Quickstart Manual for Package cardidates

Thomas Petzoldt, René Sachse and Susanne Rolinski

Abstract

The **cardidates** package provides methods for fitting four resp. six parametric Weibull functions to environmental data. Heuristics are applied for finding initial parameters in the six parametric case. As a second step, the fitted functions can be used to identify maximum, beginning and end of peaks by using a quantile-like approach.

Keywords: ecological time series, peak fitting, cardinal date, reproducible research.

This manual describes the R package **cardidates**, version 0.3-4, November 19, 2007.

1. Introduction

Phenology and seasonal succession in aquatic, and of course also terrestrial, ecosystems are strongly dependent on physical factors. In order to promote investigations into this coupling, objective and reliable methods of characterising annual time series are important.

The **cardidates** package provides methods for curve fitting "peaks" of environmental data and approaches for an "objective" characterisation of what we call "cardinal dates", i.e. beginning, time of maximum and time of end of an identified peak. Objectivity means that it is not necessarily important to get the "true" values (what is true?) for one particular time series but to get reproducible results independently of the person who performs the analysis.

The proposed methods were developed within the AQUASHIFT research program and used to determine the beginning, maximum and end of the spring mass development of phytoplankton in different lakes and water reservoirs. These time points, that we call "cardinal dates", can be analysed for temporal trends and relationships to climate variables.

The complete methodology is described in Rolinski, Horn, Petzoldt, and Paul (2007). Until now we implemented only the most reliable approach using Weibull-functions (Method B in the article); other functions may follow.

The methodology may also be useful for other ecological time series (e.g. bacteria, protozoa, insects or small mammals). Please don't hesitate to contact the authors if you feel that this package should be generalized to other processes.

2. Pre-requisites

You need the following software:

- a current version of the statistical data analysis system, R, (R Development Core Team 2007) that can be downloaded from `http://www.r-project.org`.
- the **cardidates** package, which is available as source code and as Windows, Macintosh and Linux "binary". Currently we do not yet intend to upload **cardidates** to CRAN, the comprehensive R archive network, however, it is available from the R package development server R-Forge[1] of one of the authors to obtain the most recent version.

[1] `http://r-forge.r-project.org/projects/cardidates/`

- **cardidates** requires two contributed packages named **boot** and **pastecs** (Ripley 2007; Ibanez and Etienne 2006) which can be installed directly from the internet. This can be done either using the menu (e.g. on Windows) or with the following command:

```
> install.packages(c("boot", "pastecs"))
```

Following documentation and help examples of the package should be sufficient for applying the functions. It is not necessary to be a full-experienced R programmer. If you however want to crunch large data sets automatically a little bit of R knowledge is of course helpful.

3. Using the package

Identification of cardinal dates is carried out in the following way:

1. Read your data into R,
2. Identify the time window of the desired peak if not yet done before,
3. Fit a Weibull-type function with four or six parameters (we recommend the six parameteric as it is much more flexible),
4. Extract cardinal dates with functions `CDW` or `CDWa` or use the default `summary` method.
5. Results can be visualized with standard R functions or an object-specific `plot` method.

This process can be done either step by step for single peaks as described in section 3.1 or automatically for a series of peaks using the `metaCDW` function described in section 3.2. The package is licensed under the GNU Public License 2.0 or above (`http://www.gnu.org/copyleft/gpl.html`). It can be used free of charge within the AQUASHIFT community and by everyone else who is interested.

