Technical Note
Fitting rational functions to time series in R

D G Rossiter
Department of Earth Systems Analysis
International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL)

10-December-2005

Contents

1 Introduction

These notes describe how to fit rational functions to a time series in the R environment for statistical computing and visualisation [2, 3] and its dialect of the S language.

For an explanation of the R project, including how to obtain and install the software and documentation, see Rossiter [4].

The problem context in which this note was developed and the results are described by Yemefack et al. [7] and in a chapter of Yemefack's dissertation [6]. The aim was to characterise the dynamic behaviour of soil properties in shifting cultivation systems in southern Cameroon. Soils were sampled synchronically from plots representing a chronosequence (from zero to more than 30 years) of these land use systems and also diachronically from the same plots over seven years. In these systems, previous studies had shown that the initial cutting and burning of vegetation leads to rapid changes in soil properties, followed by a gradual return to initial values ("relaxation") once plots are abandoned to fallow. If these changes can be modelled as a continuous function of time, this function could be used to compute three metrics describing soil behaviour over time: maximum proportional deviation from the base state, time to reach this maximum, and relaxation time towards the original value; these can then be interpreted in the context of the study.

Five soil properties (soil pH in water, exchangeable Ca, available P, organic C and bulk density) had been identified in a previous study as the most sensitive to these land use systems; these were to be modelled.

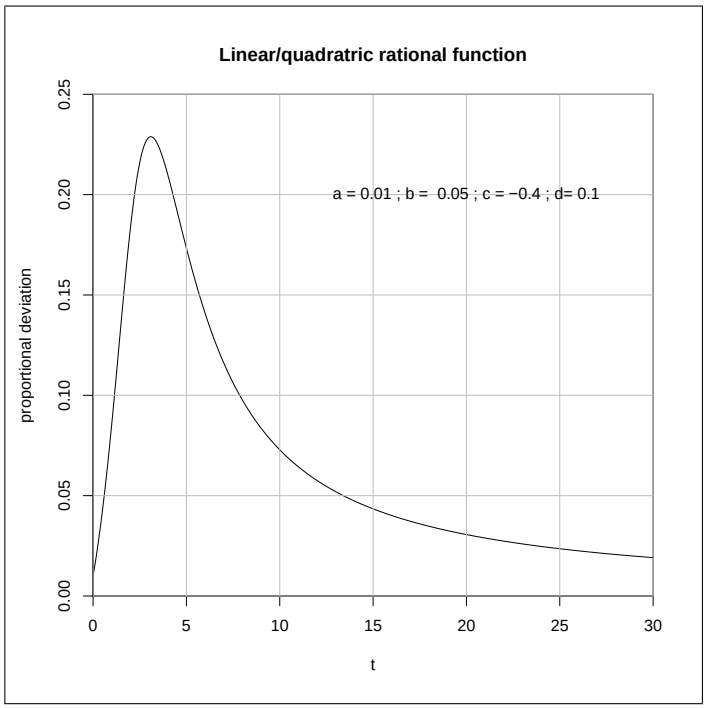
2 Rational functions

Rational functions [see for example 1] are ratios of any two polynomials in a single variable, here t because we are modelling a response over time. They are classified by the degrees of the numerator and denominator. We will restrict ourselves to the *linear/quadratic rational function*:

$$f(t) = \frac{a + bt}{1 + ct + dt^2} \tag{1}$$

since with suitable parameters ($a \approx 0,\ b > 0,\ c < 0,\ d > 0$) this shows a reasonable shape to model changing soil properties in

response to a single event: an initial S-shaped rise from near zero to a maximum, followed by an inverse-S-shaped decrease towards zero. Here is a typical curve with realistic parameters for this study:



In our study t is time since the beginning of the land use sequence. Parameter a is the intercept, i.e. function value at $t = 0$. This should be 0 by definition (see §2.2) but may be slightly different because of experimental error. Parameter b is a linear increase; parameter c is an inverse linear decrease; parameter d is an inverse quadratic decrease which dominates the equation at large values of t. In particular, if $d > 0$, $f(t) \to 0$ as $t \to \infty$, that is, the system approaches its original state.

