Robust Regression in R

An Appendix to *An R Companion to Applied Regression*, third edition

John Fox & Sanford Weisberg

last revision: 2018-09-27

Abstract

Linear least-squares regression can be very sensitive to unusual data. In this appendix to Fox and Weisberg (2019), we describe how to fit several alternative robust-regression estimators, which attempt to down-weight or ignore unusual data: M-estimators; bounded-influence estimators; MM-estimators; and quantile-regression estimators, including L_1 regression.

All estimation methods rely on assumptions for their validity. We say that an estimator or statistical procedure is *robust* if it provides useful information even if some of the assumptions used to justify the estimation method are not applicable. Most of this appendix concerns *robust regression*, estimation methods, typically for the linear regression model, that are insensitive to outliers and possibly high-leverage points. Other types of robustness, for example to model misspecification, are not discussed here. These robust-regression methods were developed between the mid-1960s and the mid-1980s. The L_1 methods described in Section 5 are now probably the most widely used of these methods.

1 Breakdown and Robustness

The finite-sample breakdown point of an estimator or procedure is the smallest fraction γ of "bad" data values such that if the $[n\gamma]$ bad values grow towards infinity then the estimator or procedure also becomes infinite. The square brackets [] here represent rounding to the nearest whole number. For example, the sample mean of $x_1, \ldots, x_n$ can be written as an explicit function of one of the cases in the sample as

$$\begin{aligned}\bar{x}_n &= \frac{1}{n}\sum_{i=1}^{n} x_i \\ &= \frac{1}{n}\left[\sum_{i=1}^{n-1} x_i + x_n\right] \\ &= \frac{n-1}{n}\bar{x}_{n-1} + \frac{1}{n}x_n\end{aligned}$$

and so if x_n is large enough then $\bar{x}_n$ can be made as large as desired regardless of the other $n-1$ values. Thus the breakdown point of the sample mean is $1/n$.

Unlike the mean, the sample median, as an estimate of a population median, can tolerate up to 50% bad values. In general, the breakdown point cannot exceed 50%. (Reader: Why is that?)

2 M-Estimation

In linear regression, the breakdown of the ordinary least squares (OLS) estimator is analogous to the breakdown of the sample mean: A few extreme cases can largely determine the value of the

OLS estimator. In Chapter 8 of the *R Companion*, methods for detecting potentially influential points are presented, and these methods provide one approach to dealing with such points. Another approach is to replace ordinary least squares with an estimation method that is less affected by the outlying and influential points and can therefore produce useful results by accommodating the non-conforming data.

A general method of robust regression is called *M-estimation*, introduced by Huber (1964). This class of estimators can be regarded as a generalization of maximum-likelihood estimation, hence the "*M*."

We consider only the linear model that we write as[1]

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i \\ &= \mathbf{x}_i'\boldsymbol{\beta} + \varepsilon_i \end{aligned}$$

for the ith of n independent cases. We assume that the model itself is not at issue, so $\mathrm{E}(y|\mathbf{x}_i) = \mathbf{x}_i'\boldsymbol{\beta}$, but the distribution of the errors may be heavy-tailed, producing possibly many outliers. Given an estimator $\mathbf{b}$ for $\boldsymbol{\beta}$, the fitted model is

$$\widehat{y}_i = b_0 + b_1 x_{i1} + b_2 x_{i2} + \cdots + b_p x_{ip} + e_i = \mathbf{x}_i'\mathbf{b}$$

and the residuals are given by

$$e_i = y_i - \widehat{y}_i = y_i - \mathbf{x}_i'\mathbf{b}$$

The idea in M-estimation is to pay less attention to cases for which the residuals are large, because these potential outliers contain less information about the location of the regression surface than will non-outliers. In M-estimation, the estimates $\mathbf{b}$ are determined by minimizing a particular *objective function* over all $\mathbf{b}$,

$$\sum_{i=1}^{n} \rho(e_i) = \sum_{i=1}^{n} \rho(y_i - \mathbf{x}_i'\mathbf{b}) \tag{1}$$

where the function ρ gives the contribution of each residual to the objective function. A reasonable ρ should have the following properties:

- always-non negative, $\rho(e) \geq 0$;
- equal to 0 when the residual is 0, $\rho(0) = 0$;
- symmetric, $\rho(e) = \rho(-e)$, although in some problems one might argue that symmetry is undesirable; and
- monotone in $|e_i|$, $\rho(e_i) \geq \rho(e_{i'})$ for $|e_i| > |e_{i'}|$.

For least squares, we have $\rho(e) = e^2$, so the objective function minimized is the sum of squared residuals. The choice of $\rho(e) = |e|$ corresponds to L_1 regression discussed in Section 5. The ρ functions of M-estimation are more complicated, and described below.

The least-squares ρ-function is not robust because $\rho(e)$ gets larger at the rate e^2, so cases with large residuals dominate the estimation process. The L_1 ρ function, and the ρ functions used in M-estimation, increase more slowly and pay less attention to large $|e|$ in computing estimates

[1] If you're unfamiliar with matrix and vector notation, simply think of $\mathbf{x}_i'$ as the collection of x values for the ith case (including an initial value 1 for the regression constant), $\boldsymbol{\beta}$ as the collection of population regression coefficients, and $\mathbf{b}$ (below) as the collection of sample regression coefficients.