3.1. A step-by-step example

We start with a simple example of artificially generated data:

```
> x <- c(0, 19, 38, 57, 76, 95, 114, 133, 152, 170, 190, 208, 227,
+     246, 265, 284, 303, 322, 341, 360)
> y <- c(0.9, 1, 1.2, 1, 2, 4, 7, 3, 4, 2, 1.1, 1, 1, 1, 0.9, 1.2,
+     1, 0.8, 1.1, 1)
```

We can visualize these data with `plot`:

```
> plot(x, y)
```

The data set contains only one peak so the first step, the detection of the time window (section 4.2) can be omitted and we go directly to the most essential step of fitting a Weibull function[2]:

```
> library("cardidates")
> res <- fitweibull6(x, y)
```

The results are now stored in `res` which belongs to the `cardiFit` class. You may now inspect the results directly entering `res` at the command line but you can also go directly to the next step and enter:

[2]This step is sometimes a little bit crucial or time-consuming. Susanne did her best to provide good and robust heuristics but please don't give up if you encounter error messages. You may read section 4 and try to provide your own user-defined initial values or contact the package authors with your particular data set.

```
> summary(res)

=========== Summary of Cardinal Dates Algorithm 'Weibull' ===========
  number of fitted parameters for Weibull function: 6
  parameter values: 0.8645348 96.51244 8.265944 0.1407681 145.2652 4.003017
               r2 = 0.8920768
  quantile        = 0.05
  cardinal dates  (tMid, tBegin, tEnd): 110.2617 74.76914 180.4869
  original ymax   = 7
```

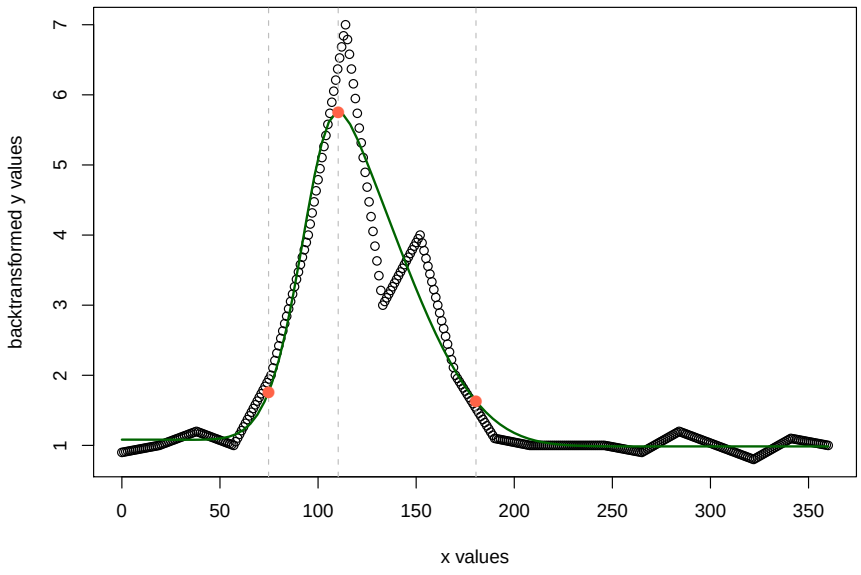
Function `fitweibull6` uses extensive heuristics to derive initial parameters for the optimization. It is intended to work with data which are defined over an interval between 0 and 365, e.g. environmental data and especially for plankton blooms. Please note that the function does internal an transformation:

$$y_{rel} = y_i / y_{max}$$

Additional data points are inserted to stabilize the algorithm between original measurements by linear interpolation with time step = 1 before curve fitting if the number of original data points is too low (currently $n < 35$). You can set `linint = 0` to switch interpolation off.

An automatic plotting method is also as simple as entering:

```
> plot(res)
```



showing cardinal dates based on the symmetric default method and the two-sided 5% quantile. That's it.

3.2. Fully automatic peak detection with metaCDW

Function `metaCDW` is a top-level function which calls `peakwindows`, `fitweibull` and `CDW` for a series of data sets and returns a table (a data frame) of all results.

