

# Package ‘parody’

July 21, 2025

**Title** Parametric And Resistant Outlier DYtection  
**Version** 1.66.1  
**Description** Provide routines for univariate and multivariate outlier detection with a focus on parametric methods, but support for some methods based on resistant statistics.  
**Depends** R ( $\geq 3.5.0$ ), tools, utils  
**Suggests** knitr, BiocStyle, testthat, rmarkdown  
**License** Artistic-2.0  
**VignetteBuilder** knitr  
**biocViews** MultipleComparison  
**git\_url** <https://git.bioconductor.org/packages/parody>  
**git\_branch** RELEASE\_3\_21  
**git\_last\_commit** 8529446  
**git\_last\_commit\_date** 2025-06-09  
**Repository** Bioconductor 3.21  
**Date/Publication** 2025-07-20  
**Author** Vince Carey [aut, cre] (ORCID: <https://orcid.org/0000-0003-4046-0063>)  
**Maintainer** Vince Carey <stvjc@channing.harvard.edu>

## Contents

box.scale . . . . .	2
bushfire . . . . .	2
calout.detect . . . . .	3
mv.calout.detect . . . . .	5
shorth . . . . .	6
tcost . . . . .	7
tukeyor . . . . .	7
<b>Index</b>	<b>9</b>

---

box.scale	<i>calibrated scaling inlier multiplier radius for various outlier detection approaches</i>
-----------	---

---

**Description**

calibrated scaling inlier multiplier radius for various outlier detection approaches

**Usage**

```
box.scale(n, alpha=0.05)
```

**Arguments**

n	n
alpha	alpha

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**Examples**

```
box.scale(20)
```

---

bushfire	<i>satellite data on bushfire scars</i>
----------	---

---

**Description**

satellite data on bushfire scars

**Usage**

```
data(bushfire)
```

**Format**

The format is: num [1:38, 1:5] 111 113 113 110 101 93 92 94 94 100 ...

**Details**

Satellite measurements on five frequency bands corresponding to each of 38 pixels.

**Source**

Maronna RA and Yohai VJ (1995) JASA 90:330.

**Examples**

```
data(bushfire)
mv.calout.detect(bushfire)
```

---

calout.detect	<i>interface to modular calibrated outlier detection system</i>
---------------	---

---

**Description**

Various classical and resistant outlier detection procedures are provided in which the outlier misclassification rate for Gaussian samples is fixed over a range of sample sizes.

**Usage**

```
calout.detect(x, alpha = 0.05, method = c("GESD", "boxplot", "medmad",
      "shorth", "hybrid"), k = ((length(x)%2) * floor(length(x)/2) +
      (1 - (length(x)%2)) * (length(x)/2 - 1)), scaling, ftype,
      location, scale, gen.region = function(x, location, scale,
      scaling, alpha) {
      g <- scaling(length(x), alpha)
      location(x) + c(-1, 1) * g * scale(x)
    })
```

**Arguments**

x	data vector, NAs not allowed
alpha	outlier mislabeling rate for Gaussian samples
method	one of c("GESD", "boxplot", "medmad", "shorth"); the first selects generalized extreme studentized deviate (Rosner, 1983); the second selects calibrated boxplot rules; the third selects the method of Hampel in which the sample median is used for location estimation, and the median absolute deviation is used for scale; and the fourth selects Rousseeuw's rule, with the midpoint of the shortest half sample used as location estimator, and the length of this shortest half sample used as scale estimator.  An important characteristic of the GESD procedure is that the critical values for outlier labeling are calibrated to preserve the overall Type I error rate of the procedure given that there will be k tests, whether or not any outliers are present in the data.
k	for GESD, the prespecified upper limit on the number of outliers suspected in the data; defaults to "half" the sample size.
scaling	for resistant methods, scaling is a sample-size dependent function that tells how many multiples of the scale estimate should be laid off on each side of the location estimate to demarcate the inlier region; see Davies and Gather (1993) for the general formulation. The main contribution of this program consists in the development of scaling functions that "calibrate" outlier detection in Gaussian

samples. The scaling function is assumed to take two arguments,  $n$  and  $\alpha$ , and it should return a real number.

If `method=="boxplot"`, the default value `scaling=box.scale` will confine the probability of erroneous detection of one or more outliers in a pure Gaussian sample to  $\alpha$ . The use of `scaling=function(n,alpha) 1.5` gives the standard boxplot outlier labeling rule.

If `method=="medmad"`, the use of `scaling=hamp.scale.4` will confine the outlier mislabeling rate to  $\alpha$ ; whereas the use of `scaling=function(n,alpha) 5.2` gives Hampel's rule (Davies and Gather, 1993, p. 790).

If `method=="shorth"`, the default value `scaling=shorth.scale` will confine the outlier mislabeling rate to  $\alpha$ .

<code>ftype</code>	The type of "fourth" calculation; the standard definition of the fourth uses $0.5 * \text{floor}((n + 3)/2)$ to obtain the sortile of the fourth value; Hoaglin and Iglewicz (1987) give an "ideal" definition of the fourth which reduces the dependence of boxplot-based outlier detection performance (in small samples) on the quantity $n \bmod 4$ .
<code>location</code>	a function on a vector returning a location estimate
<code>scale</code>	a function on a vector returning a scale estimate
<code>gen.region</code>	a function of $x$ , <code>location</code> , <code>scale</code> , <code>scaling</code> , $\alpha$ that returns the inlier region as a 2-vector

### Value

a list with components `ind` (indices of outliers in the input vector) `val` (values of these components) and `outlier.region`, which is only defined for the resistant methods.

