users may often need to adopt a manual trial-and-error approach in order to ensure an optimal model fit. **nlstools** provides functionality to assist in fitting models.

2. The validity of nonlinear regression model fits must be carefully evaluated by means of appropriate diagnostic and graphical tools. One of the reasons is that sometimes the algorithms used for parameter estimation return sub-optimal estimates, simply because the iterative procedure was not successful in converging to the optimal estimates (often caused by poor starting values or too complex model equations for the data at hand). However, these after-fitting validation steps, which cannot be easily automated, are often neglected because of the lack of dedicated functionality. The package **FSA** (the

fishR project: http://fishr.wordpress.com/packages/) provides some model checking functionality for specific nonlinear regression models (e.g., function `residPlot()`). **nlstools** provides a range of model diagnostics that will work with any `nls()` model fit.

3. Moreover, the standard confidence intervals for model parameters in nonlinear regression models are derived assuming local linearity and normally distributed parameter estimates, e.g., `confint2()` [Ritz and Streibig, 2008, p. 99]. In practice, these assumptions may not always be satisfied; this may in particular be the case for small data sets. For deriving confidence intervals, the `confint()` method in the package **MASS** provides likelihood profiling that does not rely on the linearization step \citep{venables&ripley:2002}. The use of non-parametric resampling techniques for assessing the uncertainty of parameter estimates will even rely less on asymptotic distributions [Shao and Tu, 1996]. **nlstools** provides such a non-parametric alternative.

In a nonlinear regression context there are several other ways to put less emphasis on the distributional assumptions. One is the use of sandwich estimators for the standard errors (in case of suspicion of misspecification of the distribution in general) [Ritz and Streibig, 2008, pp. 83–85]. Another is to use a robust estimation procedure to avoid that singleton data points get to much influence on the model fit, e.g., using the function `nlrob()` in the package **robustbase** [Rousseeuw et al., 2014]. There are also several ways to accommodate non-standard distributional assumptions. The packages **gnm** and **nlme** (the model fitting functions have the same names) allow flexible fitting of various extensions of the nonlinear regression model in terms of the distributions considered for the response as well as the correlation structures needed to describe dependencies between response values, respectively [Turner and Firth, 2007, Pinheiro and Bates, 2000]. However, the challenges in particular related to choosing starting values but also partly concerning model checking (points 2) and 3) above) remain. So **nlstools** offers supplementary functionality that is generally applicable for nonlinear regression analysis.

Section 2 briefly outlines the background for nonlinear regression. In Section 3 we give a detailed introduction to the salient features of **nlstools** using an example from pulmonary medicine. In Section 4 we provide some concluding remarks.

2 Methodology and implementation

2.1 Nonlinear regression

We consider standard nonlinear regression models of the following form:

$$y = f(\theta, x) + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2) \tag{1}$$

with y being the response (the dependent variable), x the (possibly multivariate) independent variable, which is often controlled by the experimenter, θ the vector of model parameters characterizing the relationship between x and y through the function f, and ϵ the residual error term that is assumed to be normally distributed, centered around 0 and with unknown variance (σ^2). Furthermore, we assume that the residual error terms are mutually independent as is usually assumed for standard nonlinear regression analysis [Bates and Watts, 1988]. In R, this nonlinear regression model may be fitted using `nls()` in the standard R installation (the package **stats**). Parameter estimation is based on an iterative procedure that involves a linearization approximation leading to a least-squares problem at each step.

Note that functions `gnls()` and `nlme()` in **nlme** allow fitting of nonlinear regression models for several curves corresponding to different covariate configurations (such as different treatments) and thus necessitating the use of correlation structures (e.g., random effects) [Pinheiro and Bates, 2000]. However, for building these more complex models (i.e., obtaining model fits that converge), `nls()` is often used initially to produce fits of individual curves, which may then subsequently be combined and supplied to enable fitting more complex nonlinear regression models (e.g., through the use of the wrapper `nlsList()`).

2.2 About nlstools

The package **nlstools** provides a number of tools to facilitate fitting standard nonlinear regression models (Equation (1)) and is specifically designed to work directly with `nls()`. The package contains functions and graphical tools that will help users to create `nls()` objects and carry out various diagnostic tests. More specifically, the **nlstools** toolbox will assist users in:

- fitting nonlinear models using function `nls()` by means of graphical tools;
- getting a summary of parameter estimates, confidence intervals, residual standard error and sum of squares, and correlation matrix of the estimates;
- visualizing the fitted curve superimposed on the observations;
- checking the validity of the error model by carrying out tests and graphical checks of residuals;
- inspecting the contours of the residual sum of squares (likelihood contours) to detect possible structural correlations between parameters and the presence of potential local minimum;
- visualizing the projection of confidence regions and investigate the nature of correlations;
- using resampling techniques in order to detect influential observations and obtain non-parametric confidence intervals of the parameter estimates.

We will elaborate on these features in the next section, using a concrete data example from pulmonary medicine.

3 Application in pulmonary medicine

3.1 Oxygen kinetics during 6-minute walk tests

In order to illustrate the features of the package **nlstools**, a worked example is taken from pulmonary medicine \citep[another nonlinear regression example from pulmonary medicine can be found in][]{skjodt&ritz&vethanayagam:2008}. Exercise testing is carried out on patients with a broad range of pulmonary diseases [Schalcher et al., 2003]. The clinical relevance of exercise testing is well established. For instance, the 6-minute walk test (6MWT) is performed, allowing to monitor the change in oxygen uptake over time. It is well known that exercise capacity as assessed by 6MWT correlates with impairment of daily life activities and patients prognosis in several pulmonary conditions [Solway et al., 2001, Tueller et al., 2010]. Peak oxygen uptake has predictive value in patients with pulmonary hypertension and is an indicator of operability in patients with pulmonary diseases.

Data from a typical oxygen uptake kinetics profile are shown in Figure~1. The change of oxygen uptake (VO_2) during exercise testing is classically monitored in 3 distinct phases including a resting phase, the 6-minute exercise testing period, and a recovery period. VO_2 kinetics are classically characterized by a series of parameters including the oxygen uptake in the resting phase (VO_2rest), the maximum oxygen uptake during exercise (VO_2peak), the rate of oxygen increase between VO_2rest and VO_2peak. Subsequent parameters of clinical importance are derived from these initial parameters. Oxygen deficit (O_2def) is defined as the area between an instantaneous increase of oxygen to the maximum upper limit and the observed asymptotic rise of oxygen (Figure 1). Mean response time (MRT) is the time constant of an exponential function describing the rate of oxygen increase. It corresponds to the time needed to attain approximately 63% of the asymptotic value VO_2peak [Sietsema et al., 1994], and is defined as follows: MRT $= O_2\text{def}/\Delta VO_2$, with $\Delta VO_2 = VO_2peak - VO_2rest$.