To use it you must supply your data in a defined format, a table (technically an R data frame) with four columns: **sample** (description of the individual time series such as year, water body, or any other code, should be numeric or factor), **doy** (Julian day of the year), `y` (the dependent variable, e.g. phytoplankton biovolume) and `flag` (of type boolean indicating whether the data point should be included in analysis). The last column (flag) can be used to mask individual data points. It may contain either FALSE or 0 (zero) to disable respectively TRUE or 1 (one) to enable data points. If flag is omitted, all data sets are enabled. In the following we use an example data set provided by the package:

```
> data(carditest)
```

You may inspect this data set as usual, e.g. simply by typing `carditest` or, optionally, by plotting y versus time (see example on help page). Now we invoke the analysis with an important optional parameter `xstart` determining the offset of the main peak, e.g. a value of 55 to neglect winter peaks before the spring mass development. Setting the flag of the respective data manually to FALSE in the optional column provides another, more individual mechanism for this purpose:

```
> tt <- metaCDW(carditest, xstart = 55)

processing sample Year 1         converged, r2 = 0.9798
processing sample Year 2         converged, r2 = 0.9832
processing sample Year 3         converged, r2 = 0.9574
```

and inspect the results:

```
> summary(tt)

  sample     tMid   tBegin     tEnd     yMid   yBegin     yEnd        p1
1 Year 1 90.12881 69.56975 105.2697 16.95442 4.764888 4.452587 0.8400040
2 Year 2 84.98389 67.18631 101.2517 10.67672 4.196883 3.027391 0.7345372
3 Year 3 98.53680 60.79659 111.0254 25.76427 4.995066 9.087632 0.9359704
        p2        p3         p4        p5       p6        r2
1 82.94248 12.891681 0.04759207 102.29206 17.70486 0.9798084
2 78.98095 14.355165 0.08991999  97.40936 14.64137 0.9831828
3 88.96781  6.056224 0.12022559 108.83487 24.23530 0.9573789
```

The `summary` method shows the most important results of the analysis, i.e. the cardinal dates (`tMid` ... `yEnd`, the parameters of the Weibull function (`p1` ... `p6`) and the coefficient of determination (`r2`) of the fit. No additional steps are required if your data set is well-behaved, but as ever it's wise to check and visualize both, data and results. For plotting all data sets on one figure you may use:

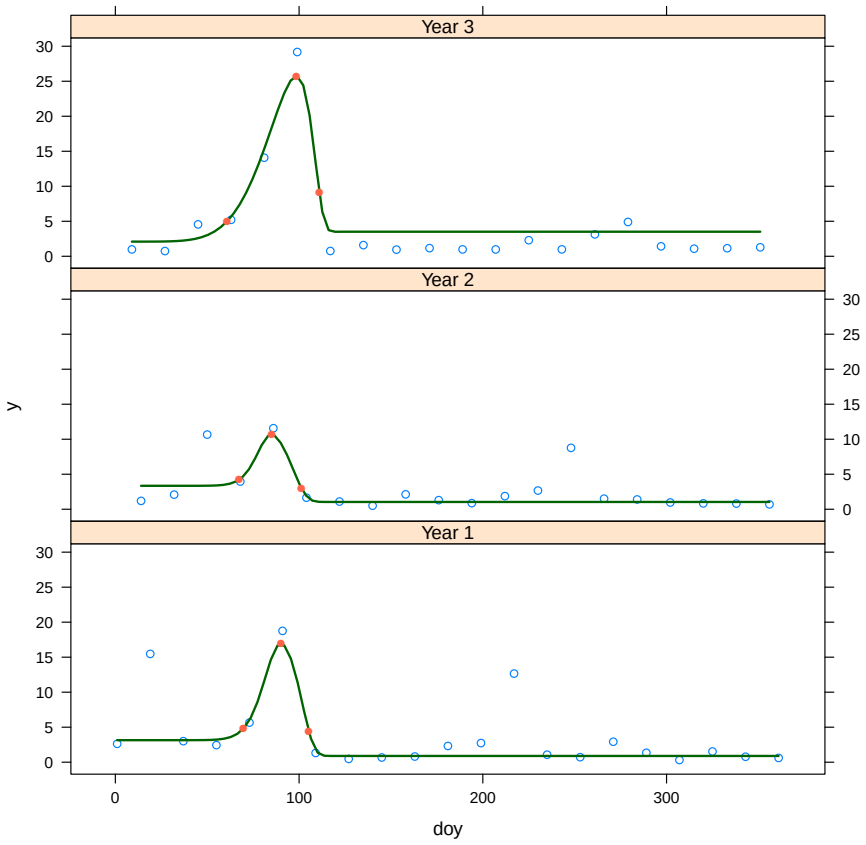
```
> plot(as.numeric(carditest$sample) * 365 + carditest$doy, carditest$y,
+     type = "b")
```

