RobustBetaReg: An R package for robust beta regression model

Yuri S. Maluf and Silvia L. P. Ferrari

Abstract

The package **RobustBetaReg** provides functions and tools for fitting robust beta regression models and allowing different estimation methods (LSMLE and LMDPDE). Output of `robustbetareg` function return an S3 object of class `LSMLE` and `LMDPDE`, for which various generic methods are implemented, including `summary`, `plot`, and `residuals`. This package also brings several graphical tools to analyse the goodness-of-fit such as residuals with simulated envelope including robust version of Wald and Saddlepoint test. Some examples involving sport and simulated data are given to illustrate the implemented functions.

1 Introduction

The beta regression model is applied to modeling continuous response variables y in the unit interval, $y \in (0, 1)$, like rates, concentration, and proportions. Its application rises in various areas such as medicine, environment research, finance, and so forth. The inference of beta regression parameters is usually based on maximum likelihood estimation (MLE), for which kind several techniques are well implemented in **betareg** package. However, it is known that MLE is sensitive to outlier observations. In such a case, it can bring a serious bias and then erroneous inference. To that end, robust methods offer a solution to cope with those cases. In this paper, we present the R package **RobustBetaReg**, which should be used to perform inference in the presence of possible discrepant observations for beta regression models.

Base on the minimum density power divergence estimator (MDPDE), Ghosh (2019) developed a robust estimator for beta regression. Similarly, Ribeiro and Ferrari (2020) proposed a robust estimator which is called surrogate maximum likelihood estimator (SMLE) using L_q-likelihood, introduced by D. Ferrari and Yang (2010). Both estimators require a tuning parameter $\alpha \in [0, 1)$ which gives the trade-off between robustness and efficiency. For $\alpha > 0$ it provides a robust estimator and when $\alpha = 0$, it corresponds to the MLE. Nevertheless, those estimators suffer from a serious drawback. Given a tuning parameter value α, the existence of MDPDE and SMLE is limited to a subspace of the parameter space Θ, that is, $\Theta_\alpha \subset \Theta$. Clearly, it is not a desired feature of any sort of estimator. Employing a suitable transformation, this drawback is fixed and becomes applicable for the whole parameter space, which we call LMDPDE and LSMLE.

The `robustbetareg` function estimates beta regression parameters via robust estimators LMDPDE or LSMLE. This function returns LMDPDE and LSMLE objects

which belong to the S3 class. In the development of **RobustBetaReg**, we adopted the usage of the S3 class system, for all package functions, since it is easily understanding by most R users. Several package routines were developed with C and C++ programming languages. In order to boost the general computation performance, the package also allows executing the multi-threading process via `openMP` API. This is especially important in the computation of robust tests.

This manuscript reviews various features of **RobustBetaReg**. The package is available from the GitHub repository at https://github.com/yurimaluf/RobustBetaReg.git. The paper is organized as follows: Section 2 presents an outline of the robust method applied in this package. Section 3 abords two examples to demonstrate implemented functions and some of their options. Section 4 concludes.

2 Robust Beta Regression Estimation

2.1 LMDPDE and LSMLE

Introduced by S. Ferrari and Cribari-Neto (2004), the class of beta regression models uses an alternative parameterization of the conventional beta density given by

$$f_\theta(y;\mu,\phi) = \frac{y^{\mu\phi-1}(1-y)^{(1-\mu)\phi-1}}{B(\mu\phi,(1-\mu)\phi)}, \quad 0<y<1,$$

with $0<\mu<1$, $\phi>0$, and $B(\cdot,\cdot)$ is the beta function. We denote by $y \sim Beta(\mu,\phi)$ and the density function by f_θ. From the above parameterization, we have $\mathrm{E}(y)=\mu$ and $\mathrm{Var}(y)=\mu(1-\mu)/(1+\phi)$, hence μ is the mean parameter and ϕ can be interpreted as a precision parameter.

Let $y_1,...,y_n$ be independent random variables such that $y_i \sim Beta(\mu_i,\phi_i)$ for $i=1,...,n$. The mean and the precision parameter are modeled by

$$\begin{aligned} g_\beta(\mu_i) &= \boldsymbol{x}_i^\top \beta = \eta_{\beta i} \\ g_\gamma(\phi_i) &= \boldsymbol{z}_i^\top \gamma = \eta_{\gamma i} \end{aligned}$$

where $\beta=(\beta_1,...,\beta_{p_1})^\top$ and $\gamma=(\gamma_1,...,\gamma_{p_2})^\top$ are vectors of unknown regression coefficients ($p=p_1+p_2<n$); $\boldsymbol{x}_i=(x_{i1},...,x_{ip_1})^\top$ and $\boldsymbol{z}_i=(z_{i1},...,z_{ip_2})^\top$ are vectors of observed covariates, $\eta_{\beta i}$ and $\eta_{\gamma i}$ are linear predictors, and $\theta \in \Theta \subseteq \mathbb{R}^p$ is the parameter vector. The link functions $g_\beta:(0,1)\to\mathbb{R}$ and $g_\gamma:(0,\infty)\to\mathbb{R}$ are strictly increasing and, at least, twice differentiable.