2.1 Fitting a linear/quadratic rational function

Given a set of ordered pairs (t_i, y_i) where in general there are repeated measurements at each value of t, the parameters of a rational function can be fitted by *non-linear least-squares estimation*, for example with the `nls` method in R (§4). One we have the four parameters, we can compute the value of t at which this is maximized, by computing the first derivative:

$$f'(t) = \frac{b}{1+ct+dt^2} - \frac{(a+bt)(c+2dt)}{(1+ct+dt^2)^2} \tag{2}$$

and solving for t where $f'(t) = 0$. For reasonable parameter values the solutions will be real:

$$t_m = \frac{-ad \pm \sqrt{b^2d - abcd + a^2d^2}}{bd} \tag{3}$$

One of the t_m gives a maximum (where the second derivative is positive), the other a minimum (negative). In a typical case of positive deviation, the function is near zero at time zero, increases, and then decreases towards zero, so the fitted maximum will be positive, somewhere between $t = 0$ and the maximum time t_m. The t_m corresponding to this maximum can be substituted into Equation 1 to obtain the maximum function value y_m:

$$y_m = \frac{a + bt_m}{1 + ct_m + dt_m{}^2} \tag{4}$$

2.2 Forcing a zero deviation at time zero

In the study which motivated this note, the rational function was used to fit *proportional changes* from an initial condition in response to land use. In this case the intercept (the deviation at time zero) should be 0, since by definition there is no change at time zero. Then $a = 0$ and the function (Equation 1) is reduced to:

$$f(t) = \frac{bt}{1 + ct + dt^2} \tag{5}$$

The derivative (Equation 2) is reduced to:

$$f'(t) = \frac{b}{1 + ct + dt^2} - \frac{(bt)(c + 2dt)}{(1 + ct + dt^2)^2} \tag{6}$$

and finally the solution (Equation 3) is reduced to a very simple form:

$$t_m = \pm\frac{1}{\sqrt{d}} \tag{7}$$

For positive deviations the function maximum will be at the positive value, and will be:

$$y_m = \frac{bt_m}{1 + ct_m + dt_m{}^2} \tag{8}$$

which can be combined with Equation 7 to express the solution directly in terms of the parameters:

$$y_m = \frac{b/\sqrt{d}}{2 + c/\sqrt{d}} \tag{9}$$

2.3 Interpretation

The fitted coefficients of the rational equation can be used to derive two metrics with which to compare the behaviour of different soil properties:

- The value of t_{m}, which is the time to reach a maximum proportional deviation, i.e. how long the property takes to respond; and
- The value of y_{m}, which is the maximum proportional deviation, i.e. how much the property changes.

A third metric is the time t_p at which the curve reaches some predefined proportion of the maximum; this is termed the *relaxation time* to that proportion. To find this value, we first define a proportion $p \in [0..1]$ of the maximum we want to reach, for example $p = 0.2$ to reach one-fifth of the maximum. We label this proportion of y_{m} as $y_p = p \cdot y_{\mathrm{m}}$, which then becomes the left-hand side of Equation 1:

$$y_p = \frac{a + bt}{1 + ct + dt^2} \tag{10}$$

This is a quadratic with positive solution

$$t_p = \frac{1}{2d \cdot y_p}\left\{ b - c \cdot y_p + \sqrt{b^2 + 2(2ad - bc) \cdot y_p + (c^2 - 4d) \cdot {y_p}^2} \right\} \tag{11}$$

3 The example data set

The dataset has been prepared in the *comma-separated values* (CSV) file `MDS_PD_12.csv`. Here are the first few lines of this file:

```
Field_ID,SOIL,Blocks,LULC,SampSeq,Depth,Time,pHw,OC,Pav,Ca,Bd
Biyem-oca,A,4,FV,10,10,0,-0.040139616,-0.235294118,0.071428571,-0.46964018,0.103710752
Bokali-ff,A,4,FV,10,10,0,-0.040139616,0.441176471,0.428571429,-0.032983508,-0.096098953
...
```

The first line gives the field name for the data frame. The fields are:

Field_ID : Name of field (usually the farmer)

SOIL : WRB soil name: A = Acrisolsl, F = Ferralsols

Blocks : Study sub-area, a village; arbitrary codes 1 . . . 4.

LULC : Land use / Land cover code

SampSeq : Sequence of sampling, from 0 (just before field is cleared) to 6 (the sixth and last sampling date); this is an ordered factor. For the time-series analysis this has been converted to years in the `time` field, see below.

Depth : Sampling depth; code 10 is 0–10 cm ("topsoil"), code 20 is 20–50 cm ("subsoil").