3.2 Model equation

The length of the resting phase (λ) is controlled by the experimenter and does not need to be estimated. Considering λ constant, the following 3-parameter asymptotic regression model with a lag period (Equation 2) is suitable for describing the first 2 phases of the VO_2 kinetics:

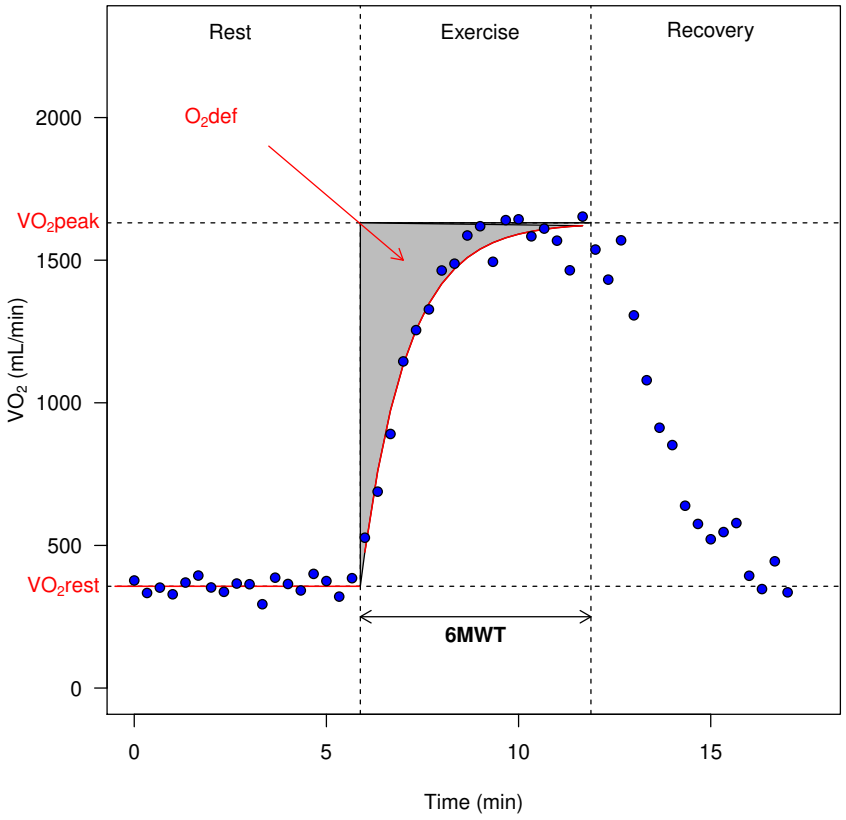

Figure 1: Oxygen uptake kinetics during a 6-minute walk test. This kinetics is characterized by three phases: a resting phase where oxygen is measured at a basal level prior exercise; an exercise phase where oxygen is rising asymptotically until reaching a plateau; a recovery phase where the oxygen uptake is declining towards the baseline asymptotic level.

$$VO_2(t) = \begin{cases} \text{if } t \leq \lambda: & VO_2rest, \\ \text{if } t > \lambda: & VO_2rest + (VO_2peak - VO_2rest)(1 - e^{-(t-\lambda)/\mu}) \end{cases} \quad (2)$$

with VO_2rest the oxygen level during the resting phase, VO_2peak, the maximum oxygen uptake during exercise testing, $\mu > 0$ the rate of change characterizing the steepness of the increase as time (t) elapses (the larger the more steep is the curve), and λ the duration of the resting time controlled by the experimenter. Other examples of segmented regression models are the hockey stick model and the no-effect-concentration model occasionally used in ecotoxicology [Pires et al., 2002]. For the latter, a self-starter routine is available in package **drc** but for the former the user will need to provide starting values by himself as explained by Weisberg [2005, pp. 244–248]. For that specific purpose, the functionality provided by **nlstools** may prove particularly useful.

3.3 Model fitting in R

As mentioned in the introduction, fitting nonlinear regression models requires the provision of starting values for model parameters. A poor choice of starting values may cause non-convergence or convergence to an unwanted local (rather than global) minimum when trying to minimize the least-squares criterion. Biologically interpretable parameter often allows the user to guess adequate starting values by assessing (often graphically) a set of plausible candidate model parameter values. For this purpose, **nlstools** provides the graphical function `preview()`, which can be used to assess the suitability of the chosen starting values, prior to fitting the model. This graphical approach for assessing candidate starting values is also used by Ritz and Streibig [2008, pp. 23–27], but it was not wrapped up in a single function. Below is an example of usage. First, you should specify the model equation to be used in the nonlinear regression as a formula in R. Use this formula as first argument of the function `preview()`, then supply the name of your dataset as second argument, and finally provide a list of values for the model parameters as third argument. An additional argument `variable` can be used to specify which independent variable is plotted against the dependent variable (column index of the original dataset; default is 1) when more than one independent variable is modeled.

```
library("nlstools")
```

```
##
## 'nlstools' has been loaded.
```

```
## IMPORTANT NOTICE: Most nonlinear regression models and data set examples
```

```
## related to predictive microbiolgy have been moved to the package 'nlsMicrobio'
```

```
formulaExp <- as.formula(VO2 ~ (t <= 5.883) * VO2rest + (t > 5.883) *
             (VO2rest + (VO2peak - VO2rest) *
              (1 - exp(-(t - 5.883) / mu))))
preview(formulaExp, data = O2K,
    start = list(VO2rest = 400, VO2peak = 1600, mu = 1))
```

```
##
## RSS:  149000
```

Both the oxygen levels during the resting phase (the parameter VO_2rest) and the maximum oxygen level reached during the 6MWT (the parameter VO_2peak) can be retrieved directly in an approximate fashion from Figure 1, whereas μ is simply initially set to 1. Note that the length of the resting phase ($\lambda = 5.883$ min) was hardcoded into the above formula. Figure 2 shows good agreement between the data and the theoretical model based on the provided set of starting values. Judged by the figure, the chosen starting values seem to be suitable for initializing `nls()`. Note that next to the plot, the residual sum of squares measuring the discrepancy between the model (based on the chosen starting values) and the observed data is provided. This value gives an idea of the magnitude of the residual sum of squares to expect from the model fit based on `nls()`.