The maximum likelihood estimator (MLE) is obtained solving

$$\sum_{i=1}^{n} \Psi(y_i;\theta) = 0 \tag{1}$$

where $\Psi(y_i;\theta)=\nabla_\theta \log(f_\theta(y_i;\mu_i,\phi_i))=(u_\mu(y_i,\mu_i,\phi_i),u_\phi(y_i,\mu_i,\phi_i))$ is the score vector with ∇_θ representing the gradient operator w.r.t. θ. In matrix notation the score equations (1) can be written as

$$\begin{aligned} X^\top \Phi T_\beta u_\mu &= 0, \\ Z^\top T_\gamma u_\phi &= 0, \end{aligned} \tag{2}$$

where X and Z are design matrices whose i-th row is $\boldsymbol{x}_i^\top$ and $\boldsymbol{z}_i^\top$ with rank p_1 and p_2 respectively, $\Phi = \text{diag}\{\phi_1, ..., \phi_n\}$, $T_\beta = \text{diag}\{1/g'_\beta(\mu_1), ..., 1/g'_\beta(\mu_n)\}$, $T_\gamma = \text{diag}\{1/g'_\gamma(\phi_1), ..., 1/g'_\gamma(\phi_n)\}$, $u_\mu = (u_{\mu_1}, ..., u_{\mu_n})$ with $\mu_i^* = E(y_i^*) = \psi(\mu_i\phi_i) - \psi((1-\mu_i)\phi_i)$, $\mu_i^\dagger = E\left(y_i^\dagger\right) = \psi((1-\mu_i)\phi_i) - \psi(\phi_i)$, $y_i^* = \log(y_i/(1-y_i))$, $y_i^\dagger = \log(1-y_i)$, and $\psi(\cdot)$ is the digamma function.

Although the MLE is known to be fully efficient, the presence of outliers leverage points might cause a critical problem since the maximum likelihood estimators are not robust for the beta regression model. To deal with this problem, estimators with robustness properties are desired. To that end, consider the logit transformation $y^* = \log\frac{y}{1-y}$. This transformation yields the following density function

$$f_\theta^*(y^*; \mu, \phi) = B^{-1}(\mu\phi, (1-\mu)\phi)\frac{e^{-y^*(1-\mu)\phi}}{(1+e^{-y^*})^\phi}, \quad \mu \in (0,1),\ \phi > 0 \quad \text{and} \quad y^* \in \mathbb{R}.$$

The distribution of y^* is called exponential generalized beta of the second type and we write $y^* \sim EGB(\mu, \phi)$. For more details see Kerman and McDonald (2015). Based on a minimum divergence estimation method, Basu et al. (1998) introduced the class of density power divergence (DPD) to obtain a robust estimator. Considering the transformed variable y^*, it yields the following estimating equation

$$\sum_{i=1}^{n} \Psi_\alpha^*(y_i^*; \theta) = 0. \tag{3}$$

We denote the estimator obtained by (3) as LMDPDE. Writing (3) in matrix notation, the robust estimator LMDPDE is given by the solution of the system of equation

$$\begin{aligned} X^\top \Phi T_\beta \left(W^{*\alpha} u_\mu^* - K_{\beta,\alpha}^*\right) &= 0 \\ Z^\top T_\gamma \left(W^{*\alpha} u_\phi^* - K_{\gamma,\alpha}^*\right) &= 0 \end{aligned} \tag{4}$$

where $W^* = \text{diag}\{f_\theta^*(y_1^*; \mu_1, \phi_1), ..., f_\theta^*(y_n^*; \mu_n, \phi_n)\}$, $K_{\beta,\alpha}^* = (\kappa_{\beta,\alpha,1}^*, ..., \kappa_{\beta,\alpha,n}^*)^\top$, $K_{\gamma,\alpha}^* = (\kappa_{\gamma,\alpha,1}^*, ..., \kappa_{\gamma,\alpha,n}^*)^\top$ with

$$\kappa_{\beta,\alpha,i}^* = c_{\alpha,i}^*\left(\mu_{\alpha,i}^* - \mu_i^*\right), \text{ and } \kappa_{\gamma,\alpha,i}^* = c_{\alpha,i}^*\left[\mu_i\left(\mu_{\alpha,i}^* - \mu_i^*\right) + (\mu_{\alpha,i}^{*\dagger} - \mu_i^\dagger)\right].$$

Note that when $\alpha = 0$ the (4) becomes the MLE, given by (2), since $W^{*0} = I$ and $K_{\beta,0}^* = K_{\gamma,0}^* = 0$. Note also that (4) resembles (2), but the main difference is the weight matrix W^*. The terms of matrix $K_{\beta,0}^*$ and $K_{\gamma,0}^*$ guarantee the Fisher consistency property of LMDPDE.

Based on $q-$entropy D. Ferrari and Yang (2010) introduced the maximum L_q-likelihood estimator (ML_qE). Using the empirical version of q-entropy, the ML_qE of beta regression model is obtained by maximization of L_q-likelihood

$$l_\alpha(\theta) = \sum_{i=1}^{n} L_{1-\alpha}(f_\theta(y_i; \mu_i, \phi_i))$$

where $L_{1-\alpha}(u) = \log(u)$ if $\alpha = 0$ and $L_{1-\alpha}(u) = (u^\alpha - 1)/\alpha$ otherwise. The L_q-likelihood of EGB provides an estimator that is not Fisher-consistent unless $\alpha = 0$. Nevertheless, D.

Ferrari and La Vecchia (2012) demonstrated that under a suitable re-parameterization, the Fisher consistency can be obtained if the postulated distribution is closed under power transformation. The power transformation of a density f_θ^* is given by

$$f_\theta^{[\xi]}(y^*) = \frac{f_\theta^{*\xi}(y^*)}{\int_{\mathbb{R}} f_\theta^{*\xi}(y^*)dy^*} = f_\theta^*(y^*; \mu, \phi_\alpha)$$

since $\int_{\mathbb{R}} f_\theta^{*\xi}(y^*)dy^* < \infty$, for $\xi \in (0, \infty)$, $\phi_\alpha = \phi/(1-\alpha)$ and the mean parameter remains the same.

