Package 'HistogramTools'

October 12, 2022

Type Package

Title Utility Functions for R Histograms

Version 0.3.2

Date 2015-07-28

Maintainer Murray Stokely <murray@stokely.org>

Copyright Copyright 2011 Google, Inc.

Imports Hmisc, ash, stringr

Suggests RUnit, emdist, gdata

Enhances RProtoBuf

Description Provides a number of utility functions useful for manipulating large histograms. This includes methods to trim, subset, merge buckets, merge histograms, convert to CDF, and calculate information loss due to binning. It also provides a protocol buffer representations of the default R histogram class to allow histograms over large data sets to be computed and manipulated in a MapReduce environment.

URL https://r-forge.r-project.org/projects/histogramtools/

Classification/ACM G.3

License Apache License 2.0

NeedsCompilation no

Author Murray Stokely [aut, cre],
Tim Hesterberg [ctb]

Repository CRAN

Date/Publication 2015-07-29 20:48:41

R topics documented:

`histogramtools-package`
HistogramTools package

Description

This package provides a number of utility functions for manipulating R's native histogram objects. The functions are focused on operations that are particularly useful when dealing with large numbers of histograms with identical buckets, such as those produced from distributed MapReduce computations. This package also provides a '`HistogramTools.HistogramState`' protocol buffer representation of the default R histogram class to allow histograms to be very concisely serialized and shared with other systems.

Details

See `library(help=HistogramTools)` for version number, dates, dependencies, and a complete list of functions.

Index (possibly out of date):

```
AddHistograms         Aggregate histogram objects that have identical breaks.
MergeBuckets          Merge adjacent buckets of a histogram.
ApproxQuantile        Approximate the quantiles of the underlying distribution.
ApproxMean            Approximate the mean of the underlying distribution.
Count                 Count of all samples in a histogram.
HistToEcdf            Approximate the ECDF of the underlying distribution.
SubsetHistogram       Subset a histogram by removing some of the buckets.
TrimHistogram         Remove empty buckets from the tails of a histogram.
ScaleHistogram        Scale histogram bucket counts by a numeric value.
PreBinnedHistogram    Generate a histogram from pre-binned data.
AshFromHist           Compute Average Shifted Histogram from a histogram.
KSDCC            Compute maximal KS-statistic of CDFs constructed from histogram.
```

```
EMDCC            Compute maximal Earth Mover's Distance of CDFs constructed from histogram.
PlotKSDCC             Plot the KSDCC metric and a CDF from the histogram.
PlotEMDCC             Plot the EMDCC metric and a CDF from the histogram.
PlotLog2ByteEcdf    Plot the CDF from a histogram with log2 scaled byte boundaries.
PlotLogTimeDurationEcdf Plot the CDF from a histogram with log scaled time duration boundaries.
PlotRelativeFrequency Plot a relative frequency histogram.
ReadHistogramsFromDtraceOutputFile  Read a list of Histograms from the output of the DTrace tool.
minkowski.dist        Compute the Minkowski difference between two histograms.
intersect.dist     Compute the histogram intersection distance between two histograms.
kl.divergence      Compute the Kullback-Leibler divergence between two histograms.
jeffrey.divergence    Compute the Jeffrey divergence between two histograms.
```

Author(s)

Murray Stokely <mstokely@google.com>

See Also

hist

Examples

```
if(require(RProtoBuf)) {
library(HistogramTools)

tmp.hist <- hist(c(1,2,4,43,20,33,1,1,3), plot=FALSE)
# The default R serialization takes a fair number of bytes
length(serialize(tmp.hist, NULL))

# Convert to a protocol buffer representation.
hist.msg <- as.Message(tmp.hist)

# Which has an ASCII representation like this:
cat(as.character(hist.msg))

# Or can be serialized and shared with other tools much more
# succinctly than R's built-in serialization format.
length(hist.msg$serialize(NULL))

# And since this isn't even compressed, we can reduce it further
# with in-memory compression:
length(memCompress(hist.msg$serialize(NULL)))

# If we read in the raw.bytes from another tool
raw.bytes <- hist.msg$serialize(NULL)

# We can parse the raw bytes as a protocol buffer
new.hist.proto <- P("HistogramTools.HistogramState")$read(raw.bytes)
new.hist.proto

# Then convert back to a native R histogram.
new.hist <- as.histogram(new.hist.proto)
```

```
# The new histogram and the old are identical except for xname
}
```

AddHistograms *Aggregate histograms that have identical breaks.*

Description

Aggregate histogram objects that have identical breaks.