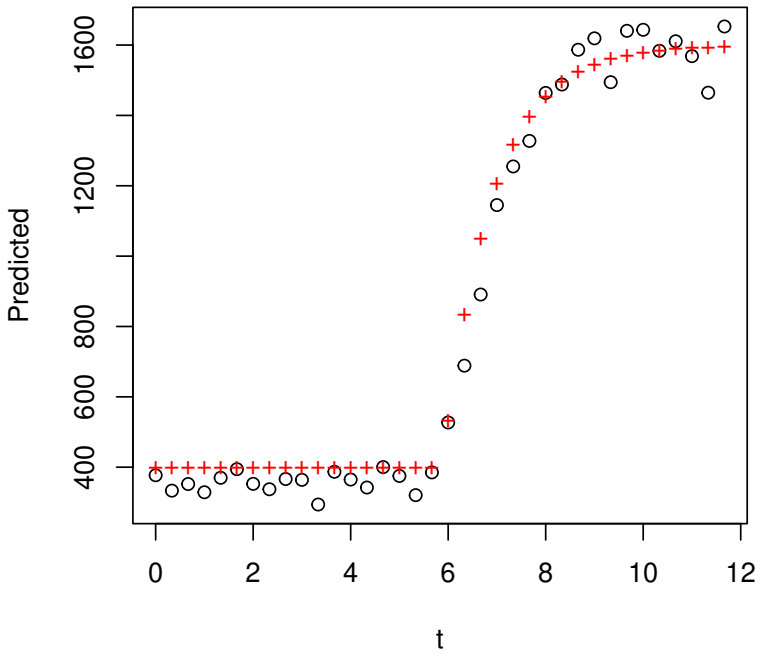

Figure 2: Graphically assessing the starting values prior the fit of a nonlinear model.

```
O2K.nls1 <- nls(formulaExp, start = list(VO2rest = 400, VO2peak = 1600,
                  mu = 1), data = O2K)
```

Once suitable starting values are obtained, the model may be fitted using `nls()` and then the function `overview()` in **nlstools** may be used for providing a single display with all relevant pieces of information about the model fit.

Specifically, `overview()` returns output containing:

- The parameter estimates with the corresponding estimated standard errors, t-test statistics (estimate/standard error) for evaluating null hypotheses that the model parameters could be equal to 0 ($H_0 : \theta = 0$) along with the corresponding $p-$values calculated using a t distribution as reference distribution (for the present example the t distribution with 33 degrees of freedom was used). The residual sum of squares $RSS_{min} = 81200$ and the residual standard error ($\sqrt{RSS_{min}/33} = 49.6$) are also reported, reflecting the variation within the walk test that is due to the device used. The number of steps needed for finding the parameters is also reported (numbers $> 10 - 20$ are often indicative of poor starting values and/or too complex model equation in view of the sample size). This output is similar to the one from the `summary()` method available for `nls()` fits [Ritz and Streibig, 2008, p. 12].
- The corresponding 95% t-based confidence intervals (in this case percentiles from the t distribution with 33 degrees of freedom), similar to the intervals obtained using the default `confint2()` method in the package **nlrwr**. Accordingly reported p-values and confidence intervals are in agreement. We refer to Huet et al. [2003, pp. 32-33] for a detailed derivation.
- The estimated correlation matrix is reported. This piece of output allows assessment of the degree of correlation between the parameter estimates in order to detect highly correlated parameters that may indicate redundancies and perhaps point towards simplification of the model equation. In our example, the highest correlation (between μ and VO_2peak) is 0.76, which does not indicate any problems, but merely is a consequence of these two parameters being entangled in the same term in Equation 2.

```
overview(O2K.nls1)
```

```
##
```

```
## ------
## Formula: VO2 ~ (t <= 5.883) * VO2rest + (t > 5.883) * (VO2rest + (VO2peak -
##     VO2rest) * (1 - exp(-(t - 5.883)/mu)))
##
## Parameters:
##          Estimate Std. Error t value Pr(>|t|)
## VO2rest 3.568e+02  1.141e+01   31.26   <2e-16 ***
## VO2peak 1.631e+03  2.149e+01   75.88   <2e-16 ***
## mu      1.186e+00  7.661e-02   15.48   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 49.59 on 33 degrees of freedom
##
## Number of iterations to convergence: 5
## Achieved convergence tolerance: 7.598e-06
##
## ------
## Residual sum of squares: 81200
##
## ------
## t-based confidence interval:
##                2.5%       97.5%
## VO2rest  333.537401  379.980302
## VO2peak 1587.155300 1674.611703
## mu         1.030255    1.342002
##
## ------
## Correlation matrix:
##            VO2rest    VO2peak        mu
## VO2rest 1.00000000 0.07907046 0.1995377
## VO2peak 0.07907046 1.00000000 0.7554924
## mu      0.19953773 0.75549241 1.0000000
```

In order to facilitate the visualization of the model fit together with the data, **nlstools** provides the function `plotfit()`, which offers functionality similar to `abline()` with a simple linear regression model fit as argument. Thus `plotfit()` avoids manual definition of a grid of values for the independent variable, subsequent prediction, and use of `lines()`.

```
plotfit(O2K.nls1, smooth = TRUE)
```

The function superimposes the fitted curve on top of the plot of the data (Figure~3).

Notice that the argument `smooth = TRUE` provides a smoothed representation of the fitted regression curve. This option is only available when one single (one-dimensional) independent variable is involved. For plots of a model fit involving more than one independent variable (e.g., see worked example `michaelis` in **nlstools**), it is necessary to specify the argument `variable` in the function `plotfit()` to indicate which variable is used for the x axis. In such a case, no smoothing is possible as it would also depend on the other independent variables, i.e., `smooth = FALSE`.

3.4 Assessing the goodness of fit through the residuals

An examination of the quality of the obtained nonlinear regression model fit may be based on the residuals calculated from the fit as follows:

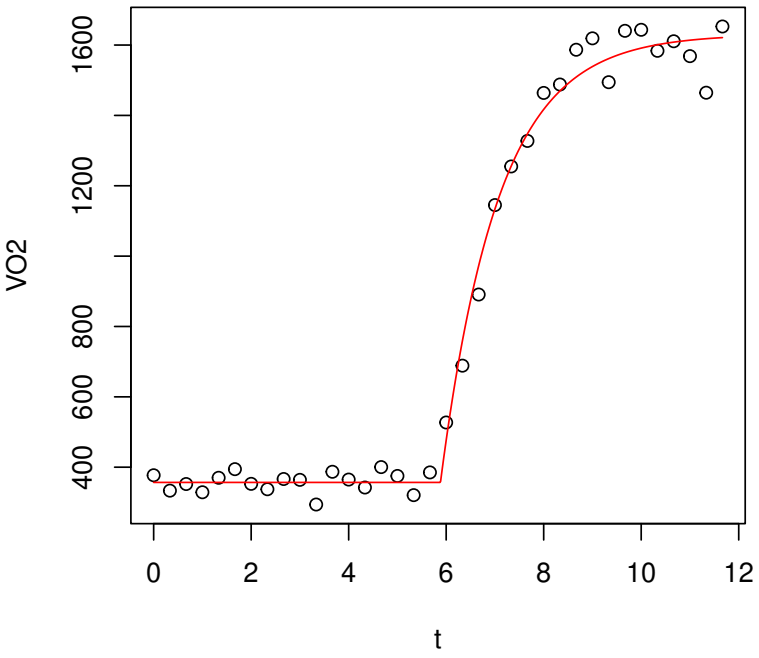

Figure 3: Plot of the data (dependent vs. independent variable) with the fitted model superimposed.

$$\hat{\epsilon} = y - f(\hat{\theta}, x)$$

Standardized residuals are obtained by dividing the centered residuals by the residual standard error. **nlstools** provides the function `nlsResiduals()`, which extracts the residuals from an `nls` object.