It easily verified that EGB is closed under power transformation. Finally, using the $\mathrm{ML}_q\mathrm{E}$ for re-parameterization EGB with $\theta_\alpha = (\mu, \phi_\alpha)$ gives the following estimating equation

$$\sum_{i=1}^{n} \tilde{\Psi}_\alpha^*(y_i; \theta) = \sum_{i=1}^{n} \tilde{U}^*(y_i^*; \theta) f_{\theta'_\alpha}^{*\alpha}(y_i^*; \mu_i, \phi_i) = 0.$$

where $\tilde{U}^*(y_i^*; \theta) = \nabla_\theta \log\left(f_{\theta'_\alpha}^*(y_i^*; \mu_i, \phi_i)\right)$. In matrix notation we have

$$\begin{aligned} X^\top \Phi T_\beta \tilde{W}^{*\alpha} \tilde{u}_\mu^* &= 0 \\ Z^\top \tilde{T}_\gamma^* \tilde{W}^{*\alpha} \tilde{u}_\phi^* &= 0 \end{aligned} \tag{5}$$

where $\tilde{T}_\gamma^* = \mathrm{diag}\{1/\tilde{g}_\gamma^{*'}(\phi_1), ..., 1/\tilde{g}_\gamma^{*'}(\phi_n)\}$, $\tilde{W}^* = \mathrm{diag}\{f_{\theta'_\alpha}^*(y_1^*; \mu_1, \phi_1), ..., f_{\theta'_\alpha}^*(y_n^*; \mu_n, \phi_n)\}$, $\tilde{u}_\mu^* = (y^* - \mu_\alpha^*)$ and $\tilde{u}_\phi^* = \mu(y^* - \mu_\alpha^*) + y^\dagger - \mu_\alpha^{*\dagger}$. The Fisher consistency property of the estimator LSMLE, can be easily proved This brings some advantage of this estimator over LMDPDE, since it does not require the computation of an additional term to guarantee the Fisher consistency.

2.2 Robust test: Wald-type and Saddlepoint

In the framework of beta regression models, the standard classical asymptotic tests of parameters can be easily affected by the presence of the contaminated sample. Thus, it may lead to worrisome performance acceptance of the hypothesis. An alternative to cope with this problem is robust tests. Based on the asymptotic properties of MDPDE, Ghosh and Basu (2013) developed a robust version of the Wald test. It can also be extended to the LMDPDE and LSMLE. Considering the null hypothesis $h(\beta_0, \gamma_0) = \eta_0$, $h : \mathbb{R}^{p_1+p_2} \longrightarrow \mathbb{R}^d$ with $d \leq p_1 + p_2$, the robust version of Wald is given by

$$W_{d,\alpha}(\theta_\alpha) = (h(\theta_\alpha) - \eta_0)^\top \left[J_h(\theta_\alpha)\Omega(\theta_\alpha)J_h(\theta_\alpha)^\top\right]^{-1} (h(\theta_\alpha) - \eta_0),$$

with

$$\Omega(\theta_\alpha) = \Lambda(\theta_\alpha)^{-1}\Sigma(\theta_\alpha)\left[\Lambda(\theta_\alpha)^{-1}\right]^\top$$

where the $J_h(\theta)$ is the Jacobian matrix of h with rank d, $\Sigma(\theta_\alpha) = \mathrm{E}_{\theta_\alpha}\left[\Psi^\top\Psi\right]$ and $\Lambda(\theta_\alpha) = \mathrm{E}_{\theta_\alpha}\left[\nabla_\theta\Psi\right]$. The $W_{d,\alpha}$ is avaliated at the robust estimative. This test is called Wald-type test. $W_{d,\alpha}$ is asymptotically χ_d^2-distributed with d degree of freedom with an absolute error of order $O(n^{-\frac{1}{2}})$. When $\alpha = 0$ the test is equivalent to the non-robust classical Wald test. Thus, the test is a robust generalization of the classical Wald test.

Aiming to achieve a robust test statistic and good accuracy for small sample sizes, Robinson, Ronchetti, and Young (2003) introduced the saddlepoint test. Likewise, the Wald-type test, the saddlepoint test is asymptotically χ^2-distributed, but with relative error of order $O(n^{-1})$. Considering the null hypothesis $u(\theta) = \eta_0$, where $u : \mathbb{R}^p \longrightarrow \mathbb{R}^{p_1}$ with $p_1 \leq p$. The saddlepoint statistic test is $h(u(\theta))$, where

$$h(w) = \inf_{\{\theta : u(\theta) = w\}} \sup_{\lambda} \{-K_{\Psi}(\lambda; \theta)\} \tag{6}$$

and

$$K_{\Psi}(\lambda; \theta) = \log \mathrm{E}_{\theta_0} \left[\exp \left(\lambda^{\top} \Psi(Y; \theta) \right) \right] \tag{7}$$

is the cumulant generating function (cgf) of the score function Ψ and the expectation is taken under the null hypothesis. The test statistic $2nh(u(\hat{\theta}_{\alpha}))$ is unidimensional and under the null hypothesis is asymptotically $\chi^2_{p_1}$.

3 Fitting Robust Beta Regression

3.1 Estimation: AIS data

Manifold authors abort the MLE sensibility in the beta regression model under contaminated samples such as Bayes, Basan, and Garcia (2012), Ghosh (2019), Ribeiro and Ferrari (2020). They illustrate it through the analysis involving the body fat percentage of the 37 rowing athletes. The dataset was collected by the Australian Institute of Sport (AIS), and it is available in the R package `sn`.