To plot the complete results of a (not too large) data set use:

```
plot(tt, carditest)
```

An optional `layout` argument can be supplied to control the layout of the matrix of figures, e.g. to one column and three rows:

```
plot(tt, carditest, layout=c(1, 3))
```



In addtion to this, you can also write graphics of all fitted peaks into one pdf file:

```
> pdf("d:/myfile.pdf")
> lapply(tt$weibullfits, plot)
> dev.off()
```

where `lapply` (list apply) is an R function that applies a function (in our case an object specific plotting method) to all list elements of `tt$weibullfits`.

The result of the analysis can be written to an arbitrary textfile supplied as optional file argument to `summary`, e.g.:

```
> summary(tt, file="d:/results.txt", sep="\t", quote=FALSE, row.names=FALSE)
```

or, to write the results directly to the windows clipboard in a format compatible to spreadsheets and with German comma:

```
> summary(tt, file="clipboard", sep="\t", dec=",", quote=FALSE, row.names=FALSE)
```

4. More options

4.1. How can I enter my own data?

Entering own data is one of the most flexible things in R that can read many data formats including databases and spreadsheets. The possibilities are described in detail in the R Data

Import/Export manual that comes with all versions of R and is also available online (http://cran.r-project.org/doc/manuals/R-data.html). Here is one of the most simple methods:

1. Copy your data as usual into your favorite spreadsheet program in two columns with title head **X** and **Y**, without empty lines above and without measurement units below the **X, Y** header or with three colums **sample**, **doy**, **y** in the `metaCDW` format.
2. Save this table as **text only** to your harddisk and remember where you have stored this file.
3. Now in R enter the lines below and your data should be in R as a table or so-called data frame:

```
> dat <- read.table(file.choose(), header=TRUE, sep="\t")
```

Please note that some languages use "," as decimal separator. In such cases use:

```
> dat <- read.table(file.choose(), header=TRUE, sep="\t", dec=",")
```

The data stored in `dat` can be passed directly to `fitweibull` if it has exactly two columns:

```
> fitweibull6(dat)
```

or to `metaCDW` if it has three or four columns named according to the above mentioned specification:

```
> metaCDW(dat)
```

Note that there are many other ways to enter, transform or manipulate your data, please see the online manuals for details.

4.2. Determining the time window of one peak

Sometimes it is not trivial to decide which data have to be included and which data are to be omitted before fitting the Weibull function, especially when we have several successive peaks. Instead of cutting the data manually the function `peakwindow`, which is based on an algorithm of Petzoldt (1996) can be used. This has not only the advantage of an automatic procedure, it is also more objective than the manual method and therefore better reproducible. If you use the `metaCDW` function then `peakwindow` is automatically invoked.