The corresponding `plot()` method allows a convenient display of the diagnostic plots outlined by Ritz and Streibig [2008]. Specifically, `plot()` produces by default a four-panel display:

- Top left panel: The plot of raw residuals against fitted values is useful for assessing whether or not the chosen model equation is appropriate (the scatter is similar above and below the horizontal axis along the range of fitted values in case of an appropriate model equation). This plot is similar to the one obtained for linear models by using `plot(lmFit, which = 1)`.
- Top right panel: The plot of the standardized residuals vs. the fitted values is useful for evaluation if there is any indication of variance inhomogeneity, which would show up as an uneven spread across the range of the fitted values.
- Bottom left panel: The plot of each raw residual vs. the previous raw residual (lag one) may be useful to detect correlation along the scale of the independent variable (to be meaningful it requires the data to be order in increasing order according to the independent variable). A systematic departure away from a random scatter around the x axis is indicative of correlation among the values of the dependent variable. This may often be the case if the independent variable corresponds to some kind of time scale. For more details we refer to Glasbey [1979] and Ritz and Streibig [2008, pp. 69–70].
- Bottom right panel: The normal probability plot (or QQ plot) compares the standardized residuals vs. the theoretical values from a standard normal distribution, both of which are expected to range from -2 and -2 for most of the values. This functionality is similar to what is available for linear models, e.g., using `plot(lmFit, which = 2)`.

The argument `which` may be used to choose which diagnostic plots should be shown (there are 6 options in total as explained in the help page). For the model fit `O2K.nls1` the resulting plots are shown in Figure 4.

```
O2K.res1 <- nlsResiduals(O2K.nls1)
plot(O2K.res1)
```

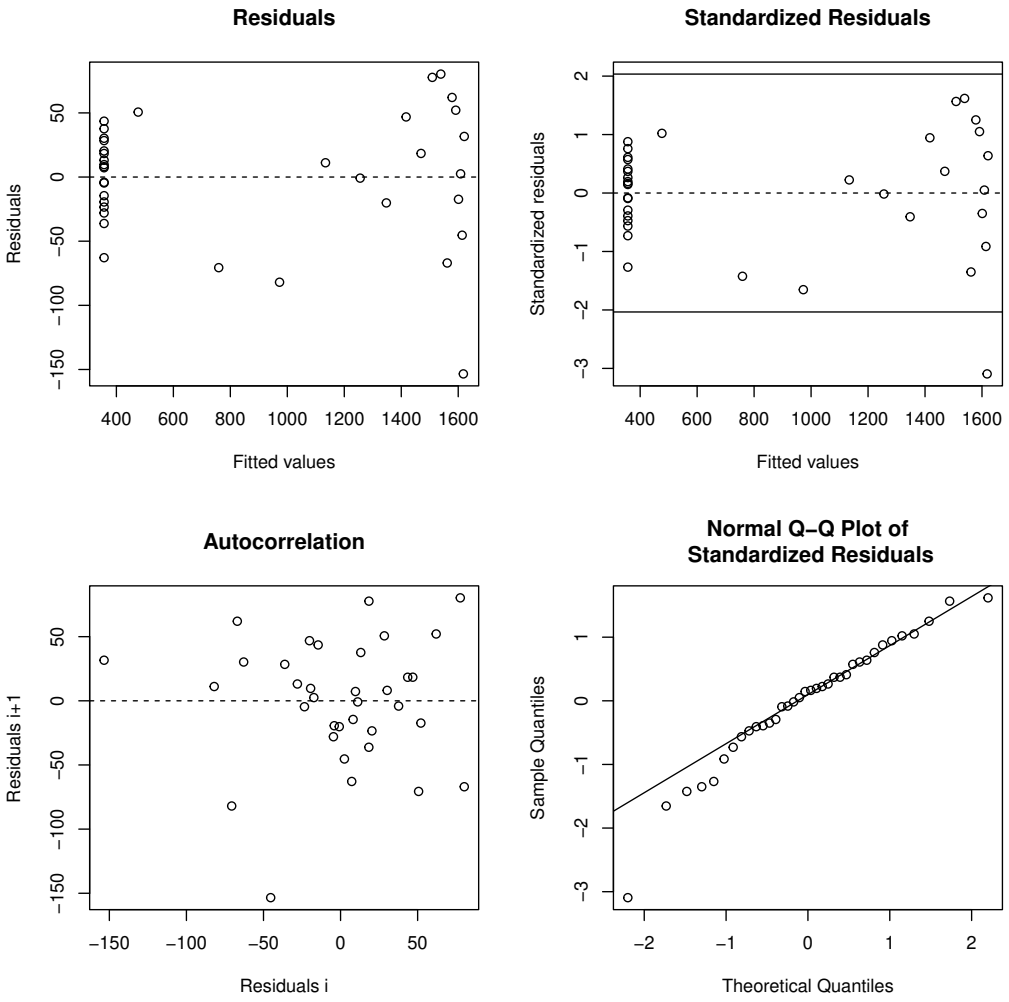

Figure 4: Plot of residuals. The top left panel shows the raw residuals vs. the fitted values. The top right panel shows the standardized residuals (with mean $\mu = 0$ and standard deviation $\sigma = 1$) vs. the fitted values. The bottom left panel shows the autocorrelation plot and the bottom right panel the QQ plot of the standardized residuals.

Figure 4 shows no indication of problems with the model assumptions as the residuals seem to be approximately normally distributed (a clear alignment along the diagonal in the QQ plot) and without evidence of autocorrelation or heteroscedastic variance.

In addition to the visual assessment of the model assumptions, the normality of residuals may be evaluated using the Shapiro-Wilk test [Ritz and Streibig, 2008, p. 69]. This test is one of the most powerful tests of normality (the function `shapiro.test()` is part of the package **stats** in the standard R installation). Similarly, the lack of autocorrelation in residuals may be assessed by means of the runs test [e.g., Motulsky and Ransnas, 1987, López et al., 2000, Motulsky and Christopoulos, 2004], using the function `runs.test()` in the package **tseries** [Trapletti and Hornik, 2013]. However, note that this is not a very strong test as it essentially only utilizes the signs of the residuals but not their actual values \citep{ritz&martinussen:2011}. We consider these tests as supplements that are occasionally useful next to the routine visual assessment of the model assumptions. Both tests are available through the function `test.nlsResiduals()`.

```
test.nlsResiduals(O2K.res1)
```

```
##
## ------
##  Shapiro-Wilk normality test
##
## data:  stdres
## W = 0.95205, p-value = 0.1214
##
##
## ------

##
##  Runs Test
##
## data:  as.factor(run)
## Standard Normal = 0.76123, p-value = 0.4465
## alternative hypothesis: two.sided
```

In our example, the null hypothesis of normal distribution could not be rejected (Shapiro-Wilk test: $p = 0.12$) and there was also no indication of autocorrelation (runs test: $p = 0.45$).