The aim is to model the mean of the body fat percentage (BFP), as a function of the lean body mass (LBM). To illustrate the use of various **RobustBetaReg** package functions, we shall start with this dataset with the following model

$$\log \left(\frac{\mu_i}{1 - \mu_i} \right) = \beta_1 + \beta_2 LBM_i \text{ and } \log(\phi_i) = \gamma_1,$$

where μ_i and ϕ_i are the mean and precision of response variable BFP for the i-th athlete, with $i = 1, ..., 37$.

```
library(RobustBetaReg)
library(sn)
data(ais)
ais.row=subset(ais,sport=="Row")
ais.row$Bfat=ais.row$Bfat/100
fit.mle=robustbetareg(Bfat~LBM|1,data=ais.row,alpha = 0)
fit.lsmle=robustbetareg(Bfat~LBM|1,data=ais.row,alpha=0.2,type="LSMLE")
```

By default, the function `robustbetareg` considers the LSMLE, then the argument can be suppressed. Setting `alpha=0` is equivalent to MLE, see equations (5) and (4). Through the `type` argument, it is possible to select LMDPDE. The class LMDPDE has the same properties of LSMLE. Calling the generic function `summary` to the objects `fit.mle` and `fit.lsmle` gives the following results.

```r
summary(fit.mle)
```

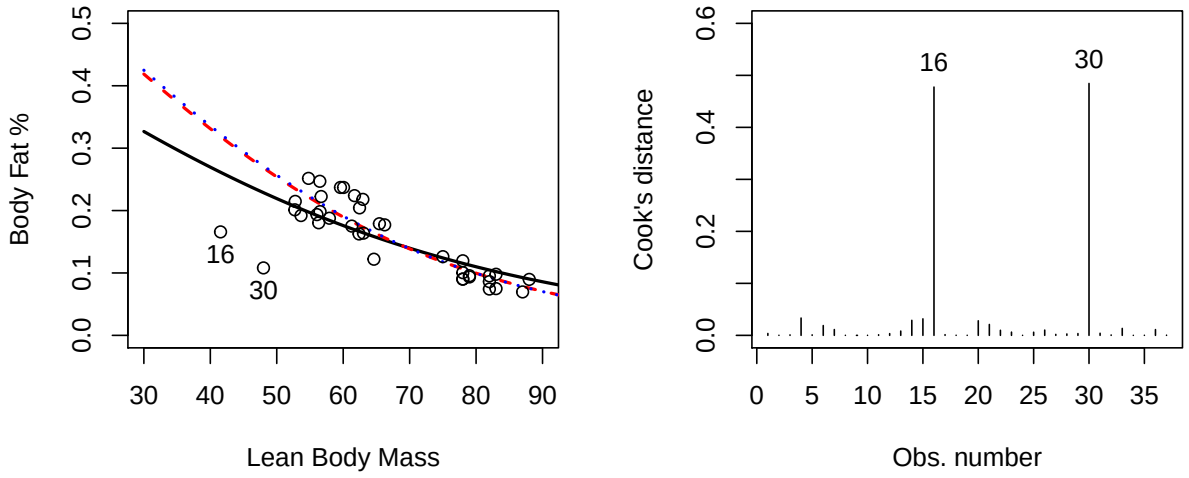
```
## Call:
## robustbetareg(formula = Bfat ~ LBM | 1, data = ais.row, alpha = 0)
##
## Coefficients (mean model with logit link):
##               Estimate Std. Error   z value  Pr(>|z|)
##  (Intercept)   0.09787     0.2531    0.3866     0.699
##          LBM  -0.02736   0.003893    -7.029 2.086e-12 ***
##
## Phi coefficients (precision model with log link):
##              Estimate Std. Error z value Pr(>|z|)
##  (Intercept)    4.571     0.2322   19.69   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Type of estimator: MLE
## Pseudo R-squared: 0.6699
## Tuning value: alpha=0
## Tuning of LSMLE selected by the user
```

```r
summary(fit.lsmle)
```

```
## Call:
## robustbetareg(formula = Bfat ~ LBM | 1, data = ais.row, alpha = 0.2,
##     type = "LSMLE")
##
## Coefficients (mean model with logit link):
##               Estimate Std. Error   z value  Pr(>|z|)
##  (Intercept)    0.7959     0.1788     4.451 8.539e-06 ***
##          LBM  -0.03746    0.00279    -13.43    <2e-16 ***
##
## Phi coefficients (precision model with log link):
##              Estimate Std. Error z value Pr(>|z|)
##  (Intercept)    5.341     0.2471   21.61   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Type of estimator: LSMLE
## Pseudo R-squared: 0.6699
## Tuning value: alpha=0.2
## Tuning of LSMLE selected by the user
```

Figure 1 shows the scatter plot of data along with the fitted line. This data contains two prominent outliers $\{16, 30\}$. Applying the robust estimator, the significance of the intercept term alters substantially. Moreover, the fitted line of LSMLE for full data and MLE for reduced data without the pair of observations $\{16, 30\}$ almost coincide. In other words, the influence of observations 16 and 30 in estimation is low for robust

estimator LSMLE. Before estimating, it is necessary to choose a tuning value. To help the user in this choice, we implemented the data-driven algorithm tuning selector. This algorithm is triggered if the value of tuning `alpha` is suppressed in `robustbetareg` function. Details about this algorithm can be found in Ribeiro and Ferrari (2020).



Fitted Line Cook's Distance

Figure 1: The fitted lines for the AIS data based on the MLE (solid black line) and LSMLE (dashed red line) at $\alpha = 0.2$ for the full data and the MLE (dotted blue line) for reduced data without observations 16 and 30.

