Package 'terra'

March 20, 2020

Type Package

Title Classes and Methods for Spatial Data

Version 0.5-2

Date 2020-03-19

Depends R (>= 3.3.0)

Suggests testthat, rgdal

LinkingTo Rcpp

Imports methods, Rcpp, raster (>= 3.0-12)

SystemRequirements C++11, GDAL (>= 2.0.1), GEOS (>= 3.4.0), PROJ (>= 4.8.0)

Maintainer Robert J. Hijmans <r.hijmans@gmail.com>

Description Classes and methods for spatial data, especially raster data. Methods allow for low-level data manipulation as well as high-level global, local, zonal, and focal computation. The predict and interpolate methods facilitate the use of regression type (interpolation, machine learning) models for spatial prediction. The user-interface is very similar to that of the 'raster' package; but it is simpler and faster. Processing of very large files is supported. See the manual and tutorials on <https://rspatial.org/terra/> to get started.

License GPL (>= 3)

URL https://github.com/rspatial/terra

BugReports https://github.com/rspatial/terra/issues/

LazyLoad yes

NeedsCompilation yes

Author Robert J. Hijmans [cre, aut] (<https://orcid.org/0000-0001-5872-2872>),
Roger Bivand [ctb] (install script, GDAL/GEOS code),
Karl Forner [ctb] (progress bar),
Charles Karney [ctb] (GeographicLib),
Edzer Pebesma [ctb] (GDAL/GEOS code from sf),
Darel Finley [ctb] (polygon fill approach)

Repository CRAN

Date/Publication 2020-03-20 10:20:02 UTC

R topics documented:

terra-package *The "terra" package*

Description

The "terra" package implements classes for spatial data (see SpatRaster-class) and supports handling large raster files that cannot be loaded into memory; local, focal, zonal, and global raster operations; polylygon, line and point to raster conversion; integration with modeling methods to make spatial predictions; and more.

The package is conceived as a replacement of the "raster" package. It has a very similar, but simpler, interface, and it is faster. Like the "raster" package, "terra" provides classes and functions to manipulate geographic (spatial) data in "raster" and "vector". Raster data divides space into cells (rectangles; pixels) of equal size (in units of the coordinate reference system). Such continuous spatial data are also referred to as "grid" data, and be contrasted with discrete "vector" spatial data such as points, lines, polygons. An important difference is that "terra" has a single main class for raster data, "SpatRaster", as opposed to the three classes in the "raster" package (RasterLayer, RasterStack, RasterBrick).

Note the following important differences in function names with the raster package

raster package	terra package
`raster, brick`	`rast`
`stack` (for creating an object from file)	`rast`
`stack` (for combining Raster* objects)	`c`
`rasterFromXYZ`	`rast( , type="xyz")`
`extent`	`ext`
`calc`	`app`
`stackApply`	`tapp`
`reclassify, subs, cut`	`classify`
`cellStats`	`global`
`projectRaster`	`warp, project`
`shapefile`	`vect`
`gridDistance, distanceFromPoints`	`distance`
`drawExtent, drawPoly, drawLine`	`draw`
`nlayers`	`nlyr`
`stackSelect`	`collapse`
`compareRaster`	`compareGeom`
`sampleRegular`	`spatSample`

The package should be particularly useful when using very large datasets that can not be loaded into the computer's memory. Functions will work correctly, because they they process large files in chunks, i.e., they read, compute, and write blocks of data, without loading all values into memory at once.

Below is a list of some of the most important methods grouped by theme. Some of these may not have been implemented yet.

