

# Package ‘swfscMisc’

March 15, 2021

**Type** Package

**Title** Miscellaneous Functions for Southwest Fisheries Science Center

**Description** Collection of conversion, analytical, geodesic, mapping, and plotting functions. Used to support packages and code written by researchers at the Southwest Fisheries Science Center of the National Oceanic and Atmospheric Administration.

**Version** 1.4.3

**URL** <https://github.com/EricArcher/swfscMisc>

**BugReports** <https://github.com/EricArcher/swfscMisc/issues>

**Depends** R (>= 4.0.0)

**Imports** graphics, grDevices, dplyr, ggplot2, ggrepel, kkn, mapdata, magrittr, tibble, maps, rlang, sf, spatstat.geom, stats, tidy, utils

**License** GPL (>= 2)

**RoxygenNote** 7.1.1

**NeedsCompilation** no

**Author** Eric Archer [aut, cre]

**Maintainer** Eric Archer <eric.archer@noaa.gov>

**Repository** CRAN

**Date/Publication** 2021-03-15 14:00:02 UTC

## R topics documented:

affin.prop	3
autoUnits	4
bearing	5
box.area	5
braces	6
catSpatInterp	7
central.quantile	9

circle.polygon . . . . .	10
color.name . . . . .	11
convert.angle . . . . .	12
convert.distance . . . . .	13
copy.tri . . . . .	13
crossing.point . . . . .	14
das.map . . . . .	15
das.merge.acoustic . . . . .	16
das.read . . . . .	16
das.spp.freq . . . . .	17
datum . . . . .	17
destination . . . . .	18
distance . . . . .	19
diversity . . . . .	20
fisher.method.p . . . . .	21
geometric.mean . . . . .	21
ggBiplot . . . . .	22
harmonic.mean . . . . .	22
isBetween . . . . .	23
lab.wid . . . . .	24
lat.lon.axes . . . . .	24
month2Season . . . . .	25
na.count . . . . .	25
odds . . . . .	26
one.arg . . . . .	27
plotAssignments . . . . .	27
pVal . . . . .	28
round . . . . .	29
row.col.page.fit . . . . .	30
sample.map . . . . .	31
scatterdens . . . . .	32
setupClusters . . . . .	33
sex.symbols . . . . .	34
swfscMisc . . . . .	34
transparent . . . . .	35
uniform.test . . . . .	35
which.nearest . . . . .	36
zero.pad . . . . .	37

---

`affin.prop`*Affinity Propagation*

---

**Description**

Runs the Affinity Propagation clustering algorithm of Frey and Dueck, 2007.

**Usage**

```
affin.prop(  
  sim.mat,  
  num.iter = 100,  
  stable.iter = 10,  
  shared.pref = "min",  
  lambda = 0.5  
)
```

**Arguments**

<code>sim.mat</code>	a similarity matrix between individuals to be clustered.
<code>num.iter</code>	maximum number of iterations to attempt.
<code>stable.iter</code>	number of sequential iterations for which consistent clustering is considered acceptable.
<code>shared.pref</code>	type of shared preference to use. Can be one of "min", "median", or a numeric value.
<code>lambda</code>	damping factor.

**Value**

A matrix with one row per sample in 'sim.mat' and one column for each iteration. Values in columns indicate cluster assignment (arbitrary numbers) for each sample.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**References**

Frey, B.J., and D. Dueck. 2007. Clustering by passing messages between data points. *Science* 315:972-976

## Examples

```
data(iris)

# Take 75 random iris rows for example
iris <- iris[sample(1:nrow(iris), 75), ]
iris <- droplevels(iris)

iris.sim <- -dist(iris[, -5])

iris.affin <- affin.prop(iris.sim, stable.iter = 5)
table(iris$Species, iris.affin[, ncol(iris.affin)])
```

---

autoUnits

*Auto Time Interval Units*

---

## Description

Convert time interval units to natural values based on magnitude of difference.

## Usage

```
autoUnits(x)
```

## Arguments

x an object inheriting from class `difftime`

## Author(s)

Eric Archer <eric.archer@noaa.gov>

## Examples

```
autoUnits(as.difftime("0:3:35"))
autoUnits(as.difftime("15:3:35"))
autoUnits(ISOdate(2000, 5, 1) - ISOdate(2000, 4, 20))
```

---

bearing	<i>Calculate Bearing Between Two Positions</i>
---------	--

---

**Description**

Calculates the bearing between two points, given each point's latitude and longitude coordinates

**Usage**

```
bearing(lat1, lon1, lat2, lon2)
```

**Arguments**

lat1, lon1	numeric. The latitude and longitude of the starting coordinate in decimal degrees.
lat2, lon2	numeric. The latitude and longitude of the ending coordinate in decimal degrees.