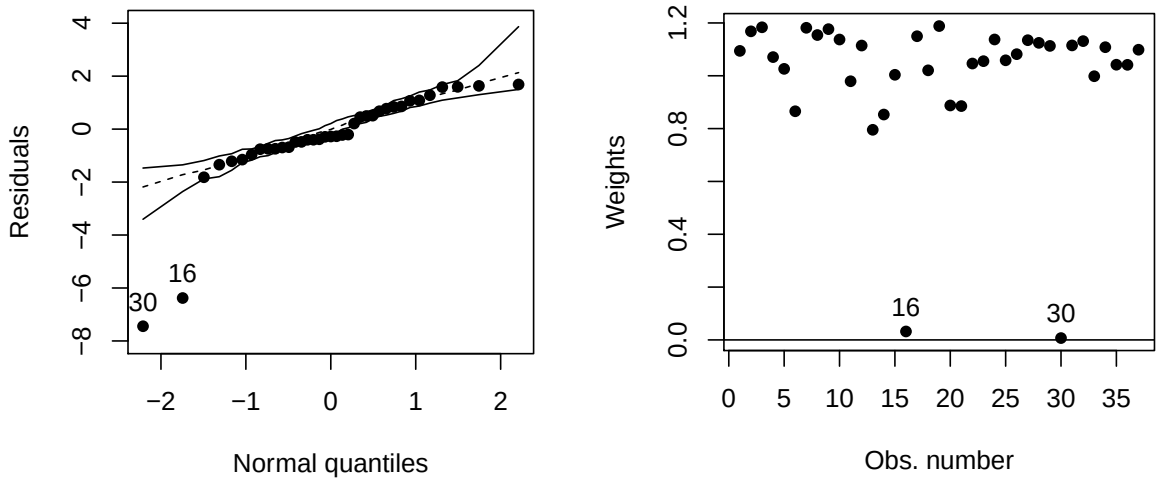
Through the arguments `link` and `link.phi` of `robustbetareg` function, we can set the links functions. The package includes several mean link and precision link functions. The implemented link functions are discribed in the Table 1. The default mean link functions is logit. In `formula` argument, if `|1` is suppressed the precision link function is the identity otherwise it is set by log function.

Table 1: List of mean and precision link functions implemented in package **RobustBetaReg**.

Mean link function	$g_\beta(\cdot)$	Precision link function	$g_\gamma(\cdot)$
logit (default)	$\log\left(\frac{\mu}{1-\mu}\right)$	log (default)	$log(\phi)$
probit	$\Phi(\mu)$ Φ means the cumulative Gaussian density	square root	$\sqrt{\phi}$
cloglog	$\log\left(-\log(1-\mu)\right)$		
cauchit	$tan(\pi(\mu - 1/2))$		
loglog	$-\log\left(-\log\left(\mu\right)\right)$		

The usual proceeding for beta regression for assessing goodness of fit is running a simulated normal envelope. The `plotenvelope` function provides the simulated envelope. The default argument is 100 replications and 95% confidence. Currently, there are 8 options of residuals, and it is set by `type` argument. All those kinds of implemented residuals can be found in Espinheira, Ferrari, and Cribari Neto (2008) and Espinheira, Santos, and Cribari Neto (2017). By default, the residual computed is the "standard weighted residual 2".

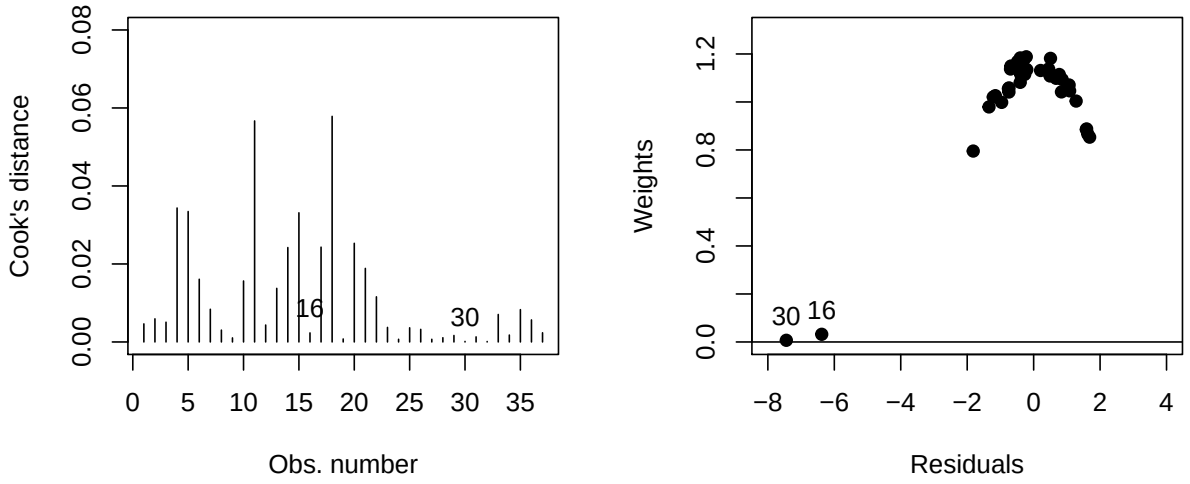
```
plotenvelope(fit.lsmle,ylim=c(-8,4))
plot(fit.lsmle)
```



(a) Simulated Envelopes

(b) Weights

Figure 2: Normal probability plot of residuals with simulated envelopes (a), the weights with $\alpha = 0.2$ (b), Cook's distance (c), and scatter plot of weights versus residuals (d), for LSMLE.