Time : Years since start of sequence

pHw : Proportional deviation of soil pH in 1:5 water from its mean

OC : Proportional deviation of soil organic carbon

Pav : Proportional deviation of available phosphorous

Ca : Proportional deviation of exchangeable calcium

Bd : Proportional deviation of soil bulk density

Since this study was to model changes, not absolute values, each response variable was converted to a *proportional deviation* (PD) from the reference sites (primary forest, LULC code `FV`) as follows: If P_i is the value of a soil property from treatment i and P_0 the (non-zero) value of the same property from the corresponding FV on the same soil type, PD_i was computed as:

$$PD_i = \frac{P_i - P_0}{P_0} \tag{12}$$

Since sites are compared only with those on the same soil, variability due to soil type "should" be removed.

R code:

```
pd <- read.csv("MDS_PD_12.csv")
pd$Blocks <- as.factor(pd$Blocks)
pd$Depth <- as.ordered(pd$Depth)
pd$SampSeq <- as.ordered(pd$SampSeq)
str(pd); attach(pd)
unique(Time)
table(LULC, as.factor(Time))
```

R console output:

```
'data.frame': 347 obs. of  12 variables:
 $ Field_ID: Factor w/ 96 levels "Adjap-yca","Afana",..: 17 19 40 52 60 72 74 77 81 83 .
 $ SOIL    : Factor w/ 2 levels "A","F": 1 1 1 1 1 1 1 1 1 1 ...
 $ Blocks  : Factor w/ 4 levels "1","2","3","4": 4 4 4 4 4 4 4 4 4 4 ...
 $ LULC    : Factor w/ 12 levels "2CF","2CL1","2CL2",..: 10 10 10 10 10 10 10 10 10 10 .
 $ SampSeq : Ord.factor w/ 6 levels "0"<"1"<"2"<"3"<..: 6 6 6 6 6 6 6 6 6 6 ...
 $ Depth   : Ord.factor w/ 2 levels "10"<"20": 1 1 1 1 1 1 1 1 1 1 ...
 $ Time    : num  0 0 0 0 0 0 0 0 0 0 ...
 $ pHw     : num  -0.0401 -0.0401  0.1169  0.1169 -0.0576 ...
 $ OC      : num  -0.235  0.441 -0.206 -0.235  0.529 ...
 $ Pav     : num   0.0714  0.4286 -0.2857 -0.2857 -0.2857 ...
 $ Ca      : num  -0.4696 -0.0330  0.3362  0.3381  0.0139 ...
 $ Bd      : num   0.1037 -0.0961 -0.0676 -0.0771  0.0276 ...

[1]  0.0  0.3  1.5  3.0  7.0 12.0 20.0 30.0

LULC    0 0.3 1.5  3  7 12 20 30
  2CF   0   0   0  0  6  0  0  0
  2CL1  0   0   4  0  0  0  0  0
  2CL2  0   0   0  4  0  0  0  0
  BF    0   0   0  0  0 30  0  0
  CF    0   0   0  0 28  0  0  0
  CL1   0   0  58  0  0  0  0  0
  CL2   0   0   0 50  0  0  0  0
  FCF1  0  20   0  0  0  0  0  0
  FF    0   0   0  0  0  0 20  0
  FV   75   0   0  0  0  0  0  0
  MCA   0   0   0  0  0 26  0  0
  OCA   0   0   0  0  0  0  0 26
```

The table shows the correspondence between land cover class and time; this is a one-to-one relation. The longest-term plots are mature and old cocoa trees (codes `MCA` and `OCA`, respectively). The time of these is not certain, and they may have other dynamics than the shifting cultivation cropping sequence, so we make a data frame without the cocoa, and then further limit it to the topsoil, where the most change is expected:

R code:

```
pda <- pd[(pd$LULC!="OCA") & (pd$LULC!="MCA"),]
pda.d1 <- split(pda,Depth)[[1]]
```

We will fit functions for bulk density, so let's take a closer look at

it now.

R code:
```
histogram(~ Bd | as.factor(Time) * SOIL)
```

4 Implementation in R

The curve fitting, the subsequent computation of three metrics, and a graphical presentation can be implemented in R. The key is the `nls` method for non-linear regression; this is well-explained in Venables and Ripley [5, § 8.2].