**Value**

vector with initial and final bearings.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
# What is the bearing from San Diego, CA to Honolulu, HI?  
bearing(32.87, -117.25, 21.35, -157.98)
```

---

box.area	<i>Area of a Box</i>
----------	----------------------

---

**Description**

Calculate the area of a square on the earth.

**Usage**

```
box.area(lat, lon, edge, units = "nm")
```

**Arguments**

lat, lon	The latitude and longitude of the lower right corner of the box in decimal degrees.
edge	The length of one side of the square in decimal degrees.
units	units of distance. Can be "km" (kilometers), "nm" (nautical miles), or "mi" (statute miles).

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
#What is the area of a 5 degree grid off of San Diego, CA?
box.area(32.87, -117.25, edge = 1, units = "nm")
box.area(32.87, -117.25, edge = 1, units = "km")
box.area(32.87, -117.25, edge = 1, units = "mi")
```

---

braces

*Braces*


---

**Description**

Adds curly braces to a plot.

**Usage**

```
braces(
  xfrom,
  xto,
  yfrom,
  yto,
  radius = 1,
  col = par("fg"),
  lty = par("lty"),
  lwd = par("lwd")
)
```

**Arguments**

xfrom, xto, yfrom, yto	start and end points of braces. Direction of brace determined by from and to arguments.
radius	radius of curve in brace.
col, lty, lwd	color, line type, and line width of braces. See <a href="#">par</a> for more details.

**Note**

Orientation of brace is either horizontal or vertical, with axis along largest range of x or y in plotting units.

**Author(s)**

Tim Gerrodette <tim.gerrodette@noaa.gov>

**Examples**

```
plot(x = c(0, 1), y = c(0, 1000), type = "n", xlab = "", ylab = "")
braces(xfrom = 0.2, xto = 0.8, yfrom = c(400, 600), yto = c(300, 700))
plot(x = c(0, 100), y = c(0, 17), type = "n", xlab = "x", ylab = "y")
text(10, 16, "radius =")
for (i in 1:8) {
  braces(xfrom = 10 * i + 10, xto = 10 * i + 18, yfrom = 1,
        yto = 15, radius = i / 4, lwd = 2)
  text(10 * i + 12, 16, round(i / 4, 2))
}
plot(c(0, 100), c(0, 17), type = "n", xlab = "x", ylab = "y")
braces(30, 80, 13, 11, 1)

plot(c(0, 100), c(0, 17), type = "n", xlab = "x", ylab = "y")
braces(c(20, 80, 30), c(10,75,40), 1, 15, radius = c(0.2, 0.5, 0.1),
      lwd = c(1, 2, 3), col = 1:2, lty = 1)

plot(c(0, 100), c(0, 17), type = "n")
braces(20, 80, 7, 5, 1)
braces(20, 80, 13, 15, 1)
```

**Description**

Create a raster of probability of categorical values interpolated across a 2-dimensional space given a set of points where each is assigned to one of several classes.

**Usage**

```
catSpatInterp(
  data,
  x.col = "x",
  y.col = "y",
  group.col = "group",
  num.grid = 100,
  knn = 10,
```

```

    hull.buffer = 0.1,
    num.cores = 1,
    num.batches = NULL
  )

```

### Arguments

**data** matrix or data.frame containing points and grouping designation.

**x.col, y.col, group.col** numbers or characters identifying which columns in data are the x and y values and grouping designation.

**num.grid** number of grid cells for k-nearest neighbor interpolation.

**knn** number of nearest neighbors to consider for interpolation.

**hull.buffer** percent increase of convex hull to use as spatial area to interpolate over.

**num.cores** number of cores to distribute interpolations over.

**num.batches** number of batches to divide grid cell interpolations into.

### Value

A list containing a raster and points of buffered convex hull.

### Author(s)

Eric Archer <eric.archer@noaa.gov>

### References

Adapted from code originally presented in a blog post on Categorical Spatial Interpolation by Timo Grossenbacher <https://timogrossenbacher.ch/2018/03/categorical-spatial-interpolation-with-r/>

### Examples

```

## Not run:
iris.mds <- stats::cmdscale(dist(iris[, 1:4]), k = 2)
mds.df <- setNames(
  cbind(iris.mds, data.frame(iris$Species)),
  c("dim1", "dim2", "Species")
)

result <- catSpatInterp(
  mds.df, x.col = "dim1", y.col = "dim2", group.col = "Species",
  num.grid = 300, knn = 20, hull.buffer = 0.05,
  num.cores = 5, num.batches = NULL
)

library(ggplot2)
ggplot(mapping = aes(dim1, dim2)) +
  geom_raster(
    aes(fill = Species, alpha = prob),