I. Creating SpatRaster objects

SpatRaster objects can be created, from scratch, files, or from objects of other classes, with the following functions:

rast	Create a SpatRaster from scratch, file, or another object
c	Combine SpatRasters (multiple layers) (like raster::stack)
subset	Select layers of a SpatRaster

II. Changing the spatial extent and/or resolution of a SpatRaster

merge	Combine SpatRasters with different extents (but same origin and resolution)
mosaic	Combine SpatRasters with different extents and a function for the values in overlapping areas
crop	Select a geographic subset of a SpatRaster
extend	Enlarge a SpatRaster
trim	Trim a SpatRaster by removing exterior rows and/or columns that only have NAs
aggregate	Combine cells of a SpatRaster to create larger cells
disaggregate	Subdivide cells
warp	Warp values to a SpatRaster with a different origin and/or resolution and/or coordinate reference syst
shift	Adjust the location of SpatRaster
flip	Flip values horizontally or vertically
rotate	Rotate values around the date-line (for lon/lat data)
t	Transpose a SpatRaster

III. Local (cell based) computation

Arith-methods	Arith functions (+, -, *, ^, %%, %/%, /)
Math-methods	Math functions like abs, sqrt, trunc, log, log10, exp, sin, round
Logic-methods	Logic functions (!, &, |)
Summary-methods	Summary functions (mean, max, min, range, prod, sum, any, all)
Compare-methods	Compare functions (==, !=, >, <, <=, >=)
app	Apply a function to cells of a SpatRaster (as in base::apply)
tapp	Apply a function to cells of a SpatRaster by groups of layers (as in base::tapply)
overlay	Computations on multiple SpatRaster objects
cover	First layer covers second layer except where the first layer is NA
mask	Use values from first SpatRaster except where cells of the mask SpatRaster are NA (or another value c
classify	(Re-)classify values
init	Initialize cells with new values
area	Compute area of cells (for longitude/latitude data)
collapse	Select cell values from different layers using an index layer

IV. Zonal and global computation

zonal	Summarize a SpatRaster by zones in another SpatRaster
global	Summarize SpatRaster cell values with a function
unique	Get the unique values in a SpatRaster
freq	Frequency table of SpatRaster cell values
crosstab	Cross-tabulate two SpatRasters
quantile	Quantiles
summary	Summary of the values of a SpatRaster (quartiles and mean)

V. Focal and other spatial contextual computation

focal	Focal (neighborhood; moving window) functions
adjacent	Identify cells that are adjacent to a set of cells of a SpatRaster
boundaries	Detection of boundaries (edges)
distance	Shortest distance to a cell that is not NA or to or from a vector object
direction	Direction (azimuth) to or from cells that are not NA
localFun	Local association (using neighborhoods) functions
clump	Find clumps (patches)
terrain	Compute slope, aspect and other terrain characteristics from elevation data
autocor	Compute global or local Moran or Geary indices of spatial autocorrelation

VI. Model predictions

predict	Predict a non-spatial model to a SpatRaster
interpolate	Predict a spatial model to a SpatRaster

VII. Data type conversion

You can coerce SpatRasters to Raster* objects after loading the raster package, using as, as in as(object,"RasterBrick"), or raster(object) or brick(object) or stack(object)

rast	SpatRaster from matrix and other objects

rasterize	Rasterizing points, lines or polygons
as.points	Create points from a SpatRaster or SpatVector
as.lines	Create points from a SpatRaster or SpatVector
as.polygons	Create polygons from a SpatRaster
as.contour	Contour lines from a SpatRaster

VIII. Accessing values of SpatRaster cells

Apart from the function listed below, you can also use indexing with `[` for cell numbers, and `[[` for row / column number combinations

values	Get all cell values (fails with very large rasters), or a row of values (safer)
as.matrix	Get cell values as a matrix
as.array	Get cell values as an array
extract	Extract cell values from a SpatRaster (e.g., by cell, coordinates, polygon)
sampleRandom	Random sample
spatSample	Regular sample
minmax	Get the minimum and maximum value of the cells of a SpatRaster (if known)
setMinMax	Compute the minimum and maximum value of a SpatRaster if these are not known

IX. Plotting

Maps

plot	Plot a SpatRaster. The main method to create a map
plotRGB	Combine three layers (red, green, blue channels) into a single "real color" image
image	Plot a SpatRaster with the image function
persp	Perspective plot of a SpatRaster
contour	Contour plot or filled-contour plot of a SpatRaster
text	Plot the values of a SpatRaster or SpatVector on top of a map

.

