Rfit: Rank-based Estimation for Linear Models

by John D. Kloke and Joseph W. McKea

Abstract In the nineteen seventies, Jurečková and Jaeckel proposed rank estimation for linear models. Since that time, several authors have developed inference and diagnostic methods for these estimators. These rank-based estimators and their associated inference are highly efficient and are robust to outliers in response space. The methods include estimation of standard errors, tests of general linear hypotheses, confidence intervals, diagnostic procedures including studentized residuals, and measures of influential cases. We have developed an R package, Rfit, for computing of these robust procedures. In this paper we highlight the main features of the package. The package uses standard linear model syntax and includes many of the main inference and diagnostic functions.

Introduction

Rank-based estimators were developed as a robust, nonparametric alternative to traditional likelihood or least squares estimators. Rank-based regression was first introduced by Jurečková (1971) and Jaeckel (1972). McKean and Hettmansperger (1978) developed a Newton step algorithm that led to feasible computation of these rank-based estimates. Since then a complete rank-based inference for linear models has been developed that is based on rank-based estimation analogous to the way that traditional analysis is based on least squares (LS) estimation; see Chapters 3-5 of the monograph by Hettmansperger and McKea (2011) and Chapter 9 of Hollander and Wolfe (1999). Furthermore, robust diagnostic procedures have been developed with which to ascertain quality of fit and to locate outliers in the data; see McKean and Sheather (2009) for a recent discussion. Kloke et al. (2009) extended this rank-based inference to mixed models. Thus rank-based analysis is a complete analysis analogous to the traditional LS analysis for general linear models. This rank-based analysis generalizes Wilcoxon procedures for simple location models and, further, it inherits the same high efficiency that these simple nonparametric procedures possess. In addition, weighted versions can have high breakdown (up to 50%) in factor space (Chang et al., 1999). In this paper, we discuss the Rfit package that we have developed for rank-based (R) estimation and inference for linear models. We illustrate its use on examples from simple regression to k-way factorial designs.

The geometry of rank-based estimation is similar to that of LS. In rank-based regression, however, we replace Euclidean distance with another measure of distance which we refer to as Jaeckel’s dispersion function; see Hettmansperger and McKea (2011) for details. For a brief overview see McKea (2004).

Jaeckel’s dispersion function depends on the choice of a score function. As discussed in Hettmansperger and McKea (2011), the rank-based fit and associated analysis can be optimized by a prudent choice of scores. If the form of the error distribution is known and the associated scores are used, then the analysis is fully efficient. In Rfit we have included a library of score functions. The default option is to use Wilcoxon (linear) scores, however it is straightforward to create user-defined score functions. We discuss score functions further in a later section.

Others have developed software for rank-based estimation. Kapenga et al. (1995) developed a Fortran package and Crimin et al. (2008) developed a web interface (cgi) with Perl for this Fortran program. Terpstra and McKea (2005) developed a set of R functions to compute weighted Wilcoxon (WW) estimates including the high breakdown point rank-based (HBR) estimate proposed by Chang et al. (1999). See McKea et al. (2009) for a recent review. Rfit differs from the WW estimates in that its estimation algorithms are available for general scores and it uses a standard linear models interface.

The package Rfit allows the user to implement rank-based estimation and inference described in Chapters 3-5 of Hettmansperger and McKea (2011) and Chapter 9 of Hollander and Wolfe (1999). There are other robust packages in R. For example, the R function rlm of the R package MASS (Venables and Ripley, 2002) computes M estimates for linear models based on the $\psi$ functions of Huber, Hampel, and Tukey (bisquare). The CRAN Task View on robust statistical methods offers robust procedures for linear and nonlinear models including methods based on M, M-S, and MM estimators. These procedures, though, do not obtain rank-based estimates and associated inference as do the procedures in Rfit.

Rfit uses standard linear model syntax so that those familiar with traditional parametric analysis can easily begin running robust analyses. In this paper, discussion of the assumptions are kept to a minimum and we refer the interested reader to the literature. All data sets used in demonstrating Rfit are included in the package.