```



```
    data = result$raster
  ) +
  geom_polygon(data = result$hull.poly, fill = NA, col = "black") +
  geom_hline(yintercept = 0, col = "white") +
  geom_vline(xintercept = 0, col = "white") +
  geom_point(
    aes(fill = Species),
    data = mds.df,
    col = "black",
    shape = 21,
    size = 4
  ) +
  theme(
    axis.ticks = element_blank(),
    axis.text = element_blank(),
    axis.title = element_blank(),
    legend.position = "top",
    panel.grid = element_blank(),
    panel.background = element_blank()
  )

## End(Not run)
```

---

central.quantile	<i>Central Quantile</i>
------------------	-------------------------

---

### Description

Upper and lower values of central quantile

### Usage

```
central.quantile(x, pct = 0.95)
```

### Arguments

x	numeric vector.
pct	central percentile desired.

### Value

a two element vector giving the lower and upper quantiles.

### Author(s)

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x <- runif(1000)
central.quantile(x)
central.quantile(x, pct = 0.75)
```

---

circle.polygon	<i>Circle Polygon (on Earth)</i>
----------------	----------------------------------

---

**Description**

Creates a circular polygon (optionally on the earth) centered at a given point with a constant radius.

**Usage**

```
circle.polygon(
  x,
  y,
  radius,
  brng.limits = 0,
  sides = 1,
  by.length = TRUE,
  units = "nm",
  ellipsoid = datum(),
  dist.method = "lawofcosines",
  destination.type = "ellipsoid",
  poly.type = "cart.earth"
)
```

**Arguments**

x, y	number specifying the coordinates of the center of the circle in decimal degrees. If poly.type is "simple.earth" or "complex.earth", this will be longitude and latitude respectively.
radius	radius of sphere.
brng.limits	number, or vector of two numbers. If one value is given, it is used as the starting bearing in degrees for the first point of the circle. If a vector of two values is given, then they are used as the start and end bearings of arc.
sides	number that represents either length of sides or number of sides, as specified by the 'by.length' argument.
by.length	logical. If TRUE, then sides is the length of sides, if FALSE, then sides is number of sides.
units	character for units of distance: Can be "km" (kilometers), "nm" (nautical miles), "mi" (statute miles).
ellipsoid	ellipsoid model parameters as returned from a call to <a href="#">datum</a> .

- `dist.method` character specifying method for calculating distance for `type = "cart.earth"`. See method argument of [distance](#) for more information.
- `destination.type` character specifying type of surface for `type = "gc.earth"`. See `type` argument of [destination](#) for more information.
- `poly.type` character specifying the type of polygon calculation to use. Can be one of "cartesian" using basic cartesian coordinates, "cart.earth" for a simple polygon on the earth's surface treating latitude and longitude as cartesian coordinates, or "gc.earth" for a more precise calculation keeping a constant great-circle radius.

**Value**

A matrix representing the desired circle polygon centered at `lat`, `lon` of `radius`.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```

cart.earth <- circle.polygon(-117.24, 32.86, 40, poly.type = "cart.earth")

lat.range <- c(32, 34)
lon.range <- c(-118.5, -116)

op <- par(mar = c(3, 5, 5, 5) + 0.1, oma = c(1, 1, 1, 1))

maps::map("mapdata::worldHires", fill = TRUE, col = "wheat3", xlim = lon.range, ylim = lat.range)
points(-117.24, 32.86, pch = 19, col = "red")
polygon(cart.earth, border = "red", lwd = 3)
lat.lon.axes(n = 3)
box(lwd = 2)
mtext("poly.type = 'cart.earth'", line = 3)

par(op)

```

---

color.name

*Color Name*

---

**Description**

Return the name of a color listed given the number.

**Usage**

```
color.name(i)
```

**Arguments**

i integer specifying color .

**Value**

character value of 'i' color.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

---

convert.angle      *Angle Conversion*

---

**Description**

Converts angles between radians and degrees.

**Usage**

```
convert.angle(x, from = c("degrees", "radians"), to = c("radians", "degrees"))
```

**Arguments**

x numeric. The angle to be converted.

from, to character. Units to convert from and to. Can be "radians" or "degrees" or any partial match (case-sensitive).