Interacting with a map

zoom	Zoom in to a part of a map
click	Query values of SpatRaster or SpatVector by clicking on a map
select	Select a spatial subset of a SpatRaster or SpatVector
draw	Create a SpatExtent or SpatVector by drawing on a map

.

Other plots

plot	x-y scatter plot of the values of two SpatRaster objects
hist	Histogram of SpatRaster values
barplot	Barplot of a SpatRaster
density	Density plot of SpatRaster values
pairs	Pairs plot for layers in a SpatRaster
boxplot	Box plot of the values of a SpatRaster

X. Getting and setting SpatRaster dimensions

Basic parameters of existing SpatRasters can be obtained, and in most cases changed. If there are values associated with a SpatRaster object (either in memory or via a link to a file) these are lost when you change the number of columns or rows or the resolution. This is not the case when the extent is changed (as the number of columns and rows will not be affected). Similarly, with **projection** you can set the projection, but this does not transform the data (see warp for that).

ncol	The number of columns
nrow	The number of rows
ncell	The number of cells (can not be set directly, only via ncol or nrow)
res	The resolution (x and y)
nlyr	Get or set the number of layers
names	Get or set the layer names
xres	The x resolution (can be set with res)
yres	The y resolution (can be set with res)
xmin	The minimum x coordinate (or longitude)
xmax	The maximum x coordinate (or longitude)
ymin	The minimum y coordinate (or latitude)
ymax	The maximum y coordinate (or latitude)
ext	Get or set the extent (minimum and maximum x and y coordinates (a.k.a. "bounding box")
origin	The origin of a SpatRaster
crs	The coordinate reference system (map projection)
isLonLat	Test if an object has a longitude/latitude coordinate reference system
filename	Filename(s) to which a SpatRaster is linked
compareGeom	Compare the geometry of SpatRasters
NAvalue	Get or set the NA value (for reading from a file)

XI. Computing row, column, cell numbers and coordinates

Cell numbers start at 1 in the upper-left corner. They increase within rows, from left to right, and then row by row from top to bottom. Likewise, row numbers start at 1 at the top of the raster, and column numbers start at 1 at the left side of the raster.

xFromCol	x-coordinates from column numbers
yFromRow	y-coordinates from row numbers
xFromCell	x-coordinates from row numbers
yFromCell	y-coordinates from cell numbers
xyFromCell	x and y coordinates from cell numbers
colFromX	Column numbers from x-coordinates (or longitude)
rowFromY	Row numbers from y-coordinates (or latitude)
rowColFromCell	Row and column numbers from cell numbers

cellFromXY	Cell numbers from x and y coordinates
cellFromRowCol	Cell numbers from row and column numbers
cellFromRowColCombine	Cell numbers from all combinations of row and column numbers
cellsFromExtent	Cell numbers from extent object

XII. Writing files

Basic

values	Assign new values to the cells of a SpatRaster
writeRaster	Write all values of SpatRaster to disk

.

Advanced

blockSize	Get suggested block size for reading and writing
writeStart	Open a file for writing
writeValues	Write some values
writeStop	Close the file after writing

XIII. Manipulation of SpatVector objects

The name in **bold** is the equivalent command in ArcGIS.

c	**append** combine "rbind" vector data of the same (vector) type
erase or "-"	**erase** parts of a polygons
intersect or "*"	**intersect** polygons
union or "+"	**union** polygons
cover	**update** and **identity** a polygons
symdif	**symmetrical difference** of two polygons
aggregate	**dissolve** smaller polygons into larger ones
disaggregate	**explode**: turn polygon parts into separate polygons
crop	**clip** vector data using a rectangle (SpatExtent)
select	**select** - interactively select spatial features
click	**identify** attributes by clicking on a map
merge	**Join table**
extract	spatial queries between SpatVector and SpatRaster objects
as.data.frame	coerce coordinates into a data.frame