(a) Cook's Distance

(b) Residual x Weights

Figure 3: Normal probability plot of residuals with simulated envelopes (a), the weights with $\alpha = 0.2$ (b), Cook's distance (c), and scatter plot of weights versus residuals (d), for LSMLE.

In Figure 2b, the weight plot regards to the main diagonal of matrix $\tilde{W}^{*\alpha}$, described in equation (5). It is useful to detect outliers. The generic function `plot` applied to the objects of class LSMLE or LMDPDE displays, in console, an iterative options via menu bar. The menu bar allows users to select the graphic tools within the following options:

1. Residuals
2. Residuals x Linear predictor
3. Cook's Distance
4. Weights
5. Weigths x Residuals

Looking at the results on this example, the Cook's distance of observations 16 and 30 shrank for LSMLE. Their values are exhibit in Figure 3a.

3.2 Robust Test: Simulated Data

We shall illustrate the performance of robust tests through simulated data. For the sake of simplicity, we consider a beta regression model with logit mean link function, log precision link function, and one explanatory variable plus an intercept with the sample size $n = 40$. Two different scenarios are considered A and B. The first scenario is set with $(\beta_1, \beta_2) = (-1.4, 2.0)$ and $\gamma_1 = 5.5$. The last scenario is $(\beta_1, \beta_2) = (-1.8, -3.0)$ and $(\gamma_1, \gamma_2) = (6.0, -1.5)$. A fixed design matrix $X_{n\times 2}$ is generated by a uniform distribution $U(0, 1)$, and it is kept constant all over the simulation sampling process. All simulation results take into account 1000 replications and the tuning `alpha=0.05`. For the non-constant precision scenario, the used design matrix $Z_{n\times 2}$ is the same as

employed to the mean. Looking at the results on this example, the Cook's distance of observations 16 and 30 shrank for LSMLE. Their values are exhibit in Figure

The simulation process considers non-contaminated and contaminated sample. In the contamination process, we first generate a sample $y_1, ..., y_n$ from $y_i \sim Beta(\mu_i, \phi_i)$ with $i = 1, ..., n$. Then, m points $(y_{i_1}, ..., y_{i_m})$ are randomly selected from the sample with $m < n$. The percentage of contamination in the sample for both scenarios was set at 5% ($m = 0.05n$). Each contaminated response y_i is drawn by a beta distribution with different mean and precision parameters $y_{i_j} \sim Beta(\mu'_{\tau,i}, \phi'_{\tau,i})$, namely

$$\mu'_{\tau,i} = \frac{(1-\mu_i)^\tau}{(1-\mu_i)^\tau + \mu_i^\tau} \ , \ \phi'_{\tau,i} = \frac{\mu_i^\tau + (1-\mu_i)^\tau}{\phi_i^\tau \mu_i^\tau (1-\mu_i)^\tau}$$

and $\tau \geq -1$. The τ's value gives the magnitude of contamination, when $\tau = -1$ there is no contamination ($\mu'_{-1,i} = \mu_i$ and $\phi'_{-1,i} = \phi_i$). The contaminated observation y_{i_j} tends to exhibit a high value when its original selected point is low and vice versa. At the end of process, we replace each y_{i_j} into the original sample and then employ the logit transformation $y_i^* = \log(y_i/(1-y_i))$ for $i = 1, ..., n$. This contamination process is available in package through the `rbeta.cont` function.

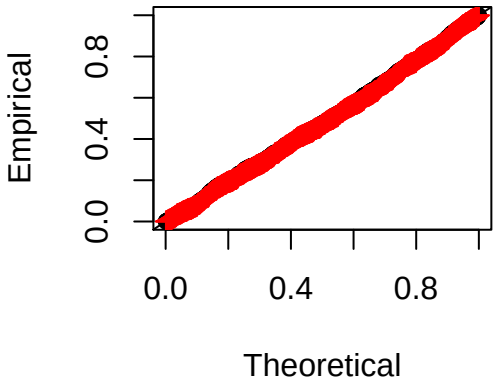
The analysis proceeds with the following null hypotheses:

$$H_0 : (\beta_1, \beta_2) = (\beta_{0,1}, \beta_{0,2}) \qquad \text{(Scenario A)}$$

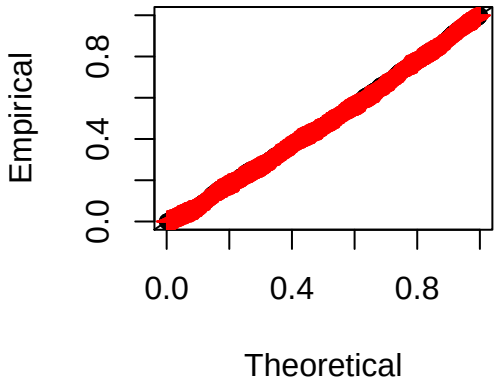
and

$$H_0 : (\beta_1, \beta_2, \gamma_2) = (\beta_{0,1}, \beta_{0,2}, \gamma_{0,2}) \qquad \text{(Scenario B)}$$

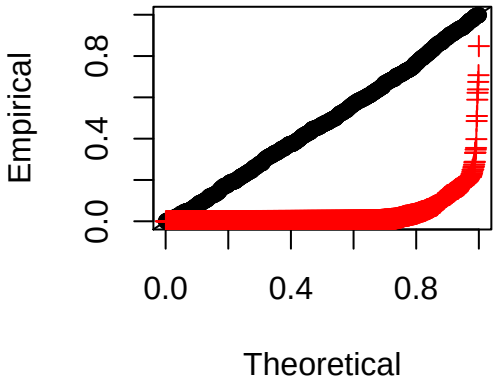
The Figures 4 and 5 display the PP-plots of p-values of Wald-type and Saddlepoint test against $U(0, 1)$ probabilities for robust and non-robust versions. In Scenario A, the contaminated sample has a slight effect on the performance for both robust tests and clearly, it is not true for the non-robust tests. In Scenario B, the robust tests softened the χ^2 approximation of the distribution when it is compared to the non-robust test. In the last scenario, the asymptotic χ^2 approximations of robust tests were relatively accurate. In general, the results show a minor better performance of Saddlepoint robust version than Wald-type robust version.