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
convert.angle(45, "deg", "rad")
convert.angle(4.5, "r", "d")
```

---

convert.distance      *Distance Conversion*

---

**Description**

Convert distances between kilometers, nautical miles, and statute miles.

**Usage**

```
convert.distance(x, from = c("nm", "km", "mi"), to = c("km", "nm", "mi"))
```

**Arguments**

x                      numeric. The distance to be converted.  
from, to                character. Units to convert from and to. Can be "km" (kilometers), "nm" (nautical miles), or "mi" (statute miles), or any partial match thereof (case sensitive).

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

---

copy.tri                *Copy Matrix Triangles*

---

**Description**

Copy between lower left and upper right triangles of a matrix.

**Usage**

```
copy.tri(x, from = "lower")
```

**Arguments**

x                      a matrix.  
from                    triangle to copy from. Can be "lower" or "upper".

**Value**

a matrix.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x <- matrix(1:9, nrow = 3)
print(x)
copy.tri(x)
```

---

`crossing.point`*Crossing Point*

---

**Description**

Return point where two lines cross

**Usage**

```
crossing.point(l1, l2)
```

**Arguments**

`l1, l2` matrices representing two lines, where first two columns are x and y values respectively

**Value**

a data.frame of x and y values of points where lines cross

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x <- 1:100
line1 <- cbind(x, 3 + 3 * x)
line2 <- cbind(x, 10 - 3 * x)
plot(line1[, 1], line1[, 2], type = "l", col = "red")
lines(line2[, 1], line2[, 2], col = "blue")
cr.pt <- crossing.point(line1, line2)
print(cr.pt)
```

---

`das.map`*Map DAS*

---

## Description

Map sightings, effort, and beaufort from DAS file

## Usage

```
das.map(  
  x,  
  main,  
  spp = NULL,  
  spp.col = rainbow(length(spp), end = max(1, length(spp) - 2)/length(spp)),  
  lat.range = NULL,  
  lon.range = NULL,  
  n.ticks = 5,  
  spp.legend.loc = "topleft",  
  effort.legend.loc = "topright"  
)
```

## Arguments

<code>x</code>	filename of a DAS file or a <code>data.frame</code> from <a href="#">das.read</a> .
<code>main</code>	main title for plot.
<code>spp</code>	vector of species codes to plot sightings of. If <code>NULL</code> , only effort will be shown.
<code>spp.col</code>	vector of colors for each species.
<code>lat.range</code> , <code>lon.range</code>	latitude and longitude range of map.
<code>n.ticks</code>	number of desired tick marks for latitude and longitude.
<code>spp.legend.loc</code> , <code>effort.legend.loc</code>	the location of the species and effort legends. Can be a single keyword as specified in <a href="#">legend</a> , or a two-element numeric vector giving the x (longitude) and y (latitude) coordinates.

## Author(s)

Eric Archer <[eric.archer@noaa.gov](mailto:eric.archer@noaa.gov)>

---

das.merge.acoustic      *Merge DAS Data Into Acoustic Detections*

---

**Description**

Fill in sighting information for acoustic detections from DAS file.

**Usage**

```
das.merge.acoustic(acoust.file, das.file, out.file)
```

**Arguments**

acoust.file      filename of a .csv file of acoustic detections exported from Access database.  
das.file          filename of a DAS file.  
out.file          desired output .csv filename.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

---

das.read                  *Read DAS File*

---

**Description**

Reads a DAS file into a data.frame where each line is data for a specific event.

**Usage**

```
das.read(file)
```

**Arguments**

file                  filename of a DAS file.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>



---

das.spp.freq	<i>Frequency of species from DAS file</i>
--------------	---

---

**Description**

Returns number of sightings of each species code in a DAS file.

**Usage**

```
das.spp.freq(x)
```

**Arguments**

x filename of a DAS file or a data.frame from [das.read](#).

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

---

datum	<i>Datum</i>
-------	--------------

---

**Description**

Return parameters specifying ellipsoid datum model.

**Usage**

```
datum(model = c("wgs84", "grs80", "airy", "international", "clarke", "grs67"))
```

**Arguments**

model character, specifying which model to use for ellipsoid model. Options are: "wgs84", "grs80", "airy", "international", "clarke", "grs67", or partial matches thereof (case-sensitive).

**Value**

vector of a, b, and f parameters.

**Note**

Model parameters are based on distances in km.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

---

destination                      *Destination on Sphere or Ellipsoid*

---

### Description

Calculates latitude and longitude of the destination along a sphere or ellipsoid.

### Usage

```
destination(  
  lat,  
  lon,  
  brng,  
  distance,  
  units = c("nm", "km", "mi"),  
  ellipsoid = datum(),  
  radius = convert.distance(6371, "km", "nm"),  
  type = c("ellipsoid", "sphere", "vincenty")  
)
```

### Arguments

lat, lon	numeric. The latitude and longitude of the coordinate in decimal degrees.
brng	numeric. The bearing, ranging from 0 to 360 degrees.
distance	numeric. The distance travelled, in units specified by <code>units</code> .
units	units of distance. Can be "km" (kilometers), "nm" (nautical miles), or "mi" (statute miles), or any partial match thereof (case sensitive).
ellipsoid	ellipsoid model parameters as returned from a call to <code>datum</code> .
radius	numeric. Define the radius for <code>type = "sphere"</code> . In units of <code>units</code> .
type	Character defining type of surface. Can be "sphere", "ellipsoid", "vincenty", or partial match thereof (case-sensitive).

### Value

latitude and longitude of destination.

### Author(s)

Eric Archer <eric.archer@noaa.gov>

### References

Ellipsoid code adapted from JavaScript by [Larry Bogan](#).  
Vincenty code adapted from JavaScript by [Chris Veness](#).  
Vincenty, T. 1975. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. [Survey Review 22\(176\):88-93](#).

**Examples**