XIV. SpatExtent objects

extent	Create a SpatExtent object

intersect	Intersect two SpatExtent objects
union	Combine two SpatExtent objects
Math-methods	round/floor/ceiling of the coordinates of a SpatExtent
align	Align a SpatExtent with a SpatRaster
draw	Create a SpatExtent object by drawing it on top of a map (plot)

XV. Miscellaneous

terraOptions	Show, set, save or get session options
tmpFiles	Show or remove temporary files
canProcessInMemory	Test whether a file can be created in memory
readStart	Open file connections for efficient multi-chunck reading
readStop	Close file connections
inMemory	Are the cell values in memory?
fromDisk	Are the cell values read from a file?

Acknowledgments

This package stands on the shoulders of giants (notably for GDAL, GEOS, GeographicLib)

Author(s)

Except where indicated otherwise, the methods and functions in this package were written by Robert J. Hijmans.

---

adjacent *Adjacent cells*

Description

Identify cells that are adjacent to a set of cells on a raster.

Usage

```
## S4 method for signature 'SpatRaster'
adjacent(x, cells, directions, include, ...)
```

Arguments

x	SpatRaster
cells	vector of cell numbers for which adjacent cells should be found. Cell numbers start with 1 in the upper-left corner and increase from left to right and from top to bottom
directions	the directions in which cells should be connected: "rook" (4 directions), "queen" (8 directions), "16" (knight and one-cell queen moves), or "bishop" to connect cells with one-cell diagonal moves.
include	logical. Should the focal cells be included in the result?
...	additional arguments. None implemented

Value

vector with adjacent cells.

Examples

```
r <- rast(nrows=10, ncols=10)
adjacent(r, cells=c(1, 5, 55), directions="queen")
r <- rast(nrows=10, ncols=10, crs="+proj=utm +zone=1 +datum=WGS84")
adjacent(r, cells=11, directions="rook")
# global lat/lon wraps around
r <- rast(nrows=10, ncols=10, crs="+proj=longlat +datum=WGS84")
adjacent(r, cells=11, directions="rook")
```

---

aggregate *Aggregate raster cells*

Description

Aggregate a SpatRaster to create a new SpatRaster with a lower resolution (larger cells). Aggregation groups rectangular areas to create larger cells. The value for the resulting cells is computed with a user-specified function.

Or aggregate ("dissolve") a SpatVector.

Usage

```
## S4 method for signature 'SpatRaster'
aggregate(x, fact=2, fun="mean", ..., filename="", overwrite=FALSE, wopt=list())

## S4 method for signature 'SpatVector'
aggregate(x, by=NULL, sums=NULL, dissolve=TRUE, vars=NULL, ...)
```

Arguments

x	SpatRaster
fact	positive integer. Aggregation factor expressed as number of cells in each direction (horizontally and vertically). Or two integers (horizontal and vertical aggregation factor) or three integers (when also aggregating over layers)
fun	function used to aggregate values
...	additional arguments passed to fun, such as na.rm=TRUE
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
by	character
sums	list
dissolve	logical
vars	character

Details

Aggregation starts at the upper-left end of a raster. If a division of the number of columns or rows with factor does not return an integer, the extent of the resulting SpatRaster will be somewhat larger then that of the original SpatRaster. For example, if an input SpatRaster has 100 columns, and fact=12, the output SpatRaster will have 9 columns and the maximum x coordinate of the output SpatRaster is also adjusted.

The function fun should take multiple numbers, and return a single number. For example mean, modal, min or max.

It should also accept a na.rm argument (or ignore it as one of the 'dots' arguments).

Value

SpatRaster

See Also

disaggregate

Examples

```
r <- rast()
# aggregated raster, no values
ra <- aggregate(r, fact=10)

values(r) <- runif(ncell(r))
# aggregated raster, max of the values
ra <- aggregate(r, fact=10, fun=max)

# multiple layers
s <- c(r, r*2)
x <- aggregate(s, 2)
```

`align` *Align a SpatExtent with a SpatRaster*

Description

Align an SpatExtent object with a SpatRaster. This can be useful to create a new SpatRaster with the same origin and resolution as an existing SpatRaster. Do not use this to force data to match that really does not match (use e.g. `resample` or (dis)aggregate for this).