Usage

```
AddHistograms(..., x=list(...), main=.NewHistogramName(x))
.NewHistogramName(x)
```

Arguments

...	Histogram objects (created by hist).
x	A list of histogram objects. If x is supplied then the ... are ignored.
main	A title to add to the aggregated/merged histogram. By default, if two histograms are provided a title will be created that includes the names of the original histograms. If more histograms are provided the title will simply include the number of aggregated histograms.

Details

This function adds the buckets of the provided histograms to return a single aggregated histogram.

.NewHistogramName is a utility that takes the list of histogram objects to be aggregated and returns a name for the new merged histogram. It is normally hidden, but can be viewed using HistogramTools:::.NewHistogramName.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package and hist.

Examples

```
hist.1 <- hist(c(1,2,3,4), plot=FALSE)
hist.2 <- hist(c(1,2,2,4), plot=FALSE)
hist.sum <- AddHistograms(hist.1, hist.2)
hist.3 <- hist(c(1,2,2,4), plot=FALSE)
hist.sum <- AddHistograms(hist.1, hist.2, hist.3)
```

`as.histogram` *Convert histogram protocol buffers to histogram objects*

Description

This package provides a number of utility functions useful for manipulating large histograms. It provides a 'HistogramTools.HistogramState' protocol buffer representation of the default R histogram class to allow histograms to be very concisely serialized and shared with other systems in a distributed MapReduce environment. It also includes a number of utility functions for manipulating large histograms.

Usage

```
  ## S3 method for class 'Message'
as.histogram(x, ...)
```

Arguments

x	An RProtoBuf Message of type "HistogramTools.HistogramState" to convert to a histogram object.
...	Not used.

Details

`as.histogram` reads the provided Protocol Buffer message and extracts the buckets and counts to populate into the standard R histogram class which can be plotted.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

`histogramtools-package`, `as.Message`, and `RProtoBuf`.

Examples

```
if(require(RProtoBuf)) {
library(HistogramTools)

tmp.hist <- hist(c(1,2,4,43,20,33,1,1,3), plot=FALSE)
# The default R serialization takes a fair number of bytes
length(serialize(tmp.hist, NULL))

# Convert to a protocol buffer representation.
hist.msg <- as.Message(tmp.hist)

# Which has an ASCII representation like this:
```

```
cat(as.character(hist.msg))

# Or can be serialized and shared with other tools much more
# succinctly than R's built-in serialization format.
length(hist.msg$serialize(NULL))

# And since this isn't even compressed, we can reduce it further
# with in-memory compression:
length(memCompress(hist.msg$serialize(NULL)))

# If we read in the raw.bytes from another tool
raw.bytes <- hist.msg$serialize(NULL)

# We can parse the raw bytes as a protocol buffer
new.hist.proto <- P("HistogramTools.HistogramState")$read(raw.bytes)
new.hist.proto

# Then convert back to a native R histogram.
new.hist <- as.histogram(new.hist.proto)

# The new histogram and the old are identical except for xname
}
```

`as.Message` *Convert R histograms to Protocol Buffer representation*

Description

This package provides a number of utility functions useful for manipulating large histograms. It provides a 'HistogramTools.HistogramState' protocol buffer representation of the default R histogram class to allow histograms to be very concisely serialized and shared with other systems in a distributed MapReduce environment. It also includes a number of utility functions for manipulating large histograms.

Usage

```
  ## S3 method for class 'histogram'
as.Message(x)
```

Arguments

x	A histogram object (created by `hist`).

Details

`as.Message` converts the provided histogram object into a protocol buffer representation that can be compactly serialized and shared with tools written in other languages.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package, as.histogram, and RProtoBuf.

