

# Package ‘robustreg’

July 23, 2025

**Version** 0.1-11  
**Date** 2019-05-09  
**Title** Robust Regression Functions  
**Author** Ian M. Johnson <ijca2013@gmail.com>  
**Maintainer** Ian M. Johnson <ijca2013@gmail.com>  
**Depends** R (>= 3.6.0)  
**Description** Linear regression functions using Huber and bisquare psi functions. Optimal weights are calculated using IRLS algorithm.  
**License** GPL (>= 2)  
**Imports** stats (>= 3.6.0), Matrix (>= 1.1.0), Rcpp (>= 0.11.3)  
**LinkingTo** Rcpp, RcppArmadillo  
**NeedsCompilation** yes  
**Repository** CRAN  
**Date/Publication** 2019-05-10 07:50:03 UTC

## Contents

|                         |          |
|-------------------------|----------|
| fit_rcpp . . . . .      | 2        |
| mad_rcpp . . . . .      | 2        |
| median_rcpp . . . . .   | 3        |
| psiBS_rcpp . . . . .    | 3        |
| psiHuber_rcpp . . . . . | 4        |
| robustRegBS . . . . .   | 5        |
| robustRegH . . . . .    | 6        |
| <b>Index</b>            | <b>8</b> |

---

`fit_rcpp`*Predict y from X and b*

---

**Description**

Predict y vector from X design matrix and b vector

**Usage**

```
fit_rcpp(X,b)
```

**Arguments**

|   |                   |
|---|-------------------|
| X | Design matrix     |
| b | Estimates of beta |

**Author(s)**

Ian M. Johnson

**Examples**

```
j <- rep(1, 5)
x1 <- rnorm(5)
x2 <- rnorm(5, 10, 20)
X = as.matrix(data.frame(j, x1, x2))
b <- 1:3
fit_rcpp(X, b)
```

---

`mad_rcpp`*Median Absolute Deviation (MAD)*

---

**Description**

Rcpp fast implementation of median absolute deviation (MAD)

**Usage**

```
mad_rcpp(r,scale_factor = 1.4826)
```

**Arguments**

|              |                  |
|--------------|------------------|
| r            | A numeric vector |
| scale_factor | Scale factor     |

### Author(s)

Ian M. Johnson

### Examples

```
mad(1:100)
```

---

median\_rcpp

*Median*

---

### Description

Rcpp fast implementation of median

### Usage

```
median_rcpp(x)
```

### Arguments

x                      A numeric vector containing the values whose median is to be computed.

### Author(s)

Ian M. Johnson

### Examples

```
median_rcpp(1:100)
```

---

psiBS\_rcpp

*Tukey's Bisquare Psi Function*

---

### Description

Rcpp fast implementation of Tukey's Bisquare psi function

### Usage

```
psiBS_rcpp(r,c)
```

### Arguments

r                      A numeric vector  
c                      Tuning constant

**Author(s)**

Ian M. Johnson

**Examples**

```
## Not run:  
psiBS_rcpp(r,c)  
  
## End(Not run)
```

---

psiHuber\_rcpp

*Huber Psi Function*

---

**Description**

Rcpp fast implementation of Huber's Psi Function

**Usage**

```
psiHuber_rcpp(r,c)
```

**Arguments**

|   |                  |
|---|------------------|
| r | A numeric vector |
| c | Tuning constant  |

**Author(s)**

Ian M. Johnson

**Examples**

```
## Not run:  
psiHuber_rcpp(r,c)  
  
## End(Not run)
```

---

robustRegBS*Robust Fitting of Linear Models using Bisquare Psi Function*

---

**Description**

Using iteratively reweighted least squares (IRLS), the function calculates the optimal weights to perform m-estimator or bounded influence regression. Returns robust beta estimates, mean squared error (MSE) and prints robust ANOVA table.