2.1 Computing *M*-Estimates*

Regardless of the specific choice for ρ, we can describe a general algorithm for the computations. The minimum value of Equation 1 can be found by differentiating with respect to the argument $\mathbf{b}$, and setting the resulting partial derivatives to 0:

$$\begin{aligned}\mathbf{0} &= \frac{\partial}{\partial \mathbf{b}} \sum_{i=1}^{n} \rho(y_i - \mathbf{x}_i'\mathbf{b}) \\ &= \sum_{i=1}^{n} \psi(y_i - \mathbf{x}_i'\mathbf{b})\mathbf{x}_i'\end{aligned}$$

where the *influence curve* ψ is defined to be the derivative of ρ with respect to its argument.

To facilitate computing, we would like to make this last equation similar to the estimating equations for a familiar problem like weighted least squares. To this end, define the *weight function* $w_i = w(e_i) = \psi(e_i)/e_i$. The estimating equations may then be written as

$$\sum_{i=1}^{n} w_i(y_i - \mathbf{x}_i'\mathbf{b})\mathbf{x}_i' = \mathbf{0}$$

Solving these estimating equations is equivalent to a weighted least-squares problem, minimizing $\sum w_i^2 e_i^2$. The weights, however, depend upon the residuals, the residuals depend upon the estimated coefficients, and the estimated coefficients depend upon the weights. An iterative solution called *iteratively reweighted least-squares* or *IRLS*) is therefore required:

1. Select initial estimates $\mathbf{b}^{(0)}$, such as the least-squares estimates.
2. At each iteration t, calculate residuals $e_i^{(t-1)}$ and associated weights $w_i^{(t-1)} = w\left[e_i^{(t-1)}\right]$ from the previous iteration.
3. Solve for new weighted-least-squares estimates

$$\mathbf{b}^{(t)} = \left[\mathbf{X}'\mathbf{W}^{(t-1)}\mathbf{X}\right]^{-1}\mathbf{X}'\mathbf{W}^{(t-1)}\mathbf{y}$$

where $\mathbf{X}$ is the model matrix, with $\mathbf{x}_i'$ as its ith row, and $\mathbf{W}^{(t-1)} = \text{diag}\left\{w_i^{(t-1)}\right\}$ is the current weight matrix.

Steps 2 and 3 are repeated until the estimated coefficients converge.

The asymptotic covariance matrix of $\mathbf{b}$ is

$$\mathcal{V}(\mathbf{b}) = \frac{E(\psi^2)}{[E(\psi')]^2}(\mathbf{X}'\mathbf{X})^{-1}$$

Using $\sum[\psi(e_i)]^2$ to estimate $E(\psi^2)$, and $[\sum \psi'(e_i)/n]^2$ to estimate $[E(\psi')]^2$ produces the *estimated* asymptotic covariance matrix, $\widehat{\mathcal{V}}(\mathbf{b})$ (which is not reliable in small samples).[2]

2.2 Objective Functions*

Figure 1 compares the objective functions, and the corresponding ψ and weight functions for three M-estimators: the familiar least-squares estimator; the *Huber* estimator; and the *Tukey bisquare* (or *biweight*) *estimator*. The objective and weight functions for the three estimators are also given in Table 1.

[2]In the on-line appendix to the *R Companion* on bootstrapping, we bootstrap a robust regression to obtain more realistic standard errors and confidence intervals in a small sample.