Examples

```
if (require(RProtoBuf)) {
library(HistogramTools)

tmp.hist <- hist(c(1,2,4,43,20,33,1,1,3), plot=FALSE)
# The default R serialization takes a fair number of bytes
length(serialize(tmp.hist, NULL))

# Convert to a protocol buffer representation.
hist.msg <- as.Message(tmp.hist)

# Which has an ASCII representation like this:
cat(as.character(hist.msg))

# Or can be serialized and shared with other tools much more
# succinctly than R's built-in serialization format.
length(hist.msg$serialize(NULL))

# And since this isn't even compressed, we can reduce it further
# with in-memory compression:
length(memCompress(hist.msg$serialize(NULL)))

# If we read in the raw.bytes from another tool
raw.bytes <- hist.msg$serialize(NULL)

# We can parse the raw bytes as a protocol buffer
new.hist.proto <- P("HistogramTools.HistogramState")$read(raw.bytes)
new.hist.proto

# Then convert back to a native R histogram.
new.hist <- as.histogram(new.hist.proto)

# The new histogram and the old are identical except for xname
}
```

ASH *Average Shifted Histograms From a Histogram.*

Description

Computes a univariate average shifted histogram (polynomial kernel) given a single input histogram.

Usage

```
HistToASH(h, m=5, kopt=c(2,2))
```

Arguments

h	A histogram object (created by hist) representing a pre-binned dataset.
m	optional integer smoothing parameter, passed to ash1().
kopt	vector of length 2 specifying the kernel, passed to ash1().

Details

This function takes a histogram and uses the counts as the input to the ash1() function in the ash package to compute the average shifted histogram.

Author(s)

Murray Stokely <mstokely@google.com>

References

Scott, David W. *Multivariate density estimation: theory, practice, and visualization.* Vol. 383. Wiley. com, 2009.

See Also

histogramtools-package, ash1, and hist.

Examples

```
x <- runif(1000, min=0, max=100)
h <- hist(x, breaks=0:100, plot=FALSE)
plot(h, freq=FALSE)

# Superimpose the Average Shifted Histogram on top of the original.
lines(HistToASH(h), col="red")
```

---

dtracehistograms *Read Histograms from text DTrace output file.*

Description

Parses the text output of the DTrace command to convert the ASCII representation of aggregate distributions into R histogram objects.

Usage

```
ReadHistogramsFromDtraceOutputFile(filename)
```

Arguments

filename	A character vector naming a file that is the output of dtrace with aggregate distribution statistics in it.

Details

The DTrace dynamic tracing framework allows users to trace applications and computer operating systems. One of its common outputs is aggregates of a distribution (for example, read request sizes) that are output as a histogram. This function takes the text output from a dtrace command and looks for text distribution representations that can be parsed into R histogram objects.

Value

A list of histogram objects representing the histograms present in the Dtrace output file.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package, hist.

Examples

```
## Not run:
system("dtrace -n 'syscall::read:return { @[execname] = quantize(arg0);}> /tmp/dtraceoutput'",
  intern=TRUE)
system.readsize.hists <- ReadHistogramsFromDtraceOutputFile("/tmp/dtraceoutput")
plot(system.readsize.hists[[1]])

## End(Not run)
```

HistogramDistance *Histogram Distance Measures*

Description

The pairs of bins in two histograms with the same bucket boundaries are compared to compute dissimilarity measures.

Usage

```
minkowski.dist(h1, h2, p)
intersect.dist(h1, h2)
kl.divergence(h1, h2)
jeffrey.divergence(h1, h2)
```

Arguments

h1,h2	"histogram" objects (created by hist) representing a binned dataset.
p	Order of the Minkowski distance between two histograms to compute.

Details

The minkowski.dist function computes the Minkowski distance of order p between two histograms. p=1 is the Manhattan distance and p=2 is the Euclidean distance.

The intersect.dist function computes the intersection distance of two histograms, as defined in Swain and Ballard 1991, p15. If histograms h1 and h2 do not contain the same total of counts, then this metric will not be symmetric.

The kl.divergence function computes the Kullback-Leibler divergence between two histograms.

The jeffrey.divergence function computes the Jeffrey divergence between two histograms.

Author(s)

Murray Stokely <mstokely@google.com>

References

Rubner, Yossi, Carlo Tomasi, and Leonidas J. Guibas. "The earth mover's distance as a metric for image retrieval." International Journal of Computer Vision 40.2 (2000): 99-121.

Puzicha, Jan, Thomas Hofmann, and Joachim M. Buhmann. "Non-parametric similarity measures for unsupervised texture segmentation and image retrieval." Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on. IEEE, 1997.

Swain, Michael J., and Dana H. Ballard. "Color indexing." International journal of computer vision 7.1 (1991): 11-32.

See Also

histogramtools-package, ecdf, and hist.