First, the rational function:

R code:
```
rat12 <- function(t, a, b, c, d) {
 (a + b*t)/(1 + c*t + d*t^2)
}
```

and the version where $a = 0$ by definition:

R code:
```
  rat12a <-  function(t, b, c, d) {
  (b*t)/(1 + c*t + d*t^2)
}
```

These can be used to examine the shape of the curve, as in the plot above, which was produced by:

R code:
```
a <- 0.01; b <- 0.05; c <- -0.4; d <- 0.1;
plot(t, rat12(t, 0.01, 0.05, -0.4, 0.1), type="l",
     ylab="proportional deviation",
     xlim=c(0,30), ylim=c(0, 0.25), xaxs="i", yaxs="i",
     main="Linear/quadratric rational function")
grid(col="gray", lty=1)
text(20, .2, paste("a =", a, "; b = ", b, "; c =", c, "; d=", d))
```

Second, a function to fit the parameters the rational function; this also computes and prints the goodness-of-fit as expressed by the

AIC and r^2; it also plots the data points and fitted curve. Finally, it computes the relaxation time to a specified proportion, here by default $p = 0.2$. This function must be called with an attached dataframe attached (`attach()` method) which has a numeric `Time` field.

R code:

```
fit2 <- function (var, var.name="",
          n2.start=list(a=.01, b=0.05, c=-0.4, d=0.1)), p=0.2) {
                                        # x at maximum
  maxim <- function(coeff) {
    a <- coeff[1]; b <- coeff[2]; c <- coeff[3]; d <- coeff[4]
    det <- sqrt(b^2*d - a*b*c*d + a^2*d^2)
    -(a*d-det)/(b*d)
    }
                                        # find x where rat12(x) = y0
  solve12 <-function(y0, coeff) {
    a <- coeff[1]; b <- coeff[2]; c <- coeff[3]; d <- coeff[4]
    det <- (b^2 + (4*a*d - 2*b*c)*y0 + (c^2 - 4*d)*y0^2)
    (b - c*y0 +  sqrt(det)) / (2*d*y0)
  }
                                        # fit
  n2 <- nls(var ~ rat12(Time, a, b, c, d), start=n2.start, trace=F)
  print(coefficients(n2))
  print(paste("AIC: ", AIC(n2)))
                                        # goodness of fit
  r2 <- round(1-(sum(residuals(n2)^2) / sum((var-mean(var))^2)), 4)
  print(paste("R^2 =", r2))
                                        # maximum x and y
  maxim <- maxim(coefficients(n2))
  mpd <- rat12(maxim, coefficients(n2)[1], coefficients(n2)[2],
            coefficients(n2)[3], coefficients(n2)[4])
  relax <- solve12(mpd*p, coefficients(n2))
                                        # print diagnostics
  print(paste("x at maximum:", round(maxim, 2)))
  print(paste("maximum proportional deviation:", round(mpd, 3)))
  print(paste("relaxation time to p=", p, ": ", round(relax,2)))
                                        # plot it
  s <- seq(0, 20, by=.1)
  plot(s,rat12(s, coef(n2)[1], coef(n2)[2], coef(n2)[3], coef(n2)[4]),
       ylim=range(var),
       type="l", lwd=2.5, col="red", xlab="Years", ylab="Proportional deviation",
       main=paste("Linear/quadratic rational function,", var.name, "0-10 cm")
       )
  points(Time, var, pch=20, col=as.numeric(SOIL)+2);
  abline(h=0); abline(h=mpd, lty=2);
  abline(v=maxim, lty=2); abline(v=relax, lty=2);
  text(maxim+0.8, min(var), round(maxim, 2));
  text(relax+0.8, min(var), round(relax, 2));
  text(max(Time)-1, mpd, round(mpd, 3), pos=3)
}
```

The curve fitting may require some more explanation. The `nls` method requires two arguments: the model to fit and starting

parameters, for example:

R code:

```
n2.start=list(a=.01, b=0.05, c=-0.4, d=0.1))
n2 <- nls(var ~ rat12(Time, a, b, c, d), start=n2.start)
```

It is convenient to give the starting parameters as a *named list*.

This function may be called for various soil properties with an optional graph title, e.g.

R code:

```
fit2(pHw, "pH (water)")
```

The non-linear solver `nls` is an iterative procedure and so requires starting values. The data we are fitting have approximately the same shape, so the default set of starting values in the above function seems to work for most of them. There are pathological cases that have no real solution, because the input data does not follow even approximately the expected curve shape. In this case it's best not to try a fit.

The goodness-of-fit is a measure of how well the data follow an expected curve. If these values are too poor, it means the model is not appropriate, i.e. the soil property does not respond to land use as we expect.