3.5 Confidence regions

We define the $1 - \alpha$ joint confidence region for the model parameters by means of the following inequality. A given set of parameters θ is included in the confidence region if the corresponding residual sum of squares (RSS) is lying within the margin defined in the following Equation~3 (often referred to as Beale's criterion).

$$RSS(\theta) < RSS_{min} \left[1 + \frac{p}{n-p} F_{1-\alpha}(p, n-p) \right] \tag{3}$$

with RSS_{min} the minimum residual sum of squares obtained from the least-squares estimation (previously defined for the function `overview()`), $F_{1-\alpha}$ the appropriate quantile of the F-distribution with $(p, n-p)$ degrees of freedom, where n is the number of observations and p the number of model parameters in f [Beale, 1960, Bates and Watts, 1988]. Two functions are implemented in **nlstools** for visualizing the joint confidence region defined in Equation~(3), one for showing contours and another one for showing projections.

For each pair of parameters the function `nlsContourRSS()` provides two-dimensional contours of the p-dimensional joint confidence region using a grid while keeping the remaining $p - 2$ parameters fixed at their least-squares estimates. The number of contour levels is defined by the user using the argument `nlev`. The RSS contours can be used both to assess the structural correlation among model parameters (a graphical analog to the correlation matrix) and to detect the presence of unexpected multiple minima, which could indicate that sub-optimal parameter estimates were obtained and perhaps the model should be fitted again with different starting values.

For the model fit `O2K.nls1`, Figure~5 (left panel) shows the RSS contours for the three pairs of two model parameters. The resulting two-dimensional 95% confidence regions, which correspond to specific contours are also shown (in dotted red lines). These contours, which are expected to be close to elliptical curves as long as the error model is valid, allow an evaluation of the two-dimensional confidence regions as compared to the one-dimensional confidence intervals that are routinely used. In particular, we get an insight on the extent of overlap between one-dimensional intervals and two-dimensional regions. For instance, for the pair μ and $VO2peak$, the projections of the elliptical confidence region onto the axes result in marginal confidence intervals that are wider as compared to the standard one-dimensional confidence intervals shown in the output from `overview()` on page 7. This means that standard one-dimensional confidence intervals seem to be too narrow (insufficient coverage).

```
O2K.cont1 <- nlsContourRSS(O2K.nls1)
plot(O2K.cont1, col = FALSE, nlev = 5)
O2K.conf1 <- nlsConfRegions(O2K.nls1, exp = 2, length = 2000)
plot(O2K.conf1, bounds = TRUE)
```

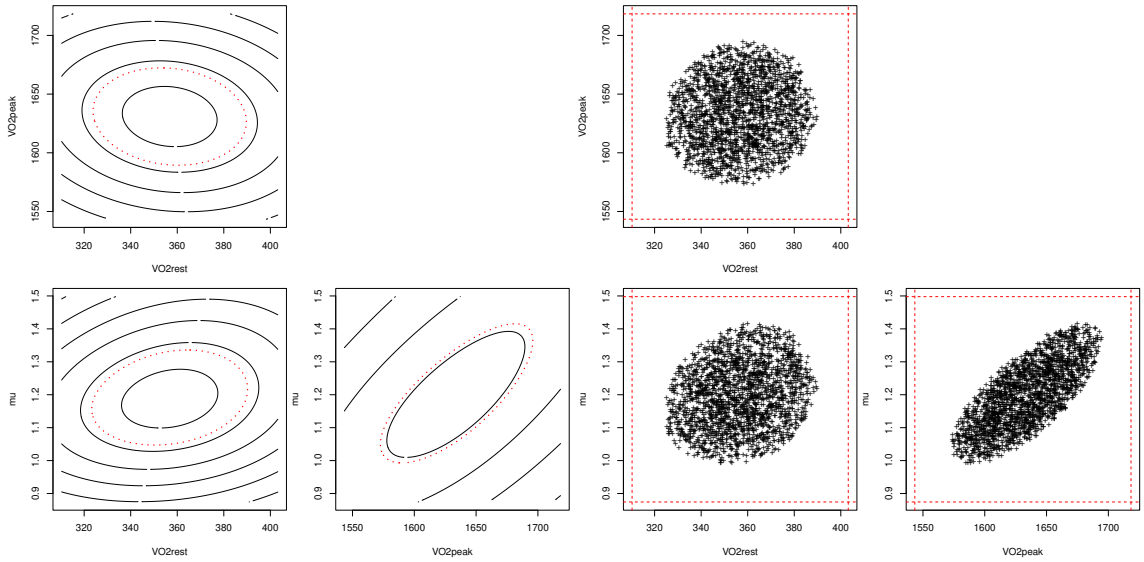

Figure 5: The left panel displays the contours based on the residual sum of squares. The contours represented by a red dotted line correspond to the section of the Beale's 95% confidence region. The right panel shows the projections of the confidence region according to the Beale's criterion. The dashed red frames around the confidence regions correspond to the limits of the sampling regions.

The function `nlsConfRegions()` allows users to plot another representation of the Beale's confidence region, also known as joint parameter likelihood region [Bates and Watts, 1988]. The method consists in randomly sampling parameter values in a hypercube centered around the least-squares estimates. A set of parameters is acceptable if the resulting residual sum of squares satisfies Beale's criterion (Equation 3). As soon as the specified number of points to be in the confidence region is reached (argument `length` in `nlsConfRegions()`), the iterative sampling is stopped. The confidence region is then plotted by projection of the sampled points in planes defined by each pair of model parameters (Figure 5, right panel). It is often necessary to zoom in or out the sampling region in order to get a better view of the overall projected region. This is done by changing argument `exp` of function `nlsConfRegions()`. The sampling region can be visualized by setting argument `bound = TRUE` in the generic plotting function `plot.nlsConfRegions()`.

It is worth noticing that the representation of the confidence region by contours does not provide exactly the same information as the representation by projections when the number of parameters is greater than two. Representations of confidence regions by contours provide smaller confidence regions than confidence regions based on projections, because the former does not incorporate the uncertainty in the $p-2$ parameter estimates left out. Therefore, representations by contours tend to slightly underestimate the size of the confidence region.