Examples

```
h1 <- hist(runif(100), plot=FALSE)
h2 <- hist(runif(100), plot=FALSE)

minkowski.dist(h1, h2, 1)
minkowski.dist(h1, h2, 2)
minkowski.dist(h1, h2, 3)
intersect.dist(h1, h2)
kl.divergence(h1, h2)
jeffrey.divergence(h1, h2)
```

HistToEcdf *Empirical Cumulative Distribution Function From a Histogram.*

Description

Computes an approximate empirical cumulative distribution function of a data set given a binned histogram representation of that dataset.

Usage

```
HistToEcdf(h, method="constant", f=0, inverse=FALSE)
```

Arguments

h	A histogram object (created by hist) representing a pre-binned dataset on which we'd like to calculate an ECDF.
method	specifies the interpolation method to be used in call to approxfun(). Choices are "linear" or "constant".
f	if method="constant" a number between 0 and 1 inclusive, indicating a compromise between left- and right-continuous step functions. See ?approxfun
inverse	if TRUE return the inverse Ecdf.

Details

This function approximates the e.c.d.f. (empirical cumulative distribution function) of a data set given a binned histogram representation of that data set.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package, ecdf, approxfun, and hist.

Examples

```
h <- hist(runif(100), plot=FALSE)
plot(HistToEcdf(h))
```

`InformationLoss` *Information Loss Metrics for Histograms*

Description

Computes a metric between 0 and 1 of the amount of information lost about the underlying distribution of data for a given histogram.

Usage

```
KSDCC(h)
EMDCC(h)
PlotKSDCC(h, arrow.size.scale=1, main=paste("KSDCC =", KSDCC(h)), ...)
PlotEMDCC(h, main=paste("EMDCC =", EMDCC(h)), ...)
```

Arguments

`h`	A `"histogram"` object (created by `hist`) representing a pre-binned dataset on which we'd like to calculate the information loss due to binning.
`arrow.size.scale`	specifies a size scaling factor for the arrow illustrating the point of Kolmogorov-Smirnov distance between the two e.c.d.fs
`main`	if 'method="constant"' a number between 0 and 1 inclusive, indicating a compromise between left- and right-continuous step functions. See ?approxfun
`...`	Any other arguments to pass to `plot`

Details

The `KSDCC` (Kolmogorov-Smirnov Distance of the Cumulative Curves) function provides the Kolmogorov-Smirnov distance between the empirical distribution functions of the smallest and largest datasets that could be represented by the binned data in the provided histogram. This quantity is also called the Maximum Displacement of the Cumulative Curves (MDCC) in the computer science performance evaluation community (see references).

The `EMDCC` (Earth Mover's Distance of the Cumulative Curves) function is like the Kolmogorov-Smirnov statistic, but uses an integral to capture the difference across all points of the curve rather than just the maximum difference. This is also known as Mallows distance, or Wasserstein distance with $p=1$.

The `PlotKSDCC` and `PlotEMDCC` functions take a histogram and generate a plot showing a geometric representation of the information loss metrics for the provided histogram.

Author(s)

Murray Stokely <mstokely@google.com>

References

Douceur, John R., and William J. Bolosky. "A large-scale study of file-system contents." *ACM SIGMETRICS Performance Evaluation Review* **27.1** (1999): 59-70.

See Also

histogramtools-package, ecdf, and hist.

Examples

```
x <- rexp(1000)
h <- hist(x, breaks=c(0,1,2,3,4,8,16,32), plot=FALSE)
KSDCC(h)

# For small enough data sets we can construct the two extreme data sets
# that can be constructed from a histogram.  One assuming every data point
# is on the left boundary of its bucket, and another assuming every data
# point is on the right boundary of its bucket.  Our KSDCC metric for
# histograms is equivalent to the ks.test statistics for these two
# extreme data sets.

x.min <- rep(head(h$breaks, -1), h$counts)
x.max <- rep(tail(h$breaks, -1), h$counts)
ks.test(x.min, x.max, exact=FALSE)

## Not run:
PlotKSDCC(h)

## End(Not run)

EMDCC(h)
## Not run:
PlotEMDCC(h)

## End(Not run)
```

IntersectHistograms *Intersect Histograms*

Description

Takes two histograms with identical bucket boundaries and returns a histogram of the intersection of the two. Each bucket of the returned histogram contains the minimum of the equivalent buckets in the two provided histograms.