The rest of the paper is organized as follows. The next section discusses the general linear model and rank-based fitting and associated inference. The
following section provides examples illustrating the computation of Rfit for linear regression. Later sections discuss Rfit’s computation of one-way and multi-way ANOVA as well as general scores and writing user-defined score functions for computation in Rfit. The final section discusses the future implementations in Rfit of rank-based procedures for models beyond the general linear model.

### Rank-regression

As with least squares, the goal of rank-based regression is to estimate the vector of coefficients, $\beta$, of a general linear model of the form:

$$ y_i = \alpha + x_i^T \beta + e_i \quad \text{for } i = 1, \ldots n $$

(1)

where $y_i$ is the response variable, $x_i$ is the vector of explanatory variables, $\alpha$ is the intercept parameter, and $e_i$ is the error term. We assume that the errors are iid with probability density function (pdf) $f(t)$. For convenience, we write (1) in matrix notation as follows

$$ y = \alpha 1 + X \beta + e $$

(2)

where $y = [y_1, \ldots, y_n]^T$ is the $n \times 1$ vector of responses, $X = [x_1, \ldots, x_n]^T$ is the $n \times p$ design matrix, and $e = [e_1, \ldots, e_n]^T$ is the $n \times 1$ vector of error terms. The only assumption on the distribution of the errors is that it is continuous; in that sense the model is general. Recall that the least squares estimator is the minimizer of Euclidean distance between $y$ and $\hat{y}_{LS} = X\hat{\beta}_{LS}$. To obtain the R estimator, a different measure of distance is used that is based on the dispersion function of Jaeckel (1972). Jaeckel’s dispersion function is given by

$$ D(\beta) = \| y - X \beta \|_\varphi $$

(3)

where $\| \cdot \|_\varphi$ is a pseudo-norm defined as

$$ \| u \|_\varphi = \sum_{i=1}^{n} a(R(u_i)) u_i, $$

where $R$ denotes rank, $a(t) = \varphi \left( \frac{t}{\varphi(t)} \right)$, and $\varphi$ is a non-decreasing, square-integrable score function defined on the interval $(0,1)$. Assume without loss of generality that it is standardized, so that $\int_0^1 \varphi(u) du = 0$ and $\int \varphi^2(u) du = 1$. Score functions are discussed further in a later section.

The R estimator of $\beta$ is defined as

$$ \hat{\beta}_\varphi = \text{Argmin} \| y - X \beta \|_\varphi. $$

(4)

This estimator is a highly efficient estimator which is robust in the Y-space. A weighted version can attain 50% breakdown in the X-space at the expense of a loss in efficiency (Chang et al., 1999).

### Inference

Under assumptions outlined in the previous section, it can be shown that the solution to (4) is consistent and asymptotically normal (Hettmansperger and McKean, 2011). We summarize this result as follows:

$$ \hat{\beta}_\varphi \sim N \left( \beta, \tau_{\varphi}^2 (X^T X)^{-1} \right) $$

where $\tau_{\varphi}$ is a scale parameter which depends on $\varphi$ and the score function $\varphi$. An estimate of $\tau_{\varphi}$ is necessary to conduct inference and Rfit implements the consistent estimator proposed by Koul et al. (1987). Denote this estimator by $\hat{\tau}_{\varphi}$. Then Wald tests and confidence regions/intervals can be calculated. Let $se(\hat{\beta}_j) = \hat{\tau}_{\varphi} (X^T X)^{-1} j$ where $(X^T X)^{-1} j$ is the jth diagonal element of $(X^T X)^{-1}$. Then an approximate $(1 - \alpha) \times 100\%$ confidence interval for $\beta_j$ is

$$ \hat{\beta}_j \pm t_{1-\alpha/2,n-p-1} se(\hat{\beta}_j). $$

A Wald test of the general linear hypothesis

$$ H_0 : M\beta = 0 \quad \text{versus} \quad H_A : M\beta \neq 0 $$

is to reject $H_0$ if

$$ \frac{(M\hat{\beta}_\varphi)^T [M (X^T X)^{-1} M]^T [M (X^T X)^{-1} M]\hat{\beta}_\varphi / q}{\hat{\tau}_{\varphi}^2} > F_{1-a,n-p-1} $$