```

destination(32.87, -117.25, 262, 4174, units = "km", type = "sphere")
destination(32.87, -117.25, 262, 4174, units = "km", type = "ellipsoid")
destination(32.87, -117.25, 262, 4174, units = "km", type = "vincenty")

```

---

distance	<i>Distance Between Coordinates</i>
----------	-------------------------------------

---

**Description**

Calculates the distance between two coordinates using the Law of Cosines, Haversine, or Vincenty methods.

**Usage**

```

distance(
  lat1,
  lon1,
  lat2,
  lon2,
  radius = convert.distance(6371, "km", "nm"),
  units = c("nm", "km", "mi"),
  ellipsoid = datum(),
  iter.limit = 20,
  method = c("lawofcosines", "haversine", "vincenty")
)

```

**Arguments**

lat1, lon1, lat2, lon2	The latitude and longitude of the first and second points in decimal degrees.
radius	radius of sphere.
units	units of distance. Can be "km" (kilometers), "nm" (nautical miles), or "mi" (statute miles), or any partial match thereof (case sensitive).
ellipsoid	ellipsoid model parameters as returned from a call to <a href="#">datum</a> .
iter.limit	An integer value defining the limit of iterations for Vincenty method.
method	Character defining the distance method to use. Can be "lawofcosines", "haversine", "vincenty", or any partial match thereof (case sensitive).

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

## References

Code adapted from JavaScript by [Chris Veness](#)  
Vincenty, T. 1975. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. [Survey Review 22\(176\):88-93](#).

## Examples

```
# What is the distance from San Diego, CA to Honolulu, HI?  
distance(32.87, -117.25, 21.35, -157.98, method = "lawofcosines")  
distance(32.87, -117.25, 21.35, -157.98, method = "haversine")  
distance(32.87, -117.25, 21.35, -157.98, method = "vincenty")
```

---

diversity

*Unbiased Estimate of Diversity*

---

## Description

Calculate unbiased estimate of diversity for a vector of items

## Usage

```
diversity(x)
```

## Arguments

x                    character or numeric vector or factor

## Author(s)

Eric Archer <[eric.archer@noaa.gov](mailto:eric.archer@noaa.gov)>

## Examples

```
x <- sample(1:5, 100, replace = TRUE)  
diversity(x)
```

---

fisher.method.p	<i>Fisher's Method p-value</i>
-----------------	--------------------------------

---

**Description**

Calculate Fisher's method p-value to summarize a vector of p-values based on a chi-squared distribution.

**Usage**

```
fisher.method.p(p.vals)
```

**Arguments**

p.vals            vector of p-values.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

---

geometric.mean	<i>Geometric Mean</i>
----------------	-----------------------

---

**Description**

Calculates the geometric mean of a vector.

**Usage**

```
geometric.mean(x)
```

**Arguments**

x                a numeric vector.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x <- rlnorm(100)
mean(x)
median(x)
geometric.mean(x)
```

ggBiplot

*ggBiplot*

---

**Description**

Plot a biplot of a Principal Components Analysis using ggplot2.

**Usage**

```
ggBiplot(pca, x = 1, y = 2, mult.fac = 0.8, arrow.size = 1.5, label.size = 6)
```

**Arguments**

<code>pca</code>	result from a call to <a href="#">princomp</a> .
<code>x, y</code>	the number or column names of the components to plot.
<code>mult.fac</code>	multiplier factor for lengths of arrows from 0:1.
<code>arrow.size</code>	thickness of arrow lines.
<code>label.size</code>	size of labels.

**Value**

the ggplot2 object is invisibly returned.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
pc.cr <- princomp(USArrests, cor = TRUE)
ggBiplot(pc.cr)
```

---

harmonic.mean*Harmonic Mean*

---

**Description**

Calculate the harmonic mean of a set of numbers.

**Usage**

```
harmonic.mean(x, na.rm = FALSE)
```

**Arguments**

x	a numeric vector.
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x <- rlnorm(100)
mean(x)
median(x)
harmonic.mean(x)
```

---

isBetween

*Between*

---

**Description**

Is a numeric value between two other values?

**Usage**

```
isBetween(x, a, b = NULL, include.ends = FALSE, na.convert = TRUE)
```

**Arguments**

x	vector of numeric values to check.
a, b	numeric values describing range.
include.ends	logical. Should test include a and b? Is test > and < or >= and <= ?
na.convert	logical. If TRUE and result of test is NA because either x, a, or b is NA, return FALSE, otherwise return NA.