**Usage**

```
robustRegBS(formula, data, tune=4.685, m=TRUE, max.it=1000, tol=1e-5, anova.table=FALSE)
```

**Arguments**

|             |                                                                                                   |
|-------------|---------------------------------------------------------------------------------------------------|
| formula     | Model                                                                                             |
| data        | A data frame containing the variables in the model.                                               |
| tune        | Tuning Constant. Default value of 4.685 is 95% asymptotically efficient against outliers          |
| m           | If TRUE, calculates m estimates of beta. If FALSE, calculates bounded influence estimates of beta |
| max.it      | Maximum number of iterations to achieve convergence in IRLS algorithm                             |
| tol         | Tolerance level in determining convergence                                                        |
| anova.table | If TRUE, prints robust ANOVA table                                                                |

**Details**

M-estimates of beta should be used when evaluating least squares estimates of beta and diagnostics show outliers. Least squares estimates of beta should be used as starting points to achieve convergence.

Bounded influence estimates of beta should be used when evaluating least squares estimates of beta and diagnostics show large values of the "Hat Matrix" diagonals and outliers.

**Note**

Original package written in 2006

**Author(s)**

Ian M. Johnson

**References**

Tukey,  
Birch, Robust F-Test, 1983

See Also

robustRegH()

Examples

```
data(stackloss)
robustRegBS(stack.loss~Air.Flow+Water.Temp,data=stackloss)

#If X matrix contained large values of H matrix (high influence points)
robustRegBS(stack.loss~Air.Flow+Water.Temp,data=stackloss,m=FALSE)
```

---

|            |                                                                 |
|------------|-----------------------------------------------------------------|
| robustRegH | <i>Robust Fitting of Linear Models using Huber Psi Function</i> |
|------------|-----------------------------------------------------------------|

---

Description

Using iteratively reweighted least squares (IRLS), the function calculates the optimal weights to perform m-estimator or bounded influence regression. Returns robust beta estimates, mean squared error (MSE) and prints robust ANOVA table

Usage

```
robustRegH(formula,data,tune=1.345,m=TRUE,max.it=1000,tol=1e-5,anova.table=FALSE)
```

Arguments

|             |                                                                                                   |
|-------------|---------------------------------------------------------------------------------------------------|
| formula     | Model                                                                                             |
| data        | A data frame containing the variables in the model.                                               |
| tune        | Tuning Constant. Default value of 1.345 is 95% asymptotically efficient against outliers          |
| m           | If TRUE, calculates m estimates of beta. If FALSE, calculates bounded influence estimates of beta |
| max.it      | Maximum number of iterations to achieve convergence in IRLS algorithm                             |
| tol         | Tolerance level in determining convergence                                                        |
| anova.table | If TRUE, prints robust ANOVA table                                                                |

Details

M-estimates of beta should be used when evaluating least squares estimates of beta and diagnostics show outliers. Least squares estimates of beta are used as starting points to achieve convergence.

Bounded influence estimates of beta should be used when evaluating least squares estimates of beta and diagnostics show large values of the "Hat Matrix" diagonals and outliers.

Note

Original package written in 2006

**Author(s)**

Ian M. Johnson

**References**

P. J. Huber (1981) Robust Statistics. Wiley.

Birch (1983) Robust F-Test

**See Also**

`robustRegBS()`

**Examples**

```
data(stackloss)
robustRegH(stack.loss~Air.Flow+Water.Temp,data=stackloss)

#If X matrix contained large values of H matrix (high influence points)
robustRegH(stack.loss~Air.Flow+Water.Temp,data=stackloss,m=FALSE)
```

# Index

## \* regression

robustRegBS, [5](#)

robustRegH, [6](#)

fit\_rcpp, [2](#)

mad\_rcpp, [2](#)

median\_rcpp, [3](#)

psiBS\_rcpp, [3](#)

psiHuber\_rcpp, [4](#)

robustRegBS, [5](#)

robustRegH, [6](#)