Usage

```
## S4 method for signature 'SpatExtent,SpatRaster'
align(x, y, snap="near", ...)
```

Arguments

`x`	SpatExtent
`y`	SpatRaster
`snap`	Character. One of "near", "in", or "out", to determine in which direction the extent should be aligned. To the nearest border, inwards or outwards
`...`	additional arguments. None implemented

Value

SpatExtent

See Also

`ext`, `draw`

Examples

```
r <- rast()
e <- ext(-10.1, 9.9, -20.1, 19.9)
ea <- align(e, r)
e
ext(r)
ea
```

app *Apply a function to the cells of a SpatRaster*

Description

Apply a function to values of a SpatRaster. Similar to apply

Usage

```
## S4 method for signature 'SpatRaster'
app(x, fun, ..., filename="", overwrite=FALSE, wopt=list())
```

Arguments

x	SpatRaster object
fun	function
...	additional arguments for fun
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster

Value

SpatRaster

See Also

overlay, math

Examples

```
r <- rast(ncols=10, nrows=10)
values(r) <- 1:ncell(r)
x <- c(r, r, r)
s <- app(x, fun=sum)
```

area *Area and perimeter*

Description

Compute area of polygons or and the surfacer area or raster cells. Computing the surface area of raster cells is only relevant for longitude/latitude rasters. For planar data, the cell area is constant, can be computed with `prod(res(x))`. For polygons, the area of each geometry is returned.

The perimeter method works only on SpatVector objects, and computes the length of lines or the perimeter of polygons.

Usage

```
## S4 method for signature 'SpatRaster'
area(x, filename="", overwrite=FALSE,  wopt=list(), ...)

## S4 method for signature 'SpatVector'
area(x, ...)
## S4 method for signature 'SpatVector'
perimeter(x)
```

Arguments

`x`	SpatRaster or SpatVector
`filename`	character. Output filename. Optional
`overwrite`	logical. If `TRUE`, `filename` is overwritten
`wopt`	list. Options for writing files as in `writeRaster`
`...`	additional arguments. None implemented

Value

If the coordinate reference system is longitude/latitude the values returned are in square meter for area and meter for perimeter.

In other cases, the unit is set by coordinate references system. In most cases it is meter too.

Examples

```
f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
a <- area(v)
a
sum(a)
perimeter(v)
```

as.character *Create a text representation of (the skeleton of) an object*

Description

Create a text representation of (the skeleton of) an object

Usage

```
## S4 method for signature 'SpatExtent'
as.character(x, ...)
## S4 method for signature 'SpatRaster'
as.character(x, ...)
```

Arguments

x	SpatRaster
...	additional arguments. None implemented

Value

character

Examples

```
r <- rast()
ext(r)
ext(c(0, 20, 0, 20))
```

as.data.frame *Get the attributes of a SpatVector*

Description

Get a data.frame with the attribute values of a SpatVector

Usage

```
## S4 method for signature 'SpatVector'
as.data.frame(x, row.names=NULL, optional=FALSE, ...)
```

Arguments

x	SpatVector
row.names	ignored
optional	ignored
...	additional arguments (none)

Value

data.frame

Examples

```
f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
as.data.frame(v)
```

---

atan2 *Two argument arc-tangent*

Description

For SpatRasters x and y, atan2(y, x) returns the angle in radians for the tangent y/x, handling the case when x is zero. See Trig

See Math-methods for other trigonometric and mathematical functions that can be used with SpatRasters.