where $q = \text{dim}(M)$. Similar to the reduced model test of classical regression rank-based regression offers a drop in dispersion test which is implemented in the R function drop.test. For the above hypotheses let $\hat{\theta}_\varphi$ be the rank-based coefficient estimate of the reduced model [Model (1) constrained by $H_0$]. As discussed in Theorem 3.7.2 of Hettmansperger and McKean (2011), the reduced model design matrix is easily obtained using a QR-decomposition on $M^T$. We have implemented this methodology in Rfit. Similar to the LS reduction in sums of squares, the rank-based test is based on a reduction of dispersion from the reduced to the full model. Let $D(\hat{\theta}_\varphi)$ and $D(\hat{\beta}_\varphi)$ denote the reduced and full model minimum dispersions, then the test is to reject $H_0$ if

$$ \frac{[D(\hat{\theta}_\varphi) - D(\hat{\beta}_\varphi)] / q}{\hat{\tau}_{\varphi}^2 / 2} > F_{1-a,n-p-1} $$

### Computation

It can be shown that Jaeckel’s dispersion function (3) is a convex function of $\beta$ (Hettmansperger and McKean, 2011). Rfit uses optim with option ‘BFGS’ to minimize the dispersion function. We investigated other minimization methods (e.g. Nelder-Mead or CG), however the quasi-Newton method works well in terms of speed and convergence. An orthonormal basis matrix, for the space spanned by the columns
of \( X \), is first calculated using \( qr \) which leads to better performance in examples. The default initial fit is based on an \( L_1 \) fit using \texttt{quantreg} (Koenker, 2011).

Computations by \texttt{Rfit} of rank-based estimation and associated inference are illustrated in the examples of the next section.

**Rfit computations of rank-based fitting of linear models**

For the general linear model (1) the package \texttt{Rfit} obtains the rank-based estimates and inference, as described in the previous section. In this section we illustrate this computation for two examples. The first is for a simple linear model while the second is for a multiple regression model.

**Example 1** (Telephone data). We begin with a simple linear regression example using the telephone data discussed in Rousseuw and Leroy (1987) These data represent the number of telephone calls (in tens of millions) placed in Belgium over the years 1950–1973. The data are plotted in Figure 1. There are several noticeable outliers which are due to a mistake in the recording units for the years 1964–1969. This is a simple dataset, containing only one explanatory variable, however it allows us to easily highlight the package and also demonstrate the robustness to outliers of the procedure. The main function of the package \texttt{Rfit} is \texttt{rfit} which, as the following code segment illustrates, uses syntax similar to \texttt{lm}.

```r
> library(Rfit)
> data(telephone)
> fit <- rfit(calls ~ year, data = telephone)
> summary(fit)
```

Call:
\texttt{rfit(formula = calls ~ year, data = telephone)}

Coefficients:

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Error</th>
<th>t.value</th>
<th>p.value</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>0.145861</td>
<td>0.077842</td>
<td>1.8738</td>
<td>0.07494</td>
</tr>
</tbody>
</table>

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Multiple R-squared (Robust): 0.3543158
Reduction in Dispersion Test: 12.07238 p-value: 0.00215

```r
> plot(telephone)
> abline(fit)
> abline(lm(calls ~ year, data = telephone),
>        col = 2, lty = 2)
> legend("topleft", legend = c("R", "LS"),
>        col = 1:2, lty = 1:2)
```

Further, the output is similar to that of \texttt{lm} and can be interpreted in the same way. The estimate of slope is 0.146 (tens of millions of calls per year) with a standard error of 0.078. The \( t \)-statistic is the ratio of the two and the \( p \)-value is calculated using a \( t \)-distribution with \( n - 2 \) degrees of freedom. Hence one could conclude that year is a marginally significant predictor of the number of telephone calls.

The overlaid fitted regression lines in the scatter plot in Figure 1 demonstrate the robustness of the Wilcoxon fit and the lack of robustness of the least squares fit.