Usage

```
IntersectHistograms(h1, h2)
```

Arguments

h1,h2	"histogram" objects (created by hist) representing a dataset summarized by binning.

Author(s)

Murray Stokely <mstokely@google.com>

References

Swain, Michael J., and Dana H. Ballard. "Color indexing." International journal of computer vision 7.1 (1991): 11-32.

See Also

histogramtools-package, hist.

Examples

```
h1 <- hist(runif(100), plot=FALSE)
h2 <- hist(runif(100), plot=FALSE)
plot(IntersectHistograms(h1, h2))
```

MergeBuckets *Merge adjacent buckets in a histogram to create a new histogram.*

Description

Merge adjacent buckets in a histogram and return a new histogram.

Usage

```
MergeBuckets(x, adj.buckets=NULL, breaks=NULL, FUN=sum)
```

Arguments

x	A histogram object (created by hist).
adj.buckets	The number of adjacent buckets to merge together.
breaks	If adj.buckets is equal to NULL, then this argument either specifies a vector giving the breakpoints between cells, or a single number giving the total number of cells in the new histogram. The vector of new buckets must have the same range as the original break list. If a single number is provided it must be less than length(x$breaks).
FUN	The user defined function that should be run to merge the counts of adjacent buckets in the histogram.

Details

Many data analysis pipelines write out histogram protocol buffers with thousands of buckets so as to be applicable in a wide range of contexts. This function provides a way to transform the histogram into one with fewer buckets.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package and hist.

Examples

```
hist.1 <- hist(c(1,2,3), breaks=c(0,1,2,3,4,5,6,7,8,9), plot=FALSE)
hist.2 <- MergeBuckets(hist.1, adj.buckets=2)

hist.1
hist.2
```

PlotLog2ByteEcdf *Plot Binned Histogram and ECDF Data.*

Description

Produces aesthetically pleasing ECDF plots for two common classes of non-equiwidth histograms. Specifically, (1) histograms with power of two bucket boundaries commonly used in computer science for measuring resource usage, and (2) histograms with log-scaled time duration buckets with a range from 1 second to 10 years.

Usage

```
PlotLog2ByteEcdf(x, xlab="Bytes (log)",
                 ylab="Cumulative Fraction", with.grid=TRUE, ...)
PlotLogTimeDurationEcdf(x, with.grid=TRUE,
                        xlab="Age (log)",
                        ylab="Cumulative Fraction",
                        cex.lab=1.6,
                        cex.axis=1.6, ...)
```

Arguments

x	A "histogram" object (created by hist) representing a pre-binned dataset or an "ecdf" object (created by ecdf or HistToEcdf).
xlab	x-axis label for the Ecdf plot.
ylab	y-axis label for the Ecdf plot.

with.grid	Logical. If TRUE, draw faint grid lines on the ECDF plot.
cex.lab	Graphical parameters for plot and axes.
cex.axis	Graphical parameters for plot and axes.
...	Additional parameters are passed to plot().

Details

The PlotLog2ByteEcdf function takes a "histogram" or "ecdf" which has power of 2 bucket boundaries representing bytes and creates an Ecdf plot.

The PlotLogTimeDurationEcdf function takes a "histogram" or "ecdf" with exponential bucket boundaries representing seconds of age or duration and creates an Ecdf plot.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package, hist. ecdf.

Examples

```
filename <- system.file("unitTests/data/buildkernel-readsize-dtrace.txt",
                        package="HistogramTools")
dtrace.hists <- ReadHistogramsFromDtraceOutputFile(filename)
x <- SubsetHistogram(dtrace.hists[["TOTAL"]], minbreak=1)
PlotLog2ByteEcdf(x, cex.lab=1.4)

x <- rexp(100000)
x <- x*(86400*300)/diff(range(x))

n <- as.integer(1+log2(max(x)))

h <- hist(x, breaks=c(0, unique(as.integer(2^seq(from=0, to=n, by=.25)))))
PlotLogTimeDurationEcdf(h)
```

PlotRelativeFrequency *Plot Relative Frequency Histogram*

Description

Produces a relative frequency histogram.

Usage

```
PlotRelativeFrequency(x, ylab="Relative Frequency", ...)
```

Arguments

x	A "histogram" object (created by hist) representing a pre-binned dataset.
ylab	y-axis label for the plot.
...	Additional parameters are passed to plot().

Details

The default plot.histogram function supports Frequency or Density plots, but does not provide a way to produce a relative frequency histogram. This function plots this type of histogram.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package, hist, plot.histogram.