Let's have the following artificial data set generated by a randomly disturbed periodic function:

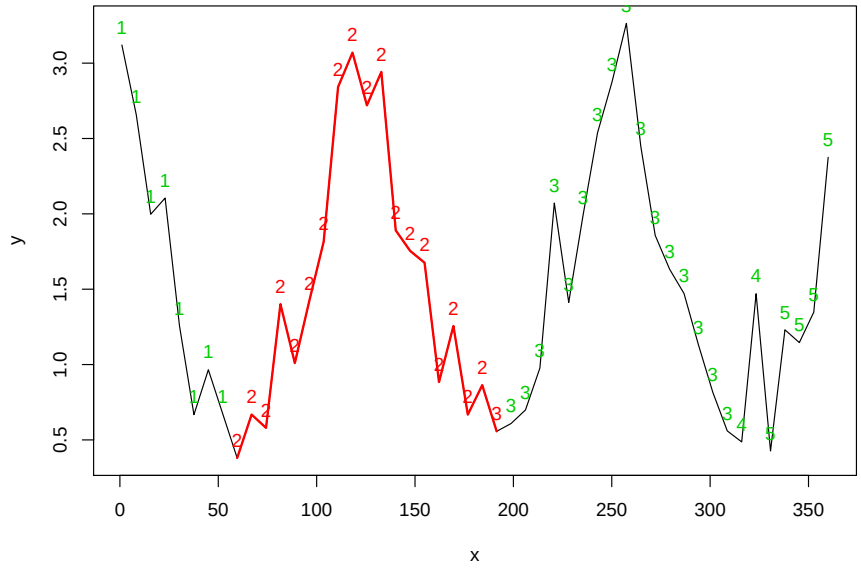
```
> set.seed(537)
> x <- seq(1, 360, length = 50)
> y <- exp(cos(x/20)) + 0.3 + rnorm(x, sd = 0.3)
```

We now apply the `peakwindow` function to the x and y data where `xstart = 70` is an optional value to indicate the time of ice out after which the phytoplankton spring mass development is expected to begin:

```
> peaks <- peakwindow(x, y, xstart = 60)
```

Again, an object specific plot method can be used to visualize the results. All identified are peaks labelled with consecutive numbers amd the spring is peak marked with a red line:

```
> plot(peaks)
```



The `peakwindow` function has some additional control parameters which, for example, determine how sub-peaks are handled. Among them `minpeak` is probably the most important. Its default (0.382) is derived from the Golden ratio, Petzoldt (1996) used a value of 1/3 and Rolinski *et al.* (2007) found a value of 0.455 to be optimal for the sum of diatoms of Saidenbach Reservoir, so you should play with this parameter if you think that the isolation of sub-peaks seems to be sub-optimal.