This code can be simplified for the case where a is forced to 0:

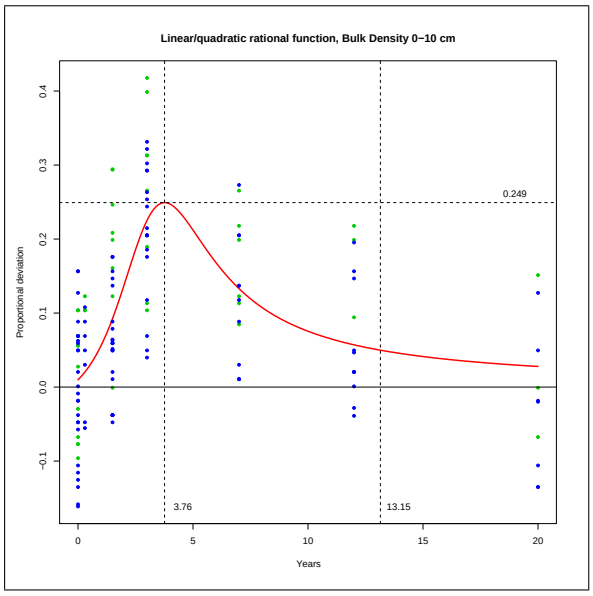
R code:

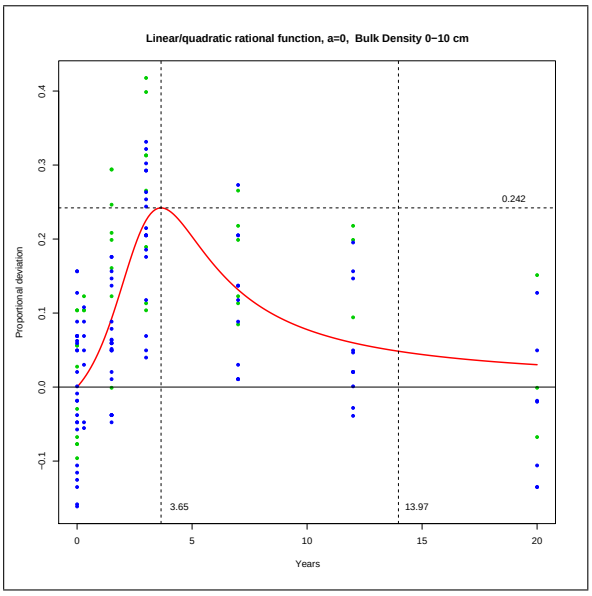
```
fit2a <- function (var, var.name="",
            n2a.start=list(b=0.05, c=-0.4, d=0.1), p=0.2) {
  solve12a <- function(y0, coeff) {
    b <- coeff[1]; c <- coeff[2]; d <- coeff[3]
    det <- ( (c*y0 - b)^2 - (4 * d*y0^2))
    ( -(c*y0 - b) + sqrt(det) ) / (2 * y0*d)
 }
                                        # fit
  n2a <- nls(var ~ rat12a(Time, b, c, d), start=n2a.start)
  print(coefficients(n2a))
  print(paste("AIC: ", AIC(n2a)))
                                        # goodness of fit
  r2 <- round(1-(sum(residuals(n2a)^2) / sum((var-mean(var))^2)), 4)
  print(paste("R^2 =", r2))
                                        # maximum x and y
  maxim <- 1/sqrt(coefficients(n2a)[3])
  mpd=rat12a(maxim,
    coefficients(n2a)[1], coefficients(n2a)[2], coefficients(n2a)[3])
  relax <- solve12a(mpd*p, coefficients(n2a))
  print(paste("x at maximum:", round(maxim, 2)))
  print(paste("maximum proportional deviation:", round(mpd, 3)))
  print(paste("relaxation time to p=", p, ": ", round(relax, 2)))
                                        # plot it
  s <- seq(0, 20, by=.1)
  plot(s,rat12a(s, coef(n2a)[1], coef(n2a)[2], coef(n2a)[3]),
       ylim=range(var), ylab="Proportional deviation",
       type="l", lwd=2.5, col="red", xlab="Years",
       main=paste("Linear/quadratic rational function, a=0,", var.name, "0-10 cm")
       )
  points(Time, var, pch=20, col=as.numeric(SOIL)+2)
  abline(h=0); abline(h=mpd, lty=2); abline(v=maxim, lty=2)
  text(maxim+0.8, min(var), round(maxim, 4))
  text(max(Time)-1, mpd, round(mpd, 4), pos=3)
  text(relax+0.8, min(var), round(relax, 2))
}
```