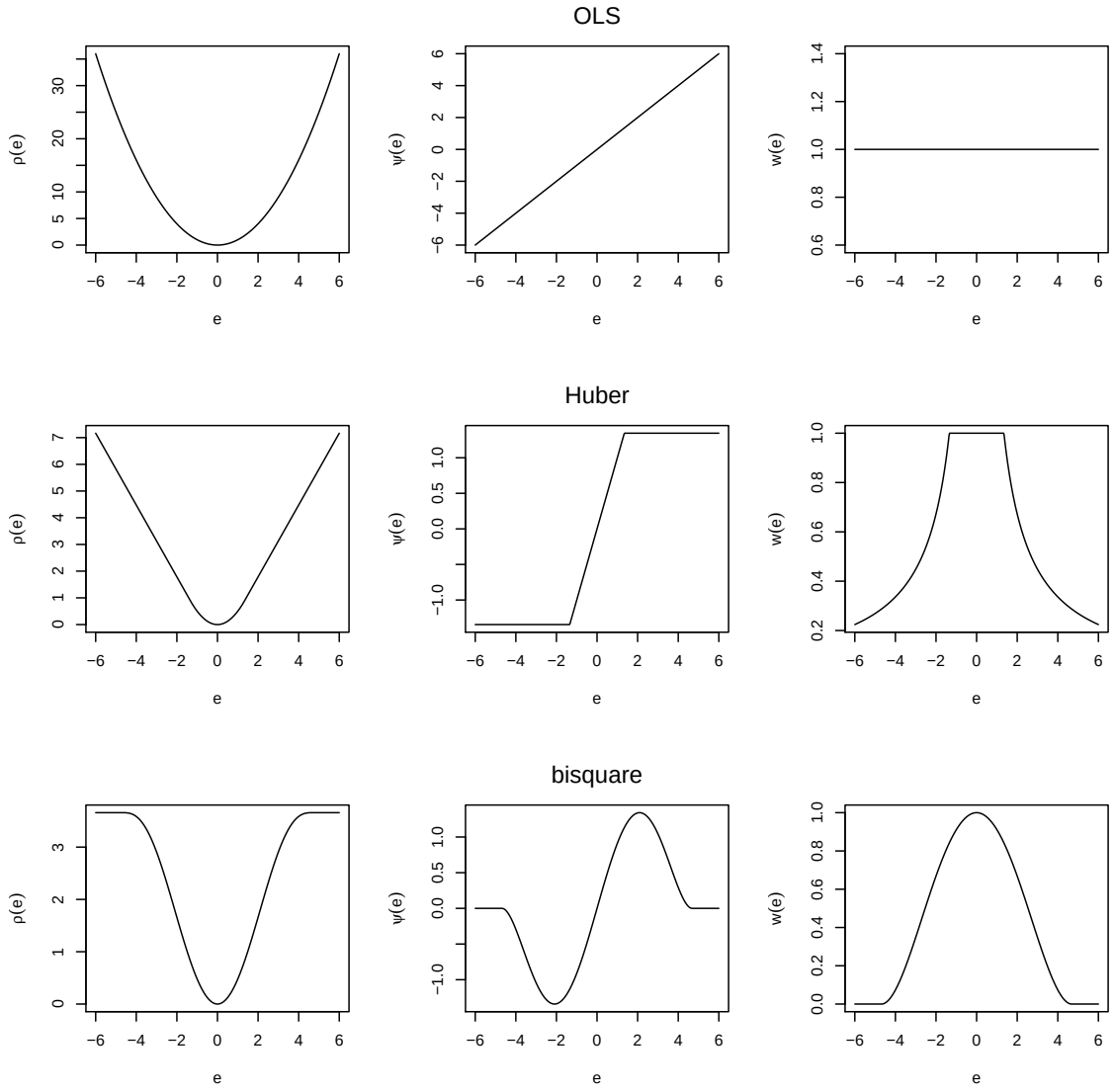

Figure 1: Objective (left), ψ (center), and weight (right) functions for the least-squares (top), Huber (middle), and bisquare (bottom) estimators. The tuning constants for these graphs are $k = 1.345$ for the Huber estimator and $k = 4.685$ for the bisquare. (One way to think about this scaling is that the standard deviation of the errors, σ, is taken as 1.)

Method	*Objective Function*	*Weight Function*
Least-Squares	$\rho_{\mathrm{LS}}(e) = e^2$	$w_{\mathrm{LS}}(e) = 1$
Huber	$\rho_{\mathrm{H}}(e) = \begin{cases} \frac{1}{2}e^2 & \text{for} \lvert e\rvert \le k \\ k\lvert e\rvert - \frac{1}{2}k^2 & \text{for} \lvert e\rvert > k \end{cases}$	$w_{\mathrm{H}}(e) = \begin{cases} 1 & \text{for} \lvert e\rvert \le k \\ k/\lvert e\rvert & \text{for} \lvert e\rvert > k \end{cases}$
Bisquare	$\rho_{\mathrm{B}}(e) = \begin{cases} \frac{k^2}{6}\left\{1 - \left[1 - \left(\frac{e}{k}\right)^2\right]^3\right\} & \text{for} \lvert e\rvert \le k \\ k^2/6 & \text{for} \lvert e\rvert > k \end{cases}$	$w_{\mathrm{B}}(e) = \begin{cases} \left[1 - \left(\frac{e}{k}\right)^2\right]^2 & \text{for} \lvert e\rvert \le k \\ 0 & \text{for} \lvert e\rvert > k \end{cases}$

Table 1: Objective functions and weight functions for least-squares, Huber, and bisquare estimators.

Both the least-squares and Huber objective functions increase without bound as the residual e departs from 0, but the least-squares objective function increases more rapidly. In contrast, the bisquare objective function eventually levels off (for $|e| > k$). Least-squares assigns equal weight to each case; the weights for the Huber estimator decline when $|e| > k$; and the weights for the bisquare decline as soon as e departs from 0, and are 0 for $|e| > k$. The ψ function of the bisquare estimator *redescends* to 0 for sufficiently large residuals.

The value k for the Huber and bisquare estimators is called a *tuning constant*; smaller values of k produce more resistance to outliers, but at the expense of lower efficiency when the errors are normally distributed. The tuning constant is generally picked to give reasonably high efficiency in the normal case; in particular, $k = 1.345\sigma$ for the Huber and $k = 4.685\sigma$ for the bisquare (where σ is the standard deviation of the errors) produce 95-percent efficiency when the errors are normal, and still offer protection against outliers.

In an application, we need an estimate of the standard deviation of the errors to use these results. Usually a robust measure of variation is employed in preference to the standard deviation of the residuals. For example, a common approach is to take $\widehat{\sigma} = \text{MAR}/0.6745$, where MAR is the median absolute residual.

3 Bounded-Influence Regression

Under certain circumstances, M-estimators can be vulnerable to high-leverage cases. Very-high-breakdown *bounded-influence estimators* for regression have been proposed and R functions for them are presented here. Very-high-breakdown estimates should be avoided, however, unless we have faith that the model we are fitting is correct, because these estimates do not allow for diagnosis of model misspecification (Cook et al., 1992).