**Details**

Order of a and b does not matter. If b is NULL the range will be taken from values in a.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

---

lab.wid	<i>Label Width</i>
---------	--------------------

---

**Description**

Calculate width of labels for plots.

**Usage**

```
lab.wid(labels)
```

**Arguments**

labels            vector of labels to be used on plots

---

lat.lon.axes	<i>Latitude and Longitude axes</i>
--------------	------------------------------------

---

**Description**

Add latitude and longitude axes to a map.

**Usage**

```
lat.lon.axes(n = 5, lon.n = n, lat.n = n)
```

**Arguments**

n, lon.n, lat.n the number of tick marks desired. Can be specified separately for longitude (lon.n) or latitude (lat.n). See [pretty](#) for more details.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>



---

month2Season	<i>Convert Months to Seasons</i>
--------------	----------------------------------

---

**Description**

Convert numeric month to season: Winter = Dec-Feb, Spring = Mar-May, Summer = Jun-Aug, Fall = Sep-Nov

**Usage**

```
month2Season(x)
```

**Arguments**

x                    a vector of months from 1:12

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
months <- sample(1:12, 10, rep = TRUE)
months
month2Season(months)
```

---

na.count	<i>Count NAs</i>
----------	------------------

---

**Description**

Counts NAs in an object.

**Usage**

```
na.count(x)
```

**Arguments**

x                    a vector, data.frame, or matrix.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x <- sample(c(1:10, NA), 30, replace = TRUE)
na.count(x)
x.df <- do.call(data.frame, lapply(1:4, function(i) sample(c(1:10, NA), 30, replace = TRUE)))
colnames(x.df) <- paste("X", 1:4, sep = "")
na.count(x.df)
```

odds

*Odds Conversion***Description**

odds	converts probability to odds
logOdds	converts odds to log-odds
invOdds	converts odds to probability
invLogOdds	converts log-odds to odds

**Usage**

```
odds(x)
```

```
logOdds(x)
```

```
invOdds(x)
```

```
invLogOdds(x)
```

**Arguments**

x a numeric vector of probabilities (0 to 1), odds (0 to Inf), or log.odds (-Inf to Inf).

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x <- sort(runif(10))
odds.df <- data.frame(x = x, odds = odds(x), logOdds = logOdds(x))
odds.df
invOdds(odds.df$odds)
invLogOdds(odds.df$logOdds)
```

---

`one.arg`*One Argument*

---

**Description**

Does the function have just one argument?

**Usage**

```
one.arg(f)
```

**Arguments**

`f` a function.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
one.arg(mean)
one.arg(one.arg)
```

---

`plotAssignments`*Plot assignment distributions*

---

**Description**

Plot individual assignment probability distributions.

**Usage**

```
plotAssignments(
  probs,
  orig,
  type = NULL,
  ylab = NULL,
  freq.sep.line = TRUE,
  plot = TRUE
)
```

**Arguments**

probs	matrix or data.frame of individual assignment probabilities. Each column represents probability of assignment to that group and rows sum to one.
orig	vector of original group assignments
type	either area for stacked continuous area plot or bar for discrete stacked bar chart. The latter is preferred for small numbers of cases. If not specified, a bar chart will be used if all classes have $\leq 30$ cases.
ylab	label for y-axis
freq.sep.line	put frequency of original group on second line in facet label? If FALSE, labels are single line. If NULL frequencies will not be included in labels.
plot	display the plot?

**Value**

the ggplot object is invisibly returned.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
n <- 40
probs <- abs(c(rnorm(n, 80, 10), rnorm(n, 20, 10)))
probs <- (probs - min(probs)) / max(probs)
probs <- cbind(probs, 1 - probs)
colnames(probs) <- NULL
orig <- rep(c("Group.1", "Group.2"), each = n)

plotAssignments(probs, orig)

n <- 15
probs <- abs(c(rnorm(n, 80, 10), rnorm(n, 20, 10)))
probs <- (probs - min(probs)) / max(probs)
probs <- cbind(probs, 1 - probs)
colnames(probs) <- NULL
orig <- rep(c("Group.1", "Group.2"), each = n)

plotAssignments(probs, orig)
```

---

pVal

*Permutation Test P-value*

---

**Description**

Calculate the p-value for a permutation test.

**Usage**

```
pVal(obs, null.dist)
```

**Arguments**

```
obs          observed value.  
null.dist    vector of values from permutation null distribution.
```

**Author(s)**