### References

Davies and Gather (1993 JASA), Rousseeuw and Leroy (1988 Stat Neer), Rosner (1983 Technom), Hoaglin and Iglewicz (1987 JASA), Carey, Walters, Wager and Rosner (1997 Technom)

### Examples

```
lead <- c(83, 70, 62, 55, 56, 57, 57, 58, 59, 50, 51, 52, 52, 52, 54, 54, 45, 46, 48,
         48, 49, 40, 40, 41, 42, 42, 44, 44, 35, 37, 38, 38, 34, 13, 14)

calout.detect(lead,alpha=.05,method="boxplot",ftype="ideal")
calout.detect(lead,alpha=.05,method="GESD",k=5)
calout.detect(lead,alpha=.05,method="medmad",scaling=hamp.scale.3)
calout.detect(lead,alpha=.05,method="shorth")
```

---

mv.calout.detect	<i>calibrated multivariate outlier detection</i>
------------------	--

---

## Description

interface to a parametric multivariate outlier detection algorithm

## Usage

```
mv.calout.detect(x, k = min(floor((nrow(x) - 1)/2), 100), Ci = C.unstr,
  lamfun = lams.unstr, alpha = 0.05, method = c("parametric",
    "rocke", "kosinski.raw", "kosinski.exch")[1], ...)
```

## Arguments

x	data matrix
k	upper bound on number of outliers; defaults to just less than half the sample size
Ci	function computing Ci, the covariance determinant ratio excluding row i. At present, sole option is C.unstr (Caroni and Prescott 1992 Appl Stat).
lamfun	function computing lambda, the critical values for Ci
alpha	false outlier labeling rate
method	string identifying algorithm to use
...	reserved for future use

## Details

bushfire is a dataset distributed by Kosinski to illustrate his method.

## Value

a list with components	
inds	indices of outlying rows
vals	values of outlying rows
k	input parameter k
alpha	input parameter alpha

## Author(s)

VJ Carey

## References

C. Caroni and P. Prescott, Journal of the Royal Statistical Society. Series C (Applied Statistics), Vol. 41, No. 2 (1992), pp. 355-364

**Examples**

```
data(tcost)
mv.calout.detect(tcost)
data(bushfire)
mv.calout.detect(bushfire)
```

---

shorth	<i>one-dimensional MVE (min. vol. ellipsoid)</i>
--------	--

---

**Description**

generalized length of shortest-half sample

**Usage**

```
shorth(x, Alpha=0.5)
```

**Arguments**

x	data vector, no NAs
Alpha	minimum fraction of data to be covered by scale estimator. if Alpha == 0.5, the shorth is calculated

**Value**

a list, say L, with components

shorth	a 2-vector with endpoints of the shortest Alpha-sample
length.shorth	see previous return component L\$shorth[2]-L\$shorth[1]
midpt.shorth	mean(L[["shorth"]])
meanshorth	mean of values in the shorth, studied by Andrews et al (1972) as a location estimator
correction.parity.dep	correction factor to be applied to achieve approximate unbiasedness and diminish small-sample parity dependence; L[["shorth"]] * L[["correction"]] is approximately unbiased for the Gaussian standard deviation, for $0 < \text{Alpha} < 1$ .
bias.correction.gau.5	correction factor to be applied along with correction.parity.dep when Alpha = .5; empirically derived bias correction useful for $10 < N < 2000$ and possibly beyond. To use, divide: (L[["shorth"]] * L[["correction"]] / L[["bias.corr"]]) is approximately unbiased for Gaussian standard deviation, when Alpha=.5.
Alpha	coverage fraction used

**References**

Rousseeuw and Leroy, Stat Neer (1988), Gruebel, Ann Stat (1988)

---

tcost	<i>Data on milk transportation costs, from Johnson and Wichern, Applied Multivariate Statistical Analysis, 3rd edition</i>
-------	--

---

**Description**

Multivariate data on milk transportation costs

**Usage**

```
data(tcost)
```

**Format**

The format is: num [1:36, 1:3] 16.44 7.19 9.92 4.24 11.2 ...  
 - attr(\*, "dimnames")=List of 2  
 ..\$: chr [1:36] "1" "2" "3" "4" ...  
 ..\$: chr [1:3] "fuel" "repair" "capital"

**Details**

Extract from Johnson and Wichern example dataset on milk transportation.

**Source**

Johnson and Wichern, Applied Multivariate Statistical Analysis, 3rd edition, p263

**Examples**

```
data(tcost)
mv.calout.detect(tcost)
```

---

tukeyor	<i>calibrated outlier region based on various algorithms</i>
---------	--

---

**Description**

calibrated outlier region based on various algorithms

**Usage**

```
tukeyor(x, alpha=0.05, g=box.scale(length(x), alpha = alpha), ftype="ideal")
```

**Arguments**

x	x
alpha	alpha
g	g
ftype	ftype

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**Examples**

```
data(tcost)
apply(tcost, 2, tukeyor)
```



# Index

## \* **datasets**

bushfire, [2](#)

tcost, [7](#)

## \* **models**

box.scale, [2](#)

calout.detect, [3](#)

mv.calout.detect, [5](#)

tukeyor, [7](#)

## \* **robust**

shorth, [6](#)

box.scale, [2](#)

bushfire, [2](#)

calout.detect, [3](#)

hamp.scale.3 (box.scale), [2](#)

hamp.scale.4 (box.scale), [2](#)

hampor (tukeyor), [7](#)

mv.calout.detect, [5](#)

rouor (tukeyor), [7](#)

shorth, [6](#)

shorth.scale (box.scale), [2](#)

tcost, [7](#)

tukeyor, [7](#)