One bounded-influence estimator is *least-trimmed squares* (*LTS*) regression. Order the squared residuals from smallest to largest:

$$\left[|e|^2_{(1)}, |e|^2_{(2)}, \ldots, |e|^2_{(n)},\right]$$

The LTS estimator chooses the regression coefficients $\mathbf{b}$ to minimize the sum of the smallest m of the squared residuals,

$$\text{LTS}(\mathbf{b}) = \sum_{i=1}^{m} |e|^2_{(i)}$$

where, typically, $m = \lfloor n/2 \rfloor + \lfloor (k+2)/2 \rfloor$, a little more than half of the cases, and the floor brackets, $\lfloor\ \rfloor$, denote rounding down to the next smallest integer.

While the LTS criterion is easily described, the mechanics of fitting the LTS estimator are complicated (Rousseeuw and Leroy, 1987). Moreover, bounded-influence estimators can produce unreasonable results in certain circumstances (Stefanski, 1991), and there is no simple formula for coefficient standard errors.[3]

One application of bounded-influence estimators is to provide starting values for M-estimation. This procedure, along with using the bounded-influence estimate of the error variance, produces the so-called *MM-estimator*. The *MM*-estimator retains the high breakdown point of the bounded-influence estimator and shares the relatively high efficiency under normality of the traditional M-estimator. *MM*-estimators are especially attractive when paired with redescending ψ-functions such as the bisquare, which can be sensitive to starting values.

[3]Statistical inference for the LTS estimator can be performed by bootstrapping, however. See Chapter 5 in the text and the on-line appendix on bootstrapping.

4 An Illustration: Duncan's Occupational-Prestige Regression

Duncan's occupational-prestige regression was introduced in Chapter 1 of the *R Companion*. The least-squares regression of `prestige` on `income` and `education`, for the `Duncan` data set in the **carData** package, produces the following results:[4]

```
library("car")
```

```
Loading required package: carData
```

```
mod.ls <- lm(prestige ~ income + education, data=Duncan)
S(mod.ls)
```

```
Call: lm(formula = prestige ~ income + education, data = Duncan)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -6.0647     4.2719   -1.42     0.16
income        0.5987     0.1197    5.00  1.1e-05
education     0.5458     0.0983    5.56  1.7e-06

Residual standard deviation: 13.4 on 42 degrees of freedom
Multiple R-squared: 0.828
F-statistic:  101 on 2 and 42 DF,  p-value: <2e-16
   AIC    BIC
365.96 373.19
```

Recall from the discussion of Duncan's data in Chapters 1 and 8 of the *R Companion* that two cases, ministers and railroad conductors (numbers 6 and 16, respectively), serve to decrease the `income` coefficient and to increase the `education` coefficient, as we may verify by omitting these two cases from the regression:

```
mod.ls.2 <- update(mod.ls, subset=-c(6, 16))
brief(mod.ls, mod.ls.2)
```

```
           income education
Estimate    0.599    0.5458
Std. Error  0.120    0.0983
```

The fit of the Huber M-estimator for Duncan's regression model, using the `rlm()` (`r`obust `l`inear `m`odel) function in the **MASS** package is

```
library("MASS")
```

```
mod.huber <- rlm(prestige ~ income + education, data=Duncan)
compareCoefs(mod.ls, mod.ls.2, mod.huber)
```

```
Calls:
1: lm(formula = prestige ~ income + education, data = Duncan)
2: lm(formula = prestige ~ income + education, data = Duncan, subset = -c(6, 16))
```

[4] Any R functions used but not described in this appendix are discussed in the *R Companion*. All the R code used in this appendix can be downloaded from `http://tinyurl.com/carbook` or via the `carWeb()` function in the **car** package.

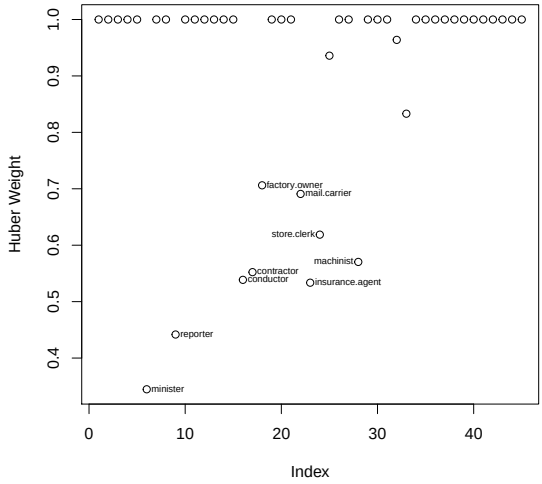

Figure 2: Weights from the robust Huber estimator for the regression of `prestige` on `income` and `education`.

```
3: rlm(formula = prestige ~ income + education, data = Duncan)

            Model 1 Model 2 Model 3
(Intercept)   -6.06   -6.41   -7.11
SE             4.27    3.65    3.88

income         0.599   0.867   0.701
SE             0.120   0.122   0.109

education    0.5458  0.3322  0.4854
SE           0.0983  0.0987  0.0893
```

The Huber regression coefficients are between those produced by the least-squares fit to the full data set and the least-squares fit eliminating the occupations `minister` and `conductor`.

It is instructive to extract and plot (in Figure 2) the final weights used in the robust fit. The `showLabels()` function from the **car** package is employed to label all cases with weights less than 0.8:

```
plot(mod.huber$w, ylab="Huber Weight")
smallweights <- which(mod.huber$w < 0.8)
showLabels(1:45, mod.huber$w, rownames(Duncan),
           method=smallweights, cex=0.6)

      minister        reporter       conductor      contractor   factory.owner
             6               9              16              17              18
  mail.carrier insurance.agent     store.clerk       machinist
            22              23              24              28
```

Ministers and conductors are among the cases that receive the smallest weight.