```
Eric Archer <eric.archer@noaa.gov>
```

**Examples**

```
null.dist <- rnorm(1000)  
obs <- rnorm(1, mean = 1)  
  
plot(density(null.dist), xlim = range(c(obs, null.dist)), main = "")  
abline(v = obs)  
print(obs)  
pVal(obs, null.dist)
```

---

round

*Rounding Numbers for Data Frames*

---

**Description**

Rounds numeric columns in data.frames

**Usage**

```
## S3 method for class 'data.frame'  
ceiling(x)  
  
## S3 method for class 'data.frame'  
floor(x)  
  
## S3 method for class 'data.frame'  
trunc(x, ...)  
  
## S3 method for class 'data.frame'  
round(x, digits = 0)  
  
## S3 method for class 'data.frame'  
signif(x, digits = 6)
```

**Arguments**

x	a data.frame with numeric columns.
...	arguments to be passed to methods.
digits	integer indicating the number of decimal places (round) or significant digits (signif) to be used. See <a href="#">round</a> for more details.

**Details**

Takes a data.frame and returns a data.frame with the specified function applied to each numeric column.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**See Also**

[Round](#)

**Examples**

```
data(mtcars)
round(mtcars, 0)
signif(mtcars, 2)
```

---

row.col.page.fit      *Number of Rows and Columns on Page*

---

**Description**

Return the number of rows and columns for n that best fits on a page of size width x height.

**Usage**

```
row.col.page.fit(n, width = 8.5, height = 11)
```

**Arguments**

n	number of items (e.g., plots) to fit on page.
width, height	dimensions of page.

**Value**

A vector listing the number of rows and columns to use.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
# 9 frames on US letter paper
row.col.page.fit(9)

# 9 frames on a square
row.col.page.fit(9, width = 10, height = 10)
```

---

sample.map

*Plot Samples on World Map*

---

**Description**

Plot a set of samples on a world map.

**Usage**

```
sample.map(
  lon,
  lat,
  lon.range = NULL,
  lat.range = NULL,
  main = NULL,
  pch = 19,
  pt.cex = 1,
  col = "black",
  bg = col,
  n = 5,
  lon.n = n,
  lat.n = n
)
```

**Arguments**

lon, lat	vectors giving the longitude and latitude of points to plot.
lon.range, lat.range	vectors giving the minimum and maximum longitude and latitude of the map. If the first value in lon.range is greater than the second value, then a Pacific-centric map ( <a href="#">world2Hires</a> ) is used and continents will not be filled in.
main	main title for the plot.
pch	point symbol to use.
pt.cex	point size to use.

col point color.  
 bg background color of point.  
 n, lon.n, lat.n the number of tick marks desired. Can be specified separately for longitude (lon.n) or latitude (lat.n). See [pretty](#) for more details.

**Value**

original [par](#) settings for mar and oma.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**See Also**

[lat.lon.axes](#)

**Examples**

```
# Some random points around San Diego, CA
lat <- runif(30, 32.5, 33.3)
lon <- runif(30, -118.3, -117.5)
lat.range <- c(32.4, 33.6)
lon.range <- c(-118.6, -117)
sample.map(lat, lon, lat.range, lon.range)

# Some random points in the Pacific
lat <- runif(30, 20, 50)
lon <- c(runif(15, 150, 180), runif(15, -180, -120))
lat.range <- c(10, 75)
lon.range <- c(110, -110)
sample.map(lat, lon, lat.range, lon.range)
```

---

 scatterdens

*Scatter Plot with Density Margins*


---

**Description**

Produce a scatter plot with a histogram or density plot in the margins

**Usage**

```
scatterdens(x, y, dens.frac = 1/5, ...)
```

```
scatterhist(x, y, xlab = "", ylab = "", dens.frac = 1/5, ...)
```



**Arguments**

`x, y` vectors of points to plot.  
`dens.frac` fraction of screen to be taken up by density plots on margins.  
`...` Arguments to be passed to [plot](#).  
`xlab, ylab` labels for x and y axes.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**References**

Original code by [Ken Kleiman](#)

**Examples**

```
x <- rnorm(100)
y <- rlnorm(100)
op <- par(ask = TRUE)
scatterdens(x, y, xlab = "x", ylab = "y")
par(op)
```

---

 setupClusters

*Setup Clusters*


---

**Description**

Setup parallel clusters for different operating systems.

**Usage**

```
setupClusters(num.cores = 1, max.cores = NULL)
```

**Arguments**

`num.cores` number of cores for multithreading. If NULL, the number used is set to the value of `parallel::detectCores() - 1`.  
`max.cores` maximum number of cores to use.

**Value**

an object of class `c("SOCKcluster", "cluster")`.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

---

sex.symbols	<i>Sex Symbols</i>
-------------	--------------------

---

**Description**

Plots male and female symbols on current plot.