Examples

```
x <- runif(100)
h <- hist(x, plot=FALSE)
PlotRelativeFrequency(h)
```

PreBinnedHistogram *PreBinnedHistogram*

Description

Takes a set of already binned data represented by a vector of $n+1$ breaks and a vector of n counts and returns a normal R histogram object.

Usage

```
PreBinnedHistogram(breaks, counts, xname="")
```

Arguments

breaks	A numeric vector of n+1 breakpoints for the histogram.
counts	A numeric vector of n counts for each bucket of the histogram.
xname	A character string with the name for this histogram.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package, hist.

quantile *Histogram Approximate Quantiles.*

Description

Approximate the quantiles of the underlying distribution for which only a histogram is available.

Usage

```
ApproxQuantile(x, probs, ...)
ApproxMean(x)
Count(x)
```

Arguments

x	A histogram object (created by hist).
probs	Numeric vector of probabilities with values in [0,1].
...	Any other arguments to pass to wtd.quantile.

Details

Many data analysis pipelines write out histogram protocol buffers with thousands of buckets so as to be applicable in a wide range of contexts. This function provides a way to transform the histogram into approximations of the quantile, median, mean, etc of the underlying distribution. Count(x) returns the number of observations in the histogram.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package and hist.

Examples

```
x <- hist(c(1,2,3), breaks=c(0,1,2,3,4,5,6,7,8,9), plot=FALSE)
Count(x)
ApproxMean(x)
ApproxQuantile(x, .5)
ApproxQuantile(x, c(.05, .95))
```

ScaleHistogram *Scale Histogram Counts*

Description

Scales the counts of the provided histograms by the provided factor and returns a new histogram.

Usage

```
ScaleHistogram(x, factor)
```

Arguments

x	A "histogram" object (created by hist) representing a dataset summarized by binning.
factor	A number value to scale the bucket counts by. Defaults to 1/Count(x) to normalize the sum of counts of the histogram to 1.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package, hist.

Examples

```
x <- runif(100)
h <- hist(x, plot=FALSE)
plot(ScaleHistogram(h, 10))
```

SubsetHistogram *Subset a histogram by removing some of the buckets.*

Description

SubsetHistogram creates a histogram by zooming in on a portion of a larger histogram and discarding tail buckets.

Usage

```
SubsetHistogram(x, minbreak=NULL, maxbreak=NULL)
```

Arguments

x	A histogram object (created by hist).
minbreak	If non-NULL, specifies a new minimum breakpoint for the returned histogram. Must be one of the existing breakpoints of x.
maxbreak	If non-NULL, specifies a new maximum breakpoint for the returned histogram. Must be one of the existing breakpoints of x.

Details

This function provides a way to "zoom-in" on a histogram by setting new minimum and maximum breakpoints and returning a histogram of only the interior part of the distribution. At least one of minbreak or maxbreak should be set, otherwise the original histogram is returned unmodified.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package and hist.

Examples

```
hist.1 <- hist(c(1,2,3), breaks=c(0,1,2,3,4,5,6,7,8,9), plot=FALSE)
hist.2 <- SubsetHistogram(hist.1, maxbreak=6)

hist.1
hist.2
```

---

trimsparse *Trim the tails of a sparse histogram.*

Description

Removes empty consecutive buckets at the tails of a histogram.

Usage

```
TrimHistogram(x, left=TRUE, right=TRUE)
```

Arguments

x	A histogram object (created by hist).
left	If TRUE, consecutive buckets in the left tail of the histogram without any elements will be removed from the returned histogram.
right	If TRUE, consecutive buckets in the right tail of the histogram without any elements will be removed from the returned histogram.

Details

Many data analysis pipelines write out histogram protocol buffers with thousands of buckets so as to be applicable in a wide range of contexts. This function provides a way to transform the histogram into one with fewer buckets by removing sparseness in the tails.

Author(s)

Murray Stokely <mstokely@google.com>

See Also

histogramtools-package and hist.

Examples

```
hist.1 <- hist(c(1,2,3), breaks=c(0,1,2,3,4,5,6,7,8,9), plot=FALSE)
length(hist.1$counts)
sum(hist.1$counts)
hist.trimmed <- TrimHistogram(hist.1)
length(hist.trimmed$counts)
sum(hist.trimmed$counts)
```

Index