The `rlm()` function can also fit the bisquare estimator. Starting values for the IRLS procedure are potentially more critical for the bisquare estimator; specifying the argument `method="MM"` to `rlm()` requests bisquare estimates with start values determined by a preliminary bounded-influence regression:

```
mod.bisq <- rlm(prestige ~ income + education, data=Duncan, method="MM")
compareCoefs(mod.huber, mod.bisq)
```

```
Calls:
1: rlm(formula = prestige ~ income + education, data = Duncan)
2: rlm(formula = prestige ~ income + education, data = Duncan, method = "MM")

            Model 1 Model 2
(Intercept)   -7.11   -7.39
SE             3.88    3.91

income        0.701   0.783
SE            0.109   0.109

education    0.4854  0.4233
SE           0.0893  0.0899
```

Compared to the Huber estimates, the bisquare estimate of the `income` coefficient is larger, and the estimate of the `education` coefficient is smaller. Figure 3 shows a graph of the weights from the bisquare fit, identifying the cases with the smallest weights:

```
plot(mod.bisq$w, ylab="Bisquare Weight")
showLabels(1:45, mod.bisq$w, rownames(Duncan),
     method=which(mod.bisq$w < 0.8), cex=0.6)
```

```
          minister           reporter          conductor         contractor
                 6                  9                 16                 17
     factory.owner       mail.carrier    insurance.agent        store.clerk
                18                 22                 23                 24
         machinist streetcar.motorman
                28                 33
```

Finally, the `ltsreg()` function in the **lqs** package is used to fit Duncan's model by LTS regression:[5]

```
(mod.lts <- ltsreg(prestige ~ income + education, data=Duncan))
```

```
Call:
lqs.formula(formula = prestige ~ income + education, data = Duncan,
    method = "lts")

Coefficients:
(Intercept)       income    education
     -6.399        0.805        0.420

Scale estimates 7.78 7.57
```

[5] LTS regression is also the default method for the `lqs()` function, which additionally can fit other bounded-influence estimators.

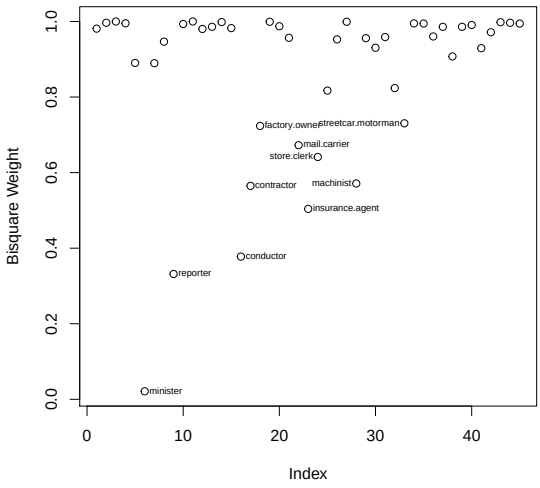

Figure 3: Weights from the robust bisquare estimator for the regression of `prestige` on `income` and `education`.

In this case, the results are similar to those produced by the M-estimators. The `print()` method for bounded-influence regression gives the regression coefficients and two estimates of the variation or scale of the errors. There is no `summary()` method for this class of models.

5 L_1 and Quantile Regression

This section follows Koenker (2005) and the vignette for quantile regression in the **quantreg** package in R. We start by assuming a model like this:

$$y_i = \mathbf{x}_i'\boldsymbol{\beta} + \varepsilon_i \tag{2}$$

where the ε_i are random errors. L_1 regression estimates $\boldsymbol{\beta}$ by solving the minimization problem

$$\widetilde{\boldsymbol{\beta}} = \arg\min \frac{1}{n}\sum_{i=1}^{n} |y_i - \mathbf{x}_i'\boldsymbol{\beta}| \tag{3}$$

If the ε_i are independent and identically distributed from a double exponential distribution, then $\widetilde{\boldsymbol{\beta}}$ is the maximum likelihood estimate for $\boldsymbol{\beta}$. In general L_1 regression estimates the *median* y at $\mathbf{x}_i'\boldsymbol{\beta}$, so one can think of this as *median regression*.

Figure 4 illustrates the results of fitting L_1 regression in various situations. In each of the four cases illustrated, $N_1 = 100$ random draws from a bivariate normal distribution with mean $E(X, Y)' = (0, 0)$‘ variances equal to 1, and correlation equal to 0.9 are generated. These are shown by the black points in each of the graphs in Figure 4. Then, a sample of N_2 “bad” observations were drawn from a bivariate normal but with mean $(1.5 - 1.5)$ with variances $(0.2, 0.2)$ and correlation zero. These are the magenta points on the plots, with $N_2 \in (20, 30, 75, 100)$.