**Example 2** (Free fatty acid data). This is a data set from Morrison (1983, p. 64) (c.f. Example 3.9.4 of Hettmansperger and McKean (2011)). The response variable is level of free fatty acid in a sample of prepubescent boys. The explanatory variables are age (in months), weight (in lbs), and skin fold thickness. For this discussion, we chose the Wilcoxon (default) scores for \texttt{Rfit}. Based on the residual and Q-Q plots below, however, the underlying error distribution appears to be right-skewed. In a later section we analyze this data set using more appropriate (bent) scores for a right-skewed distribution.

To begin with we demonstrate the reduction in dispersion test discussed in the previous section.

```r
> fitF <- rfit(ffa ~ age + weight + skin,
>              data = ffa)
> fitR <- rfit(ffa ~ skin, data = ffa)
> drop.test(fitF, fitR)
```

Drop in Dispersion Test
\texttt{F-Statistic} p-value
1.0754e+01 2.0811e-04

Figure 1: Scatter plot of the telephone data with overlaid regression lines.
As the code segment shows, the syntax is similar to that of the `anova` function used for reduced model testing in many of the parametric packages.

![](image1)

**Figure 2:** Studentized residuals versus fitted values for the free fatty acid data.

Studentized residuals for rank-based fits are calculated in a way similar to the LS studentized residuals (see Chapter 3 of Hettmansperger and McKean, 2011). We have implemented these residuals in `Rfit` and demonstrate their use next. These are the residuals plotted in the residual and Q-Q plots in Figure 2 and Figure 3 respectively. The code is similar to that of least squares analysis. The function `fitted.values` returns the fitted values and `residuals` returns the residuals from the full model fit. The function `rstudent` calculates the studentized residuals.

Common diagnostic tools are the residual plot (Figure 2)

```r
> plot(fitted.values(fitF), rstudent(fitF))
> abline(h = c(-2, 2))
```

and normal probability plot of the studentized residuals (Figure 3).

```r
> qqnorm(residuals(fitF))
```

As is shown in the plots, there are several outliers and perhaps the errors are from a right skewed distribution. We revisit this example in a later section.

### One-way ANOVA

Suppose we want to determine the effect that a single factor \( A \) has on a response of interest over a specified population. Assume that \( A \) consists of \( k \) levels or treatments. In a completely randomized design (CRD), \( n \) subjects are randomly selected from the reference population and \( n_i \) of them are randomly assigned to level \( i \), \( i = 1, \ldots, k \). Let the \( i \)th response in the \( i \)th level be denoted by \( Y_{ij}, j = 1, \ldots, n_i \). We assume that the responses are independent of one another and that the distributions among levels differ by at most shifts in location.

Under these assumptions, the full model can be written as

\[
Y_{ij} = \mu_i + \epsilon_{ij}, \quad j = 1, \ldots, n_i, \quad i = 1, \ldots, k, \tag{5}
\]

where the \( \epsilon_{ij} \)'s are iid random variables with density \( f(x) \) and the parameter \( \mu_i \) is a convenient location parameter for the \( i \)th level, (for example, the mean or median of the \( i \)th level). Generally, the parameters of interest are the effects (contrasts), \( \Delta_{i'} = \mu_{i'} - \mu_i, \quad i \neq i', 1, \ldots, k \). Upon fitting the model a residual analysis should be conducted to check these model assumptions.

Observational studies can also be modeled this way. Suppose \( k \) independent samples are drawn from \( k \) different populations. If we assume further that the distributions for the different populations differ by at most a shift in location then Model (5) is appropriate.

The analysis for this design is usually a test of the hypothesis that all the effects are 0, followed by individual comparisons of levels. The hypothesis can be written as

\[
H_0: \mu_1 = \cdots = \mu_k \quad \text{versus} \quad H_A: \mu_{i'} \neq \mu_i \quad \text{for some } i \neq i'. \tag{6}
\]

Confidence intervals for the simple contrasts \( \Delta_{i'} \) are generally used to handle the comparisons. `Rfit` offers
a reduction in dispersion test for testing (6) as well as pairwise p-values adjusted for multiple testing. The function \texttt{oneway.rfit} is illustrated in the following example.