R code:

```
pda.d1 <- split(pda,Depth)[[1]]
> attach(pda.d1)
> fit2(Bd, "Bulk Density")
         a          b          c          d
 0.009728   0.027575 -0.400075   0.067853
[1] "AIC:  -280.385335081555"
[1] "R^2 = 0.4279"
[1] "x at maximum: 3.76"
[1] "maximum proportional deviation: 0.249"
[1] "relaxation time to p= 0.2 :  13.15"
> fit2a(Bd, "Bulk Density")
         b          c          d
 0.034622 -0.404815   0.075024
[1] "AIC:  -281.848188578333"
[1] "R^2 = 0.4258"
[1] "x at maximum: 3.65"
[1] "maximum proportional deviation: 0.242"
[1] "relaxation time to p= 0.2 :  13.97"
> detach(pda.d1)
```

In this case the restriction to $a = 0$ did not appreciably change the parameters or diagnostics. Here are the corresponding graphs:





5 Advanced R

In this section we explore some of the more esoteric aspects of non-linear solving.

Derivatives The non-linear solver can in principle be speeded up if it knows the *first derivative* of the function to be minimised. R has a `deriv` method which, given a function, produces another function to evaluate its derivative.

R code:

```
> (d <- deriv(y ~ (b*x)/(1 + c*x + d*x^2), c("b", "c", "d")))
expression(
    .expr1 <- b * x
    .expr4 <- x^2
    .expr6 <- 1 + c * x + d * .expr4
    .expr10 <- .expr6^2
    .value <- .expr1/.expr6
    .grad <- array(0, c(length(.value), 3), list(NULL, c("b",
        "c", "d")))
    .grad[, "b"] <- x/.expr6
    .grad[, "c"] <- -(.expr1 * x/.expr10)
    .grad[, "d"] <- -(.expr1 * .expr4/.expr10)
    attr(.value, "gradient") <- .grad
    .value
)
```

---

6 Results

Here are the metrics for four-paramter curves, 0-10 cm, combined soils, land uses not including perennial plantations:

Variable	t_m	y_m	$y_{0.2}$	r^2	AIC
pHw	2.50	0.232	10.48	0.4832	-283.1
Bd	3.76	0.249	13.15	0.4279	-280.4
Ca	0.97	3.637	14.56	0.4335	552.6
Pav	0.65	2.432	3.40	0.4498	294.4

The computation for OC does produce a fit, but it does not have the expected shape (it shows a great deal of spread and no clear trend) and no real roots. This variable can not be used as an indicator of land use change. The r^2 for the other propereties are fairly consistent, between 43–48%, indicating a large spread of the values at each time. There are clear differences between the proper-

ties: Ca and Pav have large, very quick reactions (within the first year and to at least 2.5 times the original value), whereas pHw and Bd have much less pronounced reactions (2.5 to 3.5 years and an increase of only about 0.25). Relaxation time for Pav is very quick (about 3.5 years); the others are on the order of 11 to 15 years.

7 Working with other subsets

The same procedure can be applied to the second layer:

R code:

```
> pda.d2 <- split(pda,Depth)[[2]]
> attach(pda.d2)
...
> detach(pda.d2)
```

---

and we can analyze each soil separately, in this example for the first layer. However, there are only 40 observations for the Acrisols; there are 110 for Ferralsols.

R code:

```
> pda.d1a <- split(pda.d1,SOIL)[[1]]
> attach(pda.d1a)
... # acrisols
> detach(pda.d1a)
> pda.d1f <- split(pda.d1,SOIL)[[2]]
> attach(pda.d1f)
... # ferralsols
> detach(pda.d1f)
```

References

[1] J W Harris and H Stocker. *Handbook of mathematics and computational science*. Springer-Verlag, New York, 1998.

[2] R Ihaka and R Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.

[3] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2004. URL http://www.R-project.org. ISBN 3-900051-07-0.

[4] D G Rossiter. *Introduction to the R Project for Statistical Computing for use at ITC*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 2005.

[5] W N Venables and B D Ripley. *Modern applied statistics with S*. Springer-Verlag, New York, 4th edition, 2002. URL http://www.stats.ox.ac.uk/pub/MASS4/.

[6] M Yemefack. *Modelling and monitoring soil and land use dynamics within shifting agricultural mosaic systems*. ITC Dissertation 121. ITC Enschede and Utrecht University, Enschede and Utrecht, the Netherlands, 2005.

[7] M Yemefack, D G Rossiter, and V G Jetten. Empirical modelling of soil dynamics along a chronosequence of shifting cultivation systems in southern Cameroon. *Geoderma*, In Press, corrected proof available online 22 September 2005.