Three regression lines are shown in each plot: OLS fit to the N_1 good data points; OLS fit to all the data; and median regression fit to all the data. If the goal is to match, more or less, the OLS regression fit to the good data, then the median regression does a respectable jobs for $N_2 \leq 30$,

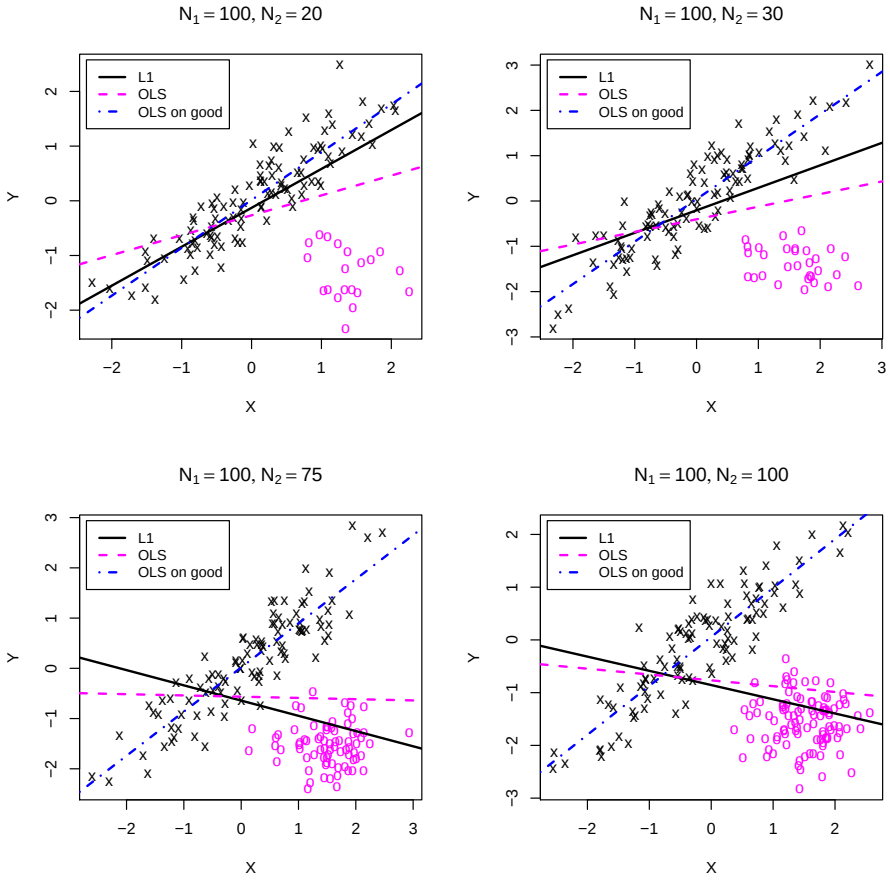

Figure 4: Simulated data with an increasing number of "bad" or "outlying" cases. Three lines are shown in each panel: the L_1 regression (solid black line); OLS fit to all of the data (broken magenta line); OLS fit to the "good" data points (dot-dash blue line).

but it does no better than OLS on all the data for larger N_2. Of course in these latter cases the distinction between "good" and "bad" data is hard to justify.

For the interested reader, here is the code:

```
library("quantreg")
library("MASS") # for the mvrnorm function

set.seed(10131986) # for reproducibility
l1.data <- function(n1=100, n2){
  data <- mvrnorm(n=n1,mu=c(0, 0),
                  Sigma=matrix(c(1, .9, .9, 1), ncol=2))
# generate n2 'bad' cases
  data <- rbind(data, mvrnorm(n=n2,
                  mu=c(1.5, -1.5), Sigma=.2*diag(c(1, 1))))
  data <- data.frame(data)
  names(data) <- c("X", "Y")
  ind <- c(rep(1, n1),rep(2, n2))
  plot(Y ~ X, data, pch=c("x", "o")[ind],
       col=c("black", "magenta")[ind],
       main=substitute(list(N[1] == n1, N[2] == n2), list(n1=n1, n2=n2)))
  r1 <-rq(Y ~ X, data=data, tau=0.5, ci=FALSE)
  abline(r1, lwd=2)
  abline(lm(Y ~ X, data), lty=2, lwd=2, col="magenta")
  abline(lm(Y ~ X, data, subset=1:n1), lty=4, lwd=2, col="blue")
  legend("topleft", c("L1","OLS","OLS on good"),
         inset=0.02, lty=c(1, 2, 4), lwd=2, col=c("black", "magenta", "blue"),
         cex=.9)}
par(mfrow=c(2, 2))
l1.data(100, 20)
l1.data(100, 30)
l1.data(100, 75)
l1.data(100, 100)
```

5.1 L_1 Facts*

1. The L_1 estimator is the maximum likelihood estimator if the errors are independent with a double-exponential distribution.

2. In Equation 2 (page 9) if $\mathbf{x}$ consists only of the constant regressor (1), then the L_1 estimator is the median.

3. Computations are not nearly as easy as for least squares, because a linear programming solution is required for L_1 regression.

4. L_1 is *equivalent*, meaning that replacing $\mathbf{y}$ by $a + b\mathbf{y}$ and $\mathbf{X}$ by $\mathbf{A} + \mathbf{B}^{-1}\mathbf{X}$ (where a, b, $\mathbf{A}$, and $\mathbf{B}$ are constants) will leave the solution essentially unchanged.

5. The breakdown point of the L_1 estimate can be shown to be $1 - 1/\sqrt{2} \approx 0.29$, so about 29% "bad" data can be tolerated.