In our example, contours are perfectly elliptical with a global minimum at the center, which is an indication of a good nonlinear regression model fit. The narrower elliptic shape of Beale's confidence region between VO_2peak and μ reflects the relatively high correlation between these two parameters (correlation: 0.76).

3.6 Resampling techniques

Both jackknife and bootstrap procedures applied to nonlinear regression are implemented in **nlstools**. Jackknife is implemented via the function `nlsJack()`. By using a leave-one-out procedure, it produces

jackknife parameter estimates together with confidence intervals [Quenouille, 1956, Fox et al., 1980, Seber and Wild, 1989]. It can also be used to assess the influence of each observation on each parameter estimates.

```
O2K.jack1 <- nlsJack(O2K.nls1)
summary(O2K.jack1)
```

```
##
## ------
## Jackknife statistics
##           Estimates        Bias
## VO2rest  356.531533 0.227317890
## VO2peak 1628.736188 2.147313511
## mu         1.184515 0.001613564
##
## ------
## Jackknife confidence intervals
##                   Low          Up
## VO2rest  342.2881464  370.774920
## VO2peak 1560.1022374 1697.370138
## mu         0.9923505    1.376679
##
## ------
## Influential values
## * Observation 21 is influential on VO2peak
## * Observation 34 is influential on VO2peak
## * Observation 35 is influential on VO2peak
## * Observation 20 is influential on mu
## * Observation 21 is influential on mu
## * Observation 35 is influential on mu
```

The generic function `summary()`, applied to an object produced by the function `nlsJack()`, returns the jackknife parameter estimates together with the associated bias and confidence intervals and, if applicable, a list of influential observations that are identified based on DFBETAs defined as follows:

$$\text{DFBETA}(i,j) = \frac{|\hat{\theta}_j^{-i} - \hat{\theta}_j|}{se(\hat{\theta}_j)} \tag{4}$$

with $\hat{\theta}_j$ the estimate of the j^{th} parameter based on the original dataset, $\hat{\theta}_j^{-i}$ the estimate of the j^{th} parameter based on the dataset without the i^{th} observation, and $se(\hat{\theta}_j)$ the standard error of the j^{th} parameter estimate based on the original dataset.

An observation is empirically denoted as influential for one parameter if the absolute difference between parameter estimates with and without this observation exceeds twice the standard error of the estimate divided by the square root of the number of observations [Belsley et al., 1980]. Applied to our dataset, three observations appear to have a substantial influence on VO_2peak estimate and, similarly, three other observations are influencing μ estimate (Figure 6, left panel).

The function `nlsBoot()` uses non-parametric bootstrap of mean centered residuals to obtain a given number (argument `niter`) of bootstrap estimates \citep[Chapter~8]{venables&ripley:2002}. Bootstrap estimates, standard errors together with median and percentile confidence intervals (2.5% and 97.5% percentiles of bootstrapped estimates) \citep[pp.~225-226]{venables&ripley:2002} are displayed by the generic function `summary()`. The associated plotting function can be used both for a pairwise representation of the bootstrap estimates or, as shown in Figure 6 (right panel), for a boxplot representation of the distribution of each bootstrapped parameter.

```
O2K.boot1 <- nlsBoot(O2K.nls1)
summary(O2K.boot1)
```

```
##
## ------
## Bootstrap statistics
##            Estimate  Std. error
## VO2rest  356.700477 11.05574879
## VO2peak 1632.066916 20.50067919
## mu         1.188432  0.07750468
##
## ------
## Median of bootstrap estimates and percentile confidence intervals
##               Median        2.5%      97.5%
## VO2rest  357.307214  333.651644  377.57646
## VO2peak 1633.028113 1590.224014 1672.41445
## mu         1.186141    1.041741    1.35041
```

```
plot(O2K.jack1)
plot(O2K.boot1, type = "boxplot")
```

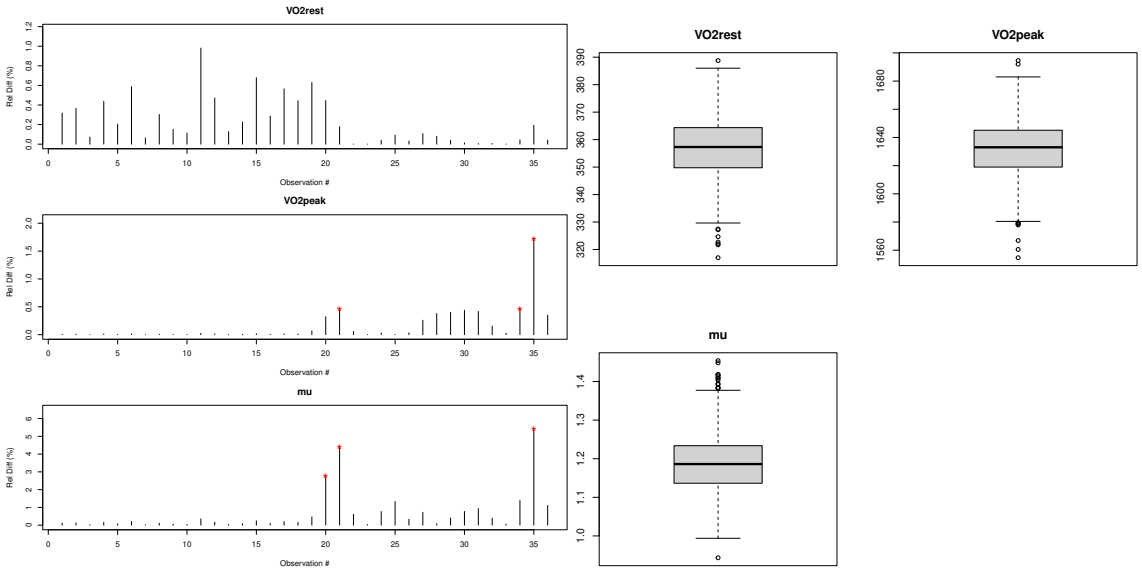

Figure 6: Resampling procedures. The left panel shows the influence of each observation on each parameter estimate according to a jackknife procedure using `nlsJack()`. The right panel shows the boxplot distribution of the bootstrapped parameter estimates obtained using `nlsBoot()`.

In some cases, problems of convergence may arise during the bootstrap resampling procedure, when the model cannot be fitted to the resampled data. If the fitting procedure fails less than 50% of cases, the bootstrap statistic is provided with a warning indicating the percentage of times the procedure successfully converged; otherwise the procedure is interrupted with an error message and no result is given.