**Example 3 (LDL cholesterol of quail).**

Hettmansperger and McKean (2011, p. 295) discuss a study that investigated the effect of four drug compounds on low density lipid (LDL) cholesterol in quail. The drug compounds are labeled as I, II, III, and IV. The sample size for each of the first three levels is 10 while 9 quail received compound IV. The boxplots shown in Figure 4 attest to a difference in the LDL levels.

```r
> data(quail)
> oneway.rfit(quail$ldl, quail$treat)
```

\> Call:
\> \texttt{oneway.rfit(y = quail$ldl, g = quail$treat)}

\> Overall Test of All Locations Equal

\> Drop in Dispersion Test
\> P-Statistic \hspace{1em} p-value
\> 3.916404 \hspace{1em} 0.016403

\> Pairwise comparisons using Rfit
data: \texttt{quail$ldl} and \texttt{quail$treat}
\> 2 3 4
\> 1 \hspace{1em} - \hspace{1em} -
\> 2 \hspace{1em} 1.00 \hspace{1em} -
```

With a \(p\)-value of 0.0164, generally, the null hypothesis would be rejected and the inference would pass to the comparisons of interest. Finally, we note that, the LS test of the null hypothesis has \(p\)-value 0.35; hence, with the LS analysis \(H_0\) would not be rejected. In practice, one would not proceed with comparisons with such a large \(p\)-value. Thus, for this data set the robust and LS analyses have different interpretations.

**Multiple comparisons**

The second stage of an analysis of a one-way design usually consists of pairwise comparisons of the treatments. The robust \((1 - \alpha)100\%\) confidence interval to compare the \(ith\) and \(i'\)th treatments is given by

\[
\hat{\Delta}_{ii'} \pm t_{\alpha/2,n-1} \hat{\tau} \sqrt{\frac{1}{n_i} + \frac{1}{n_i'}}
\]

\((7)\)

Often there are many comparisons of interest. For example, in the case of all pairwise comparisons there
are \( \binom{k}{2} \) confidence intervals. Hence, the overall family error rate is usually of concern. Multiple comparison procedures (MCP) try to control the overall error rate to some degree. There are many MCPs from which to choose; see Chapter 4 of Hettmansperger and McKean (2011) for a review of many of these procedures from a robust perspective. In Rfit we supply a summary function that adjusts confidence intervals and use three of the most popular such procedures: protected least significant difference (none); Tukey-Kramer (tukey); and the Bonferroni (bonferroni). These methods are described in many standard statistics texts.

**Example 4** (LDL cholesterol of quail, continued). For the quail data, we selected the Tukey-Kramer procedure for all six pairwise comparisons. Use of the code and example output is given below. The multiple comparison part of the output is:

```r
> summary(oneway.rfit(quail$ldl, quail$treat), +    method = "tukey")
```

Multiple Comparisons

Method Used  tukey

<table>
<thead>
<tr>
<th></th>
<th>I J</th>
<th>Estimate</th>
<th>St Err</th>
<th>Lower CI</th>
<th>Upper CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1</td>
<td>1 2</td>
<td>-25.0072</td>
<td>8.2681</td>
<td>-47.3055</td>
<td>-2.7086</td>
</tr>
<tr>
<td>1 3</td>
<td>3 1</td>
<td>-3.9998</td>
<td>8.2681</td>
<td>-13.2982</td>
<td>5.2982</td>
</tr>
<tr>
<td>2 1</td>
<td>2 3</td>
<td>-20.0073</td>
<td>8.2681</td>
<td>-42.3055</td>
<td>1.2909</td>
</tr>
<tr>
<td>2 3</td>
<td>3 1</td>
<td>-1.0004</td>
<td>8.4947</td>
<td>-9.9098</td>
<td>7.9098</td>
</tr>
<tr>
<td>2 4</td>
<td>4 2</td>
<td>-1.0004</td>
<td>8.4947</td>
<td>-9.9098</td>
<td>7.9098</td>
</tr>
<tr>
<td>3 1</td>
<td>1 3</td>
<td>-3.9998</td>
<td>8.2681</td>
<td>-13.2982</td>
<td>5.2982</td>
</tr>
<tr>
<td>3 4</td>
<td>4 3</td>
<td>-1.0004</td>
<td>8.4947</td>
<td>-9.9098</td>
<td>7.9098</td>
</tr>
</tbody>
</table>

The Tukey-Kramer procedure declares that the Drug Compounds I and II differ significantly.

**Multi-way ANOVA**

In this section, we consider a \( k \)-way crossed factorial experimental design. For these designs, the Rfit function raov computes the rank-based analysis for all \( 2^k - 1 \) hypotheses including the main effects and interactions of all orders. The design may be balanced or unbalanced. For simplicity, we briefly discuss the analysis in terms of a cell mean (median) model; see Hocking (1985) for details on the traditional LS analysis and Chapter 4 of Hettmansperger and McKean (2011) for the rank-based analysis. For this paper, we illustrate Rfit using a two-way crossed factorial design, but similarly Rfit computes the rank-based analysis of a \( k \)-way design.

Let \( A \) and \( B \) denote the two factors with levels \( a \) and \( b \), respectively. Let \( Y_{ijk} \) denote the response for the \( k \)th replication at levels \( i \) and \( j \) of factors \( A \) and \( B \), respectively. Then the full model can be expressed as

\[
Y_{ijk} = \mu_{ij} + \epsilon_{ijk} \quad k = 1 \ldots n_{ij} \\
\quad i = 1 \ldots a \\
\quad j = 1 \ldots b,
\]

where \( \epsilon_{ijk} \) are iid random variables with pdf \( f(t) \). Since the effects of interest are contrasts in the \( \mu_{ij} \)'s, these parameters can be either cell means or medians, (actually any location functional suffices). Rfit implements a reduction in dispersion tests for testing all main effects and interactions.

For the two-way model, the three hypotheses of immediate interest are the main effects hypotheses and the interaction hypothesis. We have chosen Type III hypotheses which are easy to interpret even for severely unbalanced designs. Following Hocking (1985), the hypothesis matrices \( M \) can easily be computed in terms of Kronecker products. As discussed in a previous section, for these tests the drop in dispersion test statistics can easily be constructed. We have implemented this formulation in Rfit.

**Example 5** (Box-Cox data). Consider the data set discussed by Box and Cox (1964). The data are the results of a \( 3 \times 4 \) two-way design, where forty-eight animals were exposed to three different poisons and four different treatments. The design is balanced with four replications per cell. The response was the log survival time of the animal. An interaction plot using the cell medians is presented in Figure 6. Obviously the profiles are not parallel and interaction is present.

The output below displays the Wilcoxon ANOVA table, which indicates that interaction is highly significant, \( p = 0.0143 \), confirming the profile plot. On the other hand, the LS test \( F \) statistic for interaction is 1.87 with \( p = 0.1123 \). Hence, the LS test fails to detect interaction.

```r
> data(BoxCox)
> attach(BoxCox)
> fit <- raov(logSurv ~ Treatment + Poison)
> fit
```
Robust ANOVA Table

<table>
<thead>
<tr>
<th></th>
<th>DF</th>
<th>RD</th>
<th>Mean RD</th>
<th>F</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>3</td>
<td>2.9814770</td>
<td>0.9938257</td>
<td>21.263421</td>
<td>4.246022e-08</td>
</tr>
<tr>
<td>P</td>
<td>2</td>
<td>3.6987828</td>
<td>1.8493914</td>
<td>39.568699</td>
<td>8.157360e-10</td>
</tr>
<tr>
<td>T:P</td>
<td>6</td>
<td>0.8773742</td>
<td>0.1462290</td>
<td>3.128647</td>
<td>1.428425e-02</td>
</tr>
</tbody>
</table>

Writing score functions for Rfit

As discussed earlier, we must choose a score function for rank-based fitting. For most datasets the default option of Wilcoxon scores works quite well, however, occasionally choosing a different score function can lead to a more efficient analysis. In this section we first discuss score functions in general and then illustrate how the user may create his own score function. We have placed the score functions in an object of class "scores". A "scores" object consists of two objects of type function and an optional numeric object. The functions are the score function phi and it's derivative Dphi. The derivative is necessary in estimating \( \tau \phi \). Below is what the class for Wilcoxon scores looks like.