**Usage**

```
sex.symbols(x, y, sex = 1, col = par("fg"), lwd = par("lwd"), cex = 1)
```

**Arguments**

x, y	the x and y coordinates on the current plot.
sex	a numeric vector containing the values 1 (male) or 2 (female). If of length one, then value is recycled for all symbols.
col, lwd, cex	color, line width, and character expansion for each point. lwd and col are recycled as necessary to cover all points. See <a href="#">par</a> for more details.

**Author(s)**

Tim Gerrodette <tim.gerrodette@noaa.gov>

**Examples**

```
x <- runif(20, 0, 10)
y <- runif(20, 0, 200)
plot(x, y, type = "n")
sex.symbols(x, y, sex = 1:2, cex = 1.5, lwd = c(1.5, 4), col = c("blue", "red"))
```

---

swfscMisc	swfscMisc <i>package</i>
-----------	--------------------------

---

**Description**

SWFSC Miscellaneous Functions

---

transparent	<i>Transparent Colors</i>
-------------	---------------------------

---

**Description**

Return transparent form of a named color.

**Usage**

```
transparent(col, percent = 50)
```

**Arguments**

col	vector of colors as name, hexadecimal, or positive integer (see <a href="#">col2rgb</a> ).
percent	percent of transparency (0 = solid, 100 = transparent).

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
pct <- seq(0, 100, by = 10)
plot(pct, pct, bg = transparent("red", pct), pch = 21, cex = 4, xlab = "X", ylab = "Y")
```

---

uniform.test	<i>Uniform Distribution Test</i>
--------------	----------------------------------

---

**Description**

Tests whether a histogram is significantly different from a uniform distribution.

**Usage**

```
uniform.test(hist.output, B = NULL)
```

**Arguments**

hist.output	output from a call to hist.
B	number of replicates for chi-squared permutation.

**Value**

result of chi-squared test.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x.unif <- runif(100)
uniform.test(hist(x.unif), B = 1000)
x.lnorm <- rlnorm(100)
uniform.test(hist(x.lnorm), B = 1000)
```

---

which.nearest

*Which Nearest*

---

**Description**

Find values of one vector that are nearest to values in another vector.

**Usage**

```
which.nearest(x, y)
```

**Arguments**

x                   vector of values to be compared against.

y                   vector of values to examine relative to x. May be of length 1.

@return For each value in y, returns index of value of x which is nearest to y in absolute value. In the case of ties, the function returns the first index of x. If nearest value is `min(x)` or `max(x)`, a warning is issued. NAs and NaNs in x are ignored; NAs and NaNs in y are returned.

**Author(s)**

Tim Gerrodette <tim.gerrodette@noaa.gov>

**Examples**

```
x <- sort(sample(1:100, 20))
y <- sort(sample(min(x):max(x), 5))
i <- which.nearest(x, y)
x
y
x[i]
```

---

`zero.pad`*Zero Pad Integers*

---

**Description**

Return character representation of integers that are zero-padded to the left so all are the same length.

**Usage**

```
zero.pad(x)
```

**Arguments**

`x` a vector of integers.

**Author(s)**

Eric Archer <eric.archer@noaa.gov>

**Examples**

```
x <- c(0, 1, 3, 4, 10)
zero.pad(x)
x <- c(x, 11, 12, 100, 1000)
zero.pad(x)
```

# Index

affin.prop, 3  
autoUnits, 4

bearing, 5  
box.area, 5  
braces, 6

catSpatInterp, 7  
ceiling (round), 29  
central.quantile, 9  
circle.polygon, 10  
col2rgb, 35  
color.name, 11  
convert.angle, 12  
convert.distance, 13  
copy.tri, 13  
crossing.point, 14

das.map, 15  
das.merge.acoustic, 16  
das.read, 15, 16, 17  
das.spp.freq, 17  
datum, 10, 17, 18, 19  
destination, 11, 18  
difftime, 4  
distance, 11, 19  
diversity, 20

fisher.method.p, 21  
floor (round), 29

geometric.mean, 21  
ggBiplot, 22

harmonic.mean, 22

invLogOdds (odds), 26  
invOdds (odds), 26  
isBetween, 23

lab.wid, 24

lat.lon.axes, 24, 32  
legend, 15  
logOdds (odds), 26

month2Season, 25

na.count, 25

odds, 26  
one.arg, 27

par, 6, 32, 34  
plot, 33  
plotAssignments, 27  
pretty, 24, 32  
princomp, 22  
pVal, 28

Round, 30  
round, 29, 30  
row.col.page.fit, 30

sample.map, 31  
scatterdens, 32  
scatterhist (scatterdens), 32  
setupClusters, 33  
sex.symbols, 34  
signif (round), 29  
swfscMisc, 34

transparent, 35  
trunc (round), 29

uniform.test, 35

which.nearest, 36  
world2Hires, 31

zero.pad, 37