The comparison of confidence intervals based on the t-based approximation, previously obtained using `overview()`, and based on resampling procedures (jackknife or bootstrap) is illustrated in Figure 7. In that case, it shows that bootstrap confidence intervals are comparable to the t-based ones, providing slightly narrower confidence intervals for all three parameters. On the other hand, jackknife confidence intervals noticeably differ from the other two intervals. This was to be expected considering that jackknife is using only limited information about the statistic and is therefore less efficient than bootstrap [Efron, 1979]. In addition,

jackknife is resampling sequentially individuals whereas bootstrap resamples residuals with replacement. Therefore, we tentatively recommend to use the bootstrap procedure for the assessment of confidence intervals of parameter estimates, whereas the jackknife procedure should be specifically used for the detection of influential observations. It is worth noting that function `confint()` from the default R package **stats** [R Core Team, 2013] can also be used to build confidence intervals by profiling the residual sum of squares. However, this method often fails to give any result due the lack of convergence of the optimization algorithm for at least one explored value of one of the parameters. Unlike the function `confint()`, `nlsBoot()` provides confidence intervals even if the optimization algorithm fails to converge for some of the bootstrapped samples.

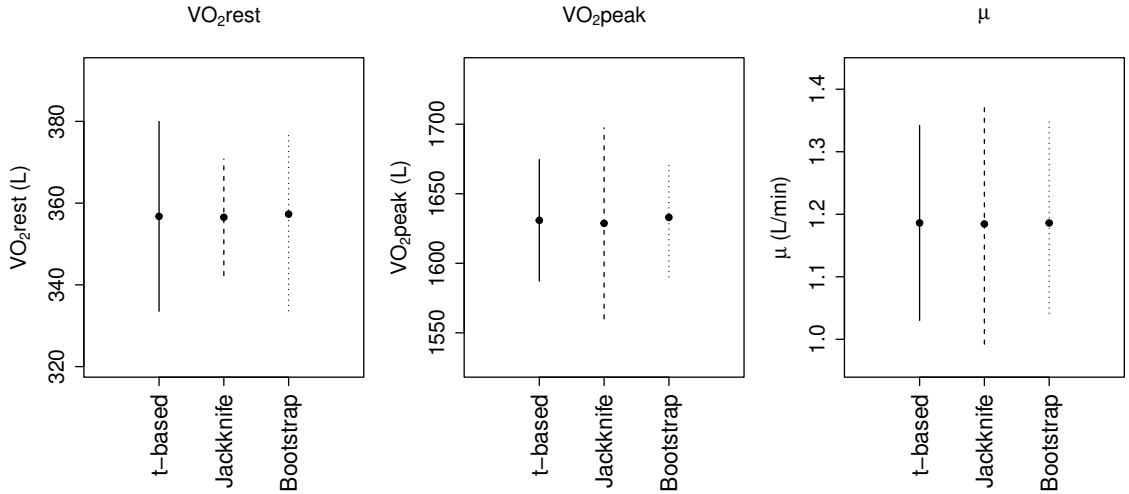

Figure 7: Comparison of parameter confidence intervals obtained by *t*-based and resampling (jackknife/bootstrap) methods.

4 Concluding remarks

We have shown the usefulness of **nlstools** for the important steps in a nonlinear regression analysis. The proposed functionality is generally applicable as seen from the variety of different areas where it has already been used successfully: biological chemistry [Slupe et al., 2013], environmental toxicology [Devos et al., 2012], forest ecology [Kapeller et al., 2012, Bošeľa et al., 2013], marine science [Cabral et al., 2013, Villegas-Ríos et al., 2013], microbiology [Baty and Delignette-Muller, 2004, Delignette-Muller, 2009, Ohkochi et al., 2013, Kiermeier et al., 2013, Tang et al., 2013], limnology [Volta et al., 2013], and plant biology [Matter et al., 2012].

Further developments of **nlstools** are envisaged, including support for self-starting nonlinear regression models. It would also be useful to provide additional sensitivity procedures in order to allow users to further minimize the risk of undesirable convergence results towards local minima during the optimization procedure (e.g., an annotated and detailed report of the entire convergence process). We would also like to extend the diagnostic tools to situations where high-throughput nonlinear regression analyses are carried out, [e.g., Stanzel et al., 2013], through the implementation of some automatic diagnostic checks.

References

D. Bates and D. G. Watts. *Nonlinear Regression Analysis and Its Applications*. John Wiley & Sons, Chichester, UK, 1988.

F. Baty and M. L. Delignette-Muller. Estimating the Bacterial Lag Time: Which Model, Which Precision? *International Journal of Food Microbiology*, 91(3):261–277, Mar 2004.

E. M. L. Beale. Confidence Regions in Non-Linear Estimation. *Journal of the Royal Statistical Society B*, 22 (1):41–88, 1960.

D. A. Belsley, E. Kuh, and R. E. Welsch. *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. John Wiley & Sons, New York, 1980.

Michal Bošeľa, Rudolf Petráš, Vladimír Šebeň, Julián Mecko, and Róbert Marušák. Evaluating competitive interactions between trees in mixed forests in the western carpathians: Comparison between long-term experiments and sibyla simulations. *Forest Ecology and Management*, 310:577–588, 2013.

Reniel B Cabral, Porfirio M Aliño, and May T Lim. A coupled stock-recruitment-age-structured model of the north sea cod under the influence of depensation. *Ecological Modelling*, 253:1–8, 2013.

M. L. Delignette-Muller. Principle of predictive modeling. In F. Toldrá, editor, *Safety of Meat and Processed Meat*, pages 535–558. Springer-Verlag, New York, 2009.

Alexandre Devos, Claire Voiseux, Christelle Caplat, and Bruno Fievet. Effect of chronic exposure to zinc in young spats of the pacific oyster (*Crassostrea gigas*). *Environmental Toxicology and Chemistry*, 31(12): 2841–2847, 2012.

B. Efron. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

Terry Fox, David Hinkley, and Kinley Larntz. Jackknifing in nonlinear regression. *Technometrics*, 22(1): 29–33, 1980.

C. A. Glasbey. Correlated residuals in non-linear regression applied to growth data. *Applied Statistics*, 28(3): 251–259, 1979.

G. Grothendieck. ***nls2**: Non-Linear Regression with Brute Force*, 2013. URL http://CRAN.R-project.org/package=nls2. R package version 0.2.

S. Huet, A. Bouvier, M. A. Poursat, and E. Jolivet. *Statistical Tools for Nonlinear Regression: A Practical Guide with S-PLUS and R Examples*. Springer-Verlag, Berlin, Heidelberg, New York, 2003.

Matthias Kahm, Guido Hasenbrink, Hella Lichtenberg-Fraté, Jost Ludwig, and Maik Kschischo. **grofit**: Fitting biological growth curves with R. *Journal of Statistical Software*, 33(7):1–21, 2010. URL http://www.jstatsoft.org/v33/i07/.

Stefan Kapeller, Manfred J Lexer, Thomas Geburek, Johann Hiebl, and Silvio Schueler. Intraspecific variation in climate response of norway spruce in the eastern alpine range: Selecting appropriate provenances for future climate. *Forest Ecology and Management*, 271:46–57, 2012.