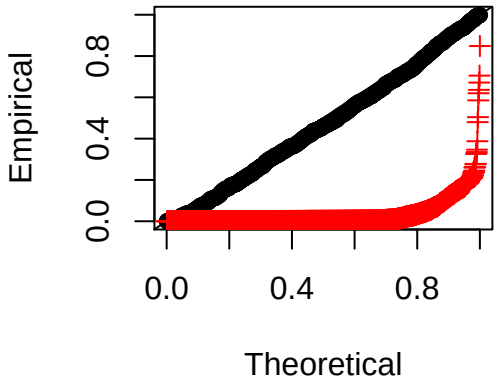
Saddlepoint for non-contaminated data



Wald-type for non-contaminated data

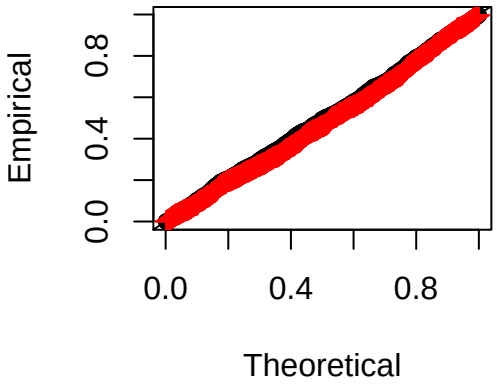


Saddlepoint for contaminated data

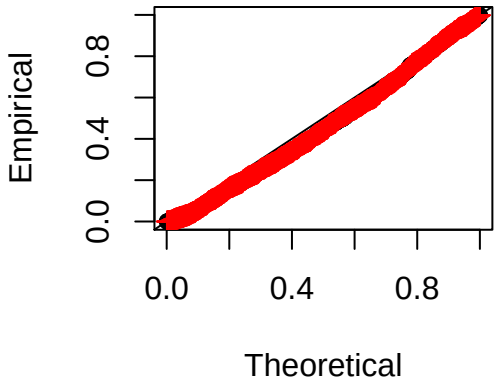


Wald-type for contaminated data

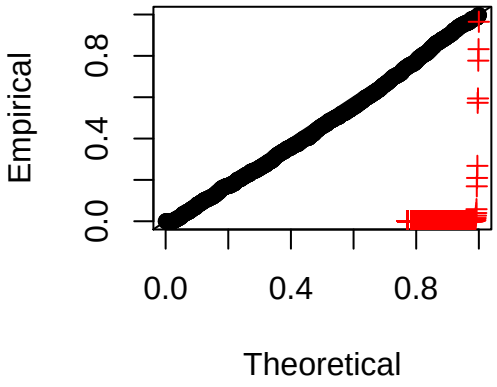
Figure 4: PP-plots of Saddlepoint (first column) and Wald-type (second column) tests null hypothesis and **Scenario A** with (second row) and without (first row) data contamination. The selected tuning parameter is $\alpha = 0.05$. The robust version is represented by black circle and non-robust version by red crosses.



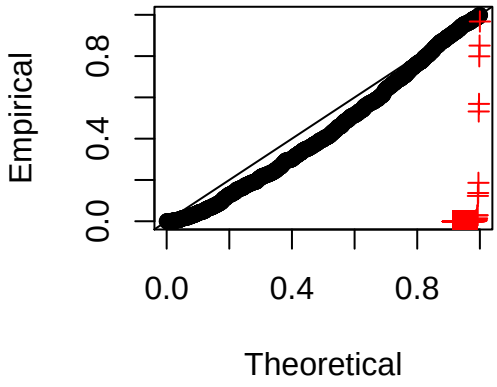
Saddlepoint for non-contaminated data



Wald-type for non-contaminated data



Saddlepoint for contaminated data



Wald-type for contaminated data

Figure 5: PP-plots of Saddlepoint (first column) and Wald-type (second column) tests null hypothesis and **Scenario B** with (second row) and without (first row) data contamination. The selected tuning parameter is $\alpha = 0.03$. The robust version is represented by black circle and non-robust version by red crosses.

For each simulated sample, y, the Wald-type test and Saddlepoint test are figured out by `WaldTypeTest` and `SaddlepointTest` functions. The first argument is object of the class LSMLE or LMDPDE. The second argument gives the hypothesis to be tested. The functions `WaldTypeTest` and `SaddlepointTest` return the following output:

```r
#Simulation setup of Scenario A
set.seed(111)
X=cbind(Z<-as.matrix(rep(1,n<-40)),X2<-runif(n))
B=c(-1.4,2)
G=c(5.5)
#Defining Hypothesis
h0.WT.L1=function(theta,B,G){theta[1:2]-B}#H0: Scenario A
h0.SP.L1=function(theta,B,G){c(B,theta[3])}#H0: Scenario A
#For each iteration
#Simulated contaminated sample
y=rbeta.cont(X%*%B,Z%*%G,ep=(n*0.05),seed=444,tau=-0.1)$sample
#Robust Estimation
fit.LSMLE=robustbetareg(y~X2|1)
#Robust Test
WaldTypeTest(fit.LSMLE,h0.WT.L1,B=B,G=G)
SaddlepointTest(fit.LSMLE,h0.SP.L1,B=B,G=G,thrd = 8)
```

```
## -- Wald Type Test --
## Null Hypothesis: set by the user
## Wald test=3.992, df=2, p-Value=0.1359
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Results based on LSMLE

## -- Saddlepoint Test --
## Null Hypothesis: set by the user
## Saddlepoint Test=3.744, df=2, p-Value=0.1538
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Results based on LSMLE with 8 threads
```