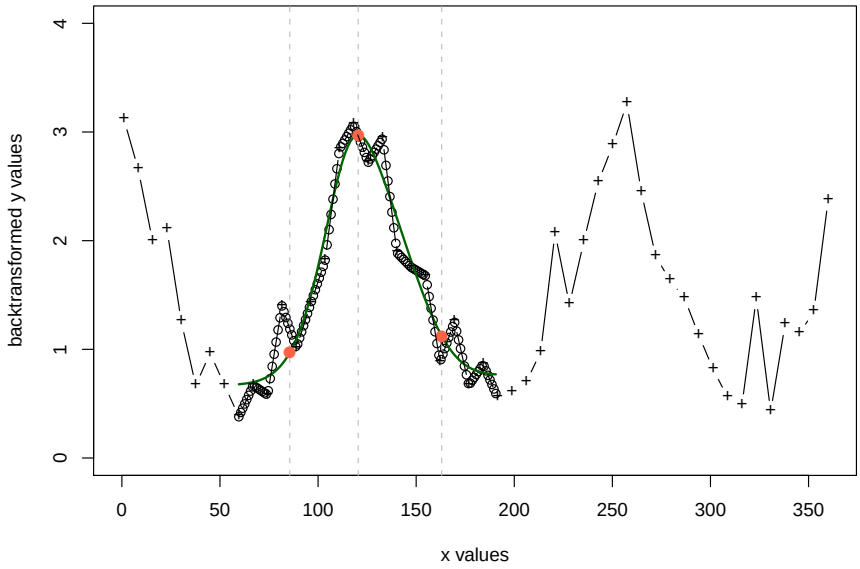
4.3. Weibull curve fitting

Fitting the Weibull function should be done using only relevant data, i.e. omitting unnecessary data before and after the peak either manually or using `peakwindow` which returns the indices of the relevant data as list element (`peaks$smd.indices`). We can store the contents of this element to another variable (`smd`) and then use this `smd`-subset to fit the Weibull function:

```
> smd <- peaks$smd.indices
> res <- fitweibull6(x[smd], y[smd])
> summary(res)
```

```
=========== Summary of Cardinal Dates Algorithm 'Weibull' ===========
  number of fitted parameters for Weibull function: 6
  parameter values: 0.8253797 108.0449 9.618016 0.2490877 146.0092 6.928396
               r2 = 0.9205135
  quantile        = 0.05
  cardinal dates  (tMid, tBegin, tEnd): 120.4515 85.58577 163.0151
  original ymax   = 3.069692
```

```
> plot(res, ylim = c(0, 4), xlim = c(0, 360))
> points(x, y, pch = "+", type = "b")
```



Cardinal date detection as described above is done with, as we think, useful defaults and heuristics. However, it is focussed on year courses (x between 0 to 365) and the characteristics of phytoplankton time series. The package contains several additional possibilities, e.g. the use of the four parametric Weibull function. Examples of typical shape for `fweibull4` and `fweibull6` are outlined on the help pages of these functions. Please note the graphical meaning of the function parameters $p_1 \ldots p_6$ for both functions. You can supply your own initial parameters in case of convergence problems and you can re-fit the curve with the measured data only avoiding the additional linear interpolation in case of small data sets.

4.4. Extraction of results

Functions `CDW` and `CDWa` can be used to extract cardinal dates. The meaning of the different versions and options is described in the help pages. Option `symmetric = FALSE` forces the use of an asymmetric algorithm as shown in Figure 2B in Rolinski *et al.* (2007):

```
> CDW(res, symmetric = FALSE)

$x
     tMid    tBegin      tEnd
120.45154  85.58577 163.01514

$y
[1] 0.9685136 0.3158143 0.3661096

$p
[1]   0.8253797 108.0449055   9.6180156   0.2490877 146.0092457   6.9283965
```

or with other quantiles:

```
> CDW(res, quantile = 0.01, symmetric = FALSE)

$x
     tMid    tBegin      tEnd
```

```
120.45154  73.53769 174.46130

$y
[1] 0.9685136 0.2416054 0.2813722

$p
[1]   0.8253797 108.0449055   9.6180156   0.2490877 146.0092457   6.9283965
```

You can also manually inspect, print or evaluate `res` or the return value of `CDW`:

```
> str(res)
> res
> cdw <- CDW(res)
> str(cdw)
> cdw
```

More examples are provided on the help pages.

5. Further information

General information about R including original and user-supplied packages and references to the most important books can be found on http://www.r-project.org. Please cite and **read** our paper when using this package (Rolinski *et al.* 2007). Feel free to contact us if you have any questions, problems or suggestions. We have several ideas in our own minds and it depends on user feedback if and when they will be included.

References

Ibanez F, Etienne PGM (2006). *pastecs: Package for Analysis of Space-Time Ecological Series*. R package version 1.3-4, URL http://www.sciviews.org/pastecs.

Petzoldt T (1996). *Möglichkeiten zur Vorhersage von Phytoplanktonmassenentwicklungen: Von der statischen Betrachtungsweise zur Kurzfristprognose*. Dissertation, TU Dresden, Fakultät Forst-, Geo- und Hydrowissenschaften.

R Development Core Team (2007). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Ripley B (2007). *boot: Bootstrap R (S-Plus) Functions (Canty)*. S original by Angelo Canty, R port by Brian Ripley.

Rolinski S, Horn H, Petzoldt T, Paul L (2007). "Identifying cardinal dates in phytoplankton time series to enable the analysis of long-term trends." *Oecologia*, **153**(4), 997 – 1008. doi: 10.1007/s00442-007-0783-2.