6. In general L_1 regression estimates the median of $y|\mathbf{x}$, not the conditional mean.

7. Suppose we have Equation 2 (page 9) with the errors independent and identically distributed from a distribution F with density f. The population median is $\xi_\tau = F^{-1}(\tau)$ with $\tau = 0.5$,

and the sample median is $\widehat{\xi}_{.5} = \widehat{F}^{-1}(0.5)$. We assume a standardized version of f so $f(u) = (1/\sigma)f_0(u/\sigma)$. Write $\mathbf{Q}_n = n^{-1}\sum \mathbf{x}_i\mathbf{x}_i'$, and suppose that in large samples $\mathbf{Q}_n \to \mathbf{Q}_0$, a fixed matrix. We will then have

$$\sqrt{n}(\widetilde{\boldsymbol{\beta}} - \boldsymbol{\beta}) \sim \mathrm{N}(\mathbf{0}, \omega\mathbf{Q}_0^{-1})$$

where $\omega = \sigma^2\tau(1-\tau)/\{f_0[F_0^{-1}(\tau)]\}^2$ and $\tau = 0.5$. For example, if f is the standard normal density, $f[F_0^{-1}(\tau)] = 1/\sqrt{2\pi} = 0.399$, and $\sqrt{\omega} = 0.5\sigma/0.399 = 1.26\sigma$, so in the normal case the standard deviations of the L_1 estimators are 26% larger than the standard deviations of the OLS estimators.

8. If f were known, asymptotic Wald tests and confidence intervals could be based on percentiles of the normal distribution. In practice, $f[F^{-1}(\tau)]$ must be estimated. One standard method due to Siddiqui is to estimate

$$\widehat{f[F^{-1}(\tau)]} = \left[\widehat{F}^{-1}(\tau + h) - \widehat{F}^{-1}(\tau - h)\right]/2h$$

for some bandwidth parameter h. This approach is closely related to density estimation, and so the value of h used in practice is selected by a method appropriate for density estimation.

Alternatively, $f[F^{-1}(\tau)]$ can be estimated using a bootstrap procedure.

9. For non-independent and identically distributed errors, suppose that $\xi_i(\tau)$ is the τ-quantile for the distribution of the ith error. One can show that

$$\sqrt{n}(\widetilde{\boldsymbol{\beta}} - \boldsymbol{\beta}) \sim \mathrm{N}\left[\mathbf{0}, \tau(1-\tau)\mathbf{H}^{-1}\mathbf{Q}_0\mathbf{H}^{-1}\right]$$

where the matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i\mathbf{x}_i' f_i\xi_i(\tau)$$

and thus a sandwich-type estimator is used for estimating the variance of $\widetilde{\boldsymbol{\beta}}$. The `rq()` function in the **quantreg** package uses a sandwich formula by default for computing coefficient standard errors.

6 Quantile regression

L_1 regression minimizes the absolute errors, providing an estimate of the median of the response at any given value of the predictors. Quantile regression allows for estimating the other quantiles such as the quartiles, or the fifth or 95-th percentiles of a distribution. In the `rq` function in the `quantreg` package the user merely needs to specify the quantile `tau` of interest.

6.1 Example: Salary Data

This example examines salary as a function of job difficulty for job classes in a large governmental unit. Points are marked according to whether or not the fraction of female employees in the class exceeds 80%. The data are shown in Figure 5. Because the dependence of the response on the predictor is apparently curved, we model the response with a 5-*df* B-spline (see Section 4.4.2 of the *R Companion*), using the model formula `MaxSalary ~ bs(Score, 5)`. We will estimate the median regression, as well as the 0.10 and 0.90 quantile regressions:

```
library("alr4") # for data
library("splines")
```

```
fdom <- with(salarygov, NW/NE > .8)
taus <- c(.1, .5, .9)  # estimate 10%, median, and 90% quantiles
ltys <- c(2, 1, 2)
cols <- c("blue", "magenta", "blue")
x <- 100:1000
plot(MaxSalary ~ Score, data=salarygov,
    xlim=c(100, 1000), ylim=c(1000, 10000),
    pch=c(2, 16)[fdom + 1], col=c("black", "cyan")[fdom + 1])

mods <- rq(MaxSalary ~ bs(Score, 5), tau=c(.1, .5, .9),
           data=salarygov[!fdom, ])
mods

Call:
rq(formula = MaxSalary ~ bs(Score, 5), tau = c(0.1, 0.5, 0.9),
    data = salarygov[!fdom, ])

Coefficients:
              tau= 0.1 tau= 0.5 tau= 0.9
(Intercept)    1207.00  1507.32  1466.54
bs(Score, 5)1  -100.93  -151.93   437.42
bs(Score, 5)2   779.87   974.06  1300.73
bs(Score, 5)3  2010.56  2255.89  3175.96
bs(Score, 5)4  3724.40  3822.21  5010.02
bs(Score, 5)5  5122.00  6147.10  5733.79

Degrees of freedom: 357 total; 351 residual

predictions <- predict(mods, data.frame(Score=x))
for( j in 1:3) lines(x, predictions[, j], col=cols[j], lty=ltys[j], lwd=2)
legend("topleft", legend=taus, title="Quantile", lty=ltys, lwd=2,
    col=cols, inset=0.01)
legend("bottomright", legend=c("Non-Female-Dominated","Female-Dominated"),
    pch=c(2, 16), inset=0.01, col=c("black", "cyan"))
```

We begin by defining an indicator variable for the female-dominated job classes, and a vector for the τs. We will graph the non-female-dominated classes in black and the female-dominated classes in cyan. The quantile regression is fit using the `rq()` function in the **quantreg** package. Its arguments are similar to those for `lm()` except for a new argument for setting `tau`; the default is `tau=0.5` for L_1 regression, and here we specify three values of τ. The fitted coefficients for the B-splines are then displayed, and although these are not easily interpretable, the important point is that they are different for each value of τ. The `predict()` function returns a matrix with three columns, one for each τ, and we use these values to add fitted regression lines to the graph. We fit the model to the non-female-dominated occupations only, as is common is gender-discrimination studies.