Andreas Kiermeier, Mark Tamplin, Damian May, Geoff Holds, Michelle Williams, and Alison Dann. Microbial growth, communities and sensory characteristics of vacuum and modified atmosphere packaged lamb shoulders. *Food Microbiology*, 36(2):305–315, Dec 2013.

S. López, J. France, W. J. J. Gerrits, M. S. Dhanoa, D. J. Humphries, and J. Dijkstra. A generalized Michaelis-Menten equation for the analysis of growth. *Journal of Animal Science*, 78(7):1816–1828, 2000.

P Matter, CJ Kettle, J Ghazoul, T Hahn, and AR Pluess. Evaluating contemporary pollen dispersal in two common grassland species *Ranunculus bulbosus* l. (ranunculaceae) and *Trifolium montanum* l. (fabaceae) using an experimental approach. *Plant Biology*, 15(3):583–592, May 2012.

H. J. Motulsky and L. A. Ransnas. Fitting Curves to Data Using Nonlinear Regression: A Practical and Nonmathematical Review. *The FASEB Journal*, 1(5):365–374, 1987.

Harvey J. Motulsky and Arthur Christopoulos. *Fitting Models to Biological Data Using Linear and Nonlinear Regression: A Practical Guide to Curve Fitting*. Oxford University Press, Oxford, 2004.

John C. Nash and Ravi Varadhan. Unifying Optimization Algorithms to Aid Software System Users: **optimx** for R. *Journal of Statistical Software*, 43(9):1–14, 2011. URL http://www.jstatsoft.org/v43/i09/.

Miho Ohkochi, Shigenobu Koseki, Masaaki Kunou, Katsuaki Sugiura, and Hirokazu Tsubone. Growth modeling of *Listeria monocytogenes* in pasteurized liquid egg. *Journal of Food Protection*, 76(9):1549–1556, 2013.

J. C. Pinheiro and D. M. Bates. *Mixed-Effects Models in S and S-PLUS*. Springer-Verlag, New York, 2000.

Ana M. Pires, Joao A. Branco, Ana Picado, and Elsa Mendonca. Models for the Estimation of a 'No Effect Concentration'. *Environmetrics*, 13(1):15–27, 2002.

M. Quenouille. Notes on Bias in Estimation. *Biometrika*, 43(3-4):353–360, 1956.

J. Ranke. Fitting Dose Response Curves from Bioassay and Toxicity Testing. *R News*, 6:7–12, 2006.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013. URL http://www.R-project.org/. ISBN 3-900051-07-0.

C. Ritz and J. C. Streibig. *Nonlinear Regression with R*. Springer-Verlag, New York, 2008.

Christian Ritz and Jens C. Streibig. Bioassay analysis using R. *Journal of Statistical Software*, 12(5):1–22, 1 2005. ISSN 1548-7660. URL http://www.jstatsoft.org/v12/i05.

Peter Rousseeuw, Christophe Croux, Valentin Todorov, Andreas Ruckstuhl, Matias Salibian-Barrera, Tobias Verbeke, Manuel Koller, and Martin Maechler. ***robustbase**: Basic Robust Statistics*, 2014. URL http://CRAN.R-project.org/package=robustbase. R package version 0.91-1.

C. Schalcher, H. Rickli, M. Brehm, D. Weilenmann, E. Oechslin, W. Kiowski, and H. P. Brunner-La Rocca. Prolonged oxygen uptake kinetics during low-intensity exercise are related to poor prognosis in patients with mild-to-moderate congestive heart failure. *Chest*, 124(2):580–586, 2003.

G. A. F. Seber and C. J. Wild. *Nonlinear Regression*. John Wiley & Sons, New York, 1989.

J. Shao and D. Tu. *The Jackknife and Bootstrap*. Springer-Verlag, New York, 1996.

K. E. Sietsema, I. Ben-Dov, Y. Y. Zhang, C. Sullivan, and K. Wasserman. Dynamics of Oxygen Uptake for Submaximal Exercise and Recovery in Patients with Chronic Heart Failure. *Chest*, 105(6):1693–1700, Jun 1994.

A. M. Slupe, R. A. Merrill, K. H. Flippo, M. A. Lobas, J. C. Houtman, and S. Strack. A Calcineurin Docking Motif (LXVP) in Dynamin-Related Protein 1 Contributes to Mitochondrial Fragmentation and Ischemic Neuronal Injury. *Journal of Biological Chemistry*, 288(17):12353–12365, Apr 2013.

S. Solway, D. Brooks, Y. Lacasse, and S. Thomas. A Qualitative Systematic Overview of the Measurement Properties of Functional Walk Tests Used in the Cardiorespiratory Domain. *Chest*, 119(1):256–270, Jan 2001.

Sven Stanzel, Marc Weimer, and Annette Kopp-Schneider. Data management in large-scale collaborative toxicity studies: How to file experimental data for automated statistical analysis. *Toxicology in Vitro*, 27 (4):1402–1409, 2013.

S. Tang, M. J. Stasiewicz, M. Wiedmann, K. J. Boor, and T. M. Bergholz. Efficacy of Different Antimicrobials on Inhibition of *Listeria monocytogenes* Growth in Laboratory Medium and on Cold-Smoked Salmon. *International Journal of Food Microbiology*, 165(3):265–275, Aug 2013.

Adrian Trapletti and Kurt Hornik. ***tseries**: Time Series Analysis and Computational Finance*, 2013. URL http://CRAN.R-project.org/package=tseries. R package version 0.10-32.

C. Tueller, L. Kern, A. Azzola, F. Baty, S. Condrau, J. Wiegand, M. Tamm, and M. Brutsche. Six-Minute Walk Test Enhanced by Mobile Telemetric Cardiopulmonary Monitoring. *Respiration*, 80(5):410–418, 2010.

Heather Turner and David Firth. **gnm**: A package for generalized nonlinear models. *R News*, 7:8–12, 2007.

David Villegas-Ríos, Alexandre Alonso-Fernández, Mariña Fabeiro, Rafael Bañón, and Fran Saborido-Rey. Demographic variation between colour patterns in a temperate protogynous hermaphrodite, the ballan wrasse *Labrus bergylta*. *PLoS ONE*, 8(8):e71591, 2013.

Pietro Volta, Erik Jeppesen, Barbara Campi, Paolo Sala, Matthias Emmrich, and Ian Winfield. The population biology and life history traits of eurasian ruffe [*Gymnocephalus cernuus* (l.), pisces: Percidae] introduced into eutrophic and oligotrophic lakes in northern italy. *Journal of Limnology*, 72(2):e22, 2013.

P. Watkins and W. N. Venables. Non-Linear Regression for Optimizing the Separation of Carboxylic Acids. *R News*, 6:2–7, 2006.

Sanford Weisberg. *Applied Linear Regression*. John Wiley & Sons, New York, 3rd edition, 2005.