```r
> wscores
An object of class "scores"
Slot "phi":
function(u) sqrt(12)*(u-0.5)
Slot "Dphi":
function(u) rep(sqrt(12),length(u))
Slot "param": NULL
```

Other score functions included in Rfit are listed in Table 1. A plot of the bent score functions is provided in Figure 7. Other score functions can be plotted by getting the scores using the method getScores. For example the commands `u<-seq(0.01,0.99,by=0.01)` `plot(u,getScores(nscores,u))` graphs the normal scores.

<table>
<thead>
<tr>
<th>Score</th>
<th>Keyword</th>
<th>Recommended usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wilcoxon</td>
<td>wscores</td>
<td>moderate tailed</td>
</tr>
<tr>
<td>Normal</td>
<td>nscores</td>
<td>light-moderate tailed</td>
</tr>
<tr>
<td>Bent1</td>
<td>bentscores1</td>
<td>highly right skewed</td>
</tr>
<tr>
<td>Bent2</td>
<td>bentscores2</td>
<td>light skewed</td>
</tr>
<tr>
<td>Bent3</td>
<td>bentscores3</td>
<td>highly left skewed</td>
</tr>
<tr>
<td>Bent4</td>
<td>bentscores4</td>
<td>moderately heavy tailed</td>
</tr>
</tbody>
</table>

Table 1: Table of available score functions. Unless otherwise noted, distribution is assumed to be symmetric.

Next we illustrate how to create the score function for the bent scores. Bent scores are recommended when the errors come from a skewed distribution.

An appropriate bent score function for skewed distribution with a right heavy tail is

\[
\phi(u) = \begin{cases} 
4u - 1.5 & \text{if } u \leq 0.5 \\
0.5 & \text{if } u > 0.5 
\end{cases}
\]

The following code segment defines the scores.

```r
> bent.phi <- function(u, ...)
+ ifelse(u < 0.5, 8/3 * u - 1, 1/3)
> bent.Dphi <- function(u, ...)
+ ifelse(u < 0.5, 8/3, 0)
> bentscores <- new("scores", phi = bent.phi,
+ Dphi = bent.Dphi)

They are displayed graphically in the top left quadrant of Figure 7.

Below we implement the newly defined score functions using the free fatty acid data previously analysed using Wilcoxon scores. One could also use the scores provided by Rfit with option `scores=bentscores1` to obtain the same result.

```r
> summary(rfit(ffa ~ age + weight + skin,
+ scores = bentscores, data = ffa))

Call:
  rfit.default(formula = ffa ~ age + weight + skin,
               scores = bentscores, data = ffa)

Coefficients:
            Estimate Std. Error t.value p.value
age       -1.5957548  0.1888274   -8.4158 4.657e-08 ***
weight     0.00139487  0.00260504    0.5274  0.6023
skin       0.35619596  0.09090132    3.9185 8.367e-07 ***

Signif. codes:  < 0.001 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 1
```
The results are similar to those presented in Hettmansperger and McKean (2011).

Summary and future work

This paper illustrates the usage of a new R package, Rfit, for rank-based estimation and inference. Rank-based methods are robust to outliers and offer the data analyst an alternative to least squares. Rfit includes algorithms for general scores and a library of score functions is included. Functions for regression as well as one-way and multi-way anova are included. We illustrated the use of Rfit on several real data sets.

We are in the process of extending Rfit to include other robust rank-based procedures which are discussed in Chapters 3 and 5 of Hettmansperger and McKean (2011). These include autoregressive time-series models, cluster correlated data (mixed models), and nonlinear models. We are also developing weighted versions of rank-based estimation that can be used in mixed effects modeling as discussed in Kloke et al. (2009) as well as the computation of high breakdown rank-based estimation discussed in Chang et al. (1999).

Bibliography