The entire computation of the saddlepoint test requires a cumbersome computational procedure to obtain the test statistic. This is especially critical for the composite hypothesis case because it requires the computation of numerical integration for each observation and for each maximization algorithm iteration within the minimization process. To improve the speedup of Saddlepoint statistic test the `RobustBetaReg` package allows the multithreading process through the argument `thrd`. By default, the package identifies the existing number of cores and sets it as the number of threads. On the other hand, the Wald-type statistic test is easily computed.

Table 2: Empirical size at 10%, 5% and 1% for Robust and non-Robust Saddlepoint and Wald-type test from null hypotheses for scenarios A ($\alpha = 0.05$) and B ($\alpha = 0.03$) under contaminated and non-contaminated sample.

			Scenario A			
Sample	Contaminated			Non-Contaminated		
Empirical Size	10%	5%	1%	10%	5%	1%
Non-Robust Saddlepoint	0.879	0.825	0.717	0.117	0.060	0.016
Non-Robust Wald-Type	0.886	0.830	0.727	0.125	0.078	0.020
Robust Saddlepoint	0.125	0.072	0.014	0.110	0.056	0.014
Robust Wald-Type	0.137	0.085	0.027	0.126	0.073	0.016
			Scenario B			
Sample	Contaminated			Non-Contaminated		
Empirical Size	10%	5%	1%	10%	5%	1%
Non-Robust Saddlepoint	0.971	0.970	0.967	0.109	0.056	0.012
Non-Robust Wald-Type	0.993	0.993	0.990	0.132	0.086	0.020
Robust Saddlepoint	0.112	0.073	0.032	0.090	0.043	0.013
Robust Wald-Type	0.182	0.120	0.043	0.131	0.086	0.022

Bearing in mind that the practitioners focus on the tail of χ^2 approximation, we compute the empirical size at the 10%, 5% and 1% levels of significance for Scenarios A and B. The results are displayed in Table 2. Robust tests are accurate, since the empirical sizes of robust tests are close to their theoretical size, which clearly do not occur from the non-robust tests. It is noteworthy that in general the results of robust Saddlepoint tend to exhibit slightly better performance than Wald-type test.

4 Conclusion

The R package `RobustBetaReg` allows the users to fit a robust beta regression. The package has also implemented robust tests and other tools. Through some examples, we introduced the main robust methods of `RobustBetaReg` package and other useful functions for fitting beta regression. Our aim while developing this package was to aggregate more methods and tools for beta regression models available for R community. The package is complemental of `betareg` package which is well known for most R users. More details of implemented robust methods can be found in Maluf and Ferrari (2021) and in the package manual available in our repository[1].

[1]https://github.com/yurimaluf/RobustBetaReg.git

References

Basu, A., I.R. Harris, N.L. Hjort, and M Jones. 1998. "Robust and Efficient Estimation by Minimising a Density Power Divergence." *Biometrika* 85: 549–59.

Bayes, C. L., J. L. Basan, and C. Garcia. 2012. "A New Robust Regression Model for Proportions." *Bayesian Analysis* 7: 841–66.

Espinheira, P. L., S. L. P. Ferrari, and F. Cribari Neto. 2008. "On Beta Regression Residuals." *Journal of Applied Statistics* 35: 407–19.

Espinheira, P. L., E. G. Santos, and F. Cribari Neto. 2017. "On Nonlinear Beta Regression Residuals." *Biometrical Journal* 59(3): 445–61.

Ferrari, D., and D. La Vecchia. 2012. "On Robust Estimation via Pseudo-Additive Information." *Biometrika* 99: 238–44.

Ferrari, D., and Y. Yang. 2010. "Maximum Lq-Likelihood Estimation." *The Annals of Statistics* 38: 753–83.

Ferrari, S.L.P., and F. Cribari-Neto. 2004. "Beta Regression for Modelling Rates and Proportions." *Journal of Applied Statistics* 31: 799–815.

Ghosh, A. 2019. "Robust Inference Under the Beta Regression Model with Application to Health Care Studies." *Statistical Methods in Medical Research* 28: 871–88.

Ghosh, A., and A. Basu. 2013. "Robust Estimation for Independent Non-Homogeneous Observations Using Density Power Divergence with Application to Linear Regression." *Electronic Journal of Statistics.* 32: 2420–56.

Kerman, S.C., and J. B. McDonald. 2015. "Skewness-Kurtosis Bounds for Egb1, Egb2, and Special Cases." *Communications in Statistics - Theory and Methods* 44: 3857–64.

Maluf, Yuri S., and Silvia L. P. Ferrari. 2021. "Robust Estimation Under Beta Regression Model."

Ribeiro, Terezinha K. A., and Silvia L. P. Ferrari. 2020. "Robust Estimation in Beta Regression via Maximum Lq-Likelihood."

Robinson, J., E. Ronchetti, and G. A. Young. 2003. "Saddlepoint Approximations and Tests Based on Multivariate M-Estimates." *The annals of statistic* 31: 1154–69.