The quantile regressions are of interest here to describe the variation in the relationship between salary and score in the non-female-dominated job classes. Most of the female-dominated classes fall below the median line and many below the 0.1-quantile. For extreme values of `Score` the more extreme quantiles are very poorly estimated, which accounts for the crossing of the median and the 0.9 estimated quantiles for large values of `Score`.

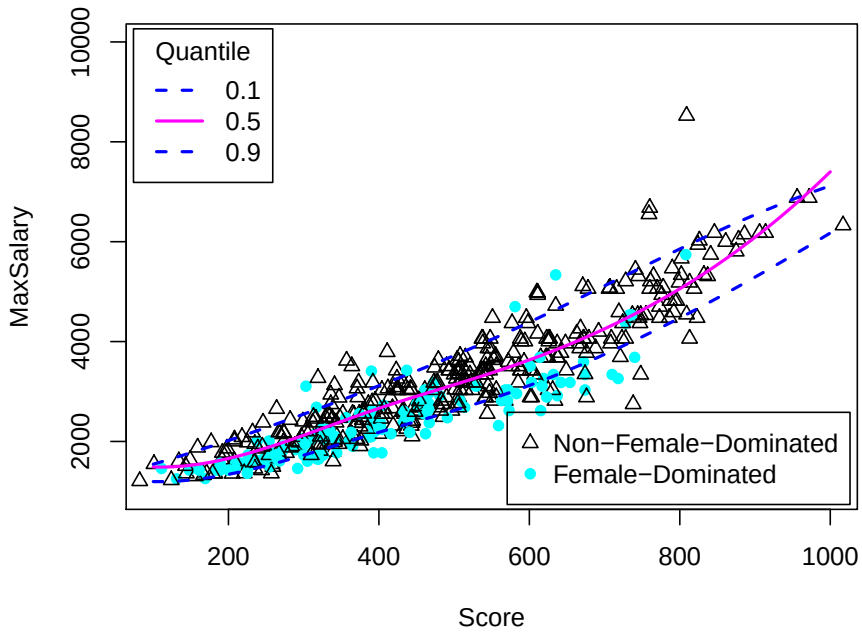

Figure 5: Quantile regressions fit to non-female-dominated job classes.

6.2 Duncan's Data

Quantile regression can also be used for multiple regression. For example, to compute the L_1 regression for Duncan's occupational-prestige data:

```
mod.quant <- rq(prestige ~ income + education, data=Duncan)
summary(mod.quant)
```

```
Call: rq(formula = prestige ~ income + education, data = Duncan)

tau: [1] 0.5

Coefficients:
            coefficients lower bd  upper bd
(Intercept)  -6.40826    -12.49552  -3.60027
income        0.74771      0.47194   0.91169
education     0.45872      0.21948   0.66095
```

The `summary()` method for `"rq"` objects reports 95-percent confidence intervals for the regression coefficients; it is also possible to obtain coefficient standard errors (see `?summary.rq`). The L_1 estimates here are very similar to the M-estimates based on Huber's weight function. Table 2 summarizes the various estimators that we applied to Duncan's regression.

Method	b_0	b_1 (`income`)	b_2 (`education`)
OLS	−6.0647	0.5987	0.5458
OLS removing cases 6 & 16	−6.4090	0.8674	0.3322
Huber M-estimator	−7.1107	0.7014	0.4854
bisquare MM-estimator	−7.3886	0.7825	0.4233
LTS estimator	−7.0145	0.8045	0.4318
L_1 estimator	−6.4083	0.7477	0.4587

Table 2: Various estimators of Duncan's occupational-prestige regression.

7 Complementary Reading and References

Robust regression is described in Fox (2016, Chap. 19). Koenker (2005) provides an extensive treatment of quantile regression. A mathematical treatment of robust regression is given by Huber and Ronchetti (2009). Andersen (2007) provides an introduction to the topic.

References

Andersen, R. (2007). *Modern Methods for Robust Regression*. Sage, Thousand Oaks, CA.

Cook, R. D., Hawkins, D. M., and Weisberg, S. (1992). Comparison of model misspecification diagnostics using residuals from least mean of squares and least median of squares fits. *Journal of the American Statistical Association*, 87(418):pp. 419–424.

Fox, J. (2016). *Applied Regression Analysis and Generalized Linear Models*. Sage, Thousand Oaks CA, third edition.

Fox, J. and Weisberg, S. (2019). *An R Companion to Applied Regression*. Sage, Thousand Oaks, CA, third edition.

Huber, P. and Ronchetti, E. M. (2009). *Robust Statistics*. Wiley, Hoboken NJ, second edition.

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):pp. 73–101.

Koenker, R. (2005). *Quantile regression*. Cambridge University Press, Cambridge.

Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust regression and outlier detection*. Wiley, Hoboken, NJ.

Stefanski, L. (1991). A note on high-breakdown estimators. *Statistics & Probability Letters*, 11(4):353 – 358.