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster'
atan2(y, x)
```

Arguments

y	SpatRaster
x	SpatRaster

See Also

Math-methods

Examples

```
r1 <- rast(nrow=10, ncol=10)
r2 <- rast(nrow=10, ncol=10)
values(r1) <- (runif(ncell(r1))-0.5) * 10
values(r2) <- (runif(ncell(r1))-0.5) * 10
atan2(r1, r2)
```

boundaries *Detect boundaries (edges)*

Description

Detect boundaries (edges). boundaries are cells that have more than one class in the 4 or 8 cells surrounding it, or, if classes=FALSE, cells with values and cells with NA.

Usage

```
## S4 method for signature 'SpatRaster'
boundaries(x, classes=FALSE, type="inner",
         directions=8, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
type	character. "inner" or "outer"
classes	character. Logical. If TRUE all different values are (after rounding) distinguished, as well as NA. If FALSE (the default) only edges between NA and non-NA cells are considered
directions	integer. Which cells are considered adjacent? Should be 8 (Queen's case) or 4 (Rook's case)
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster. Cell values are either 1 (a border) or 0 (not a border), or NA

See Also

focal, clump

Examples

```
r <- rast(nrow=18, ncol=36, xmn=0)
v <- rep(NA, ncell(r))
v[150:250] <- 1
v[251:450] <- 2
values(r) <- v
bi <- boundaries(r, type="inner")
bo <- boundaries(r, type="outer")
bc <- boundaries(r, classes=TRUE)
plot(bc)
```

buffer *Create a buffer around vector objects or raster patches*

Description

Calculate a buffer around all cells that are not NA in a SpatRaster, or around the objcts in a SpatVector (currently only implemented for points)

Note that the distance unit of the buffer `width` parameter is meters if the CRS is (`+proj=longlat`), and in map units (typically also meters) if not.

Usage

```
## S4 method for signature 'SpatRaster'
buffer(x, width, filename="", overwrite=FALSE, wopt=list(), ...)
## S4 method for signature 'SpatVector'
buffer(x, width, quadsegs=10, capstyle="round", ...)
```

Arguments

`x`	SpatRaster
`width`	numeric. Unit is meter if `x` has a longitude/latitude CRS, or mapunits in other cases. Should be > 0 for SpatRaster
`filename`	character. Output filename. Optional
`overwrite`	logical. If `TRUE`, `filename` is overwritten
`wopt`	list. Options for writing files as in `writeRaster`
`...`	additional arguments. None implemented
`quadsegs`	postive integer. Number of line segments to use to draw a quart circle
`capstyle`	character. Style of cap to use at the ends of the geometry. Allowed values: "round", "flat", "square" (ignored for now)

Value

SpatRaster

See Also

`distance`

Examples

```
r <- rast(ncol=36,nrow=18)
v <- rep(NA, ncell(r))
v[500] <- 1
values(r) <- v
b <- buffer(r, width=5000000)
plot(b)
```

```
v <- vect(rbind(c(10,10), c(0,60)))
b <- buffer(v, 20)
plot(b)
points(v)

crs(v) <- "+proj=longlat +datum=WGS84"
b <- buffer(v, 1500000)
plot(b)
points(v)
```

---

c *Combine*

Description

Combine SpatRaster objects (similar to `raster::stack`)

Usage

```
## S4 method for signature 'SpatRaster'
c(x, ...)
```

Arguments

x	SpatRaster
...	SpatRaster objects to be combined with x

Value

SpatRaster

Examples

```
r <- rast(nrow=5, ncol=9)
values(r) <- 1:ncell(r)
x <- c(r, r*2, r*3)
```

clamp *Clamp values*

Description

Clamp values to a minimum and maximum value. That is, all values below a lower threshold value and above the upper threshold value become either NA, or, if values=TRUE, become the threshold value

Usage

```
## S4 method for signature 'SpatRaster'
clamp(x, lower=-Inf, upper=Inf, values=TRUE,
             filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
lower	numeric. lowest value
upper	numeric. highest value
values	logical. If FALSE values outside the clamping range become NA, if TRUE, they get the extreme values
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

See Also

classify

Examples

```
r <- rast(ncols=10, nrows=10)
values(r) <- 1:ncell(r)
rc <- clamp(r, 25, 75)
rc
```

`classify` *Classify (or reclassify) cell values*

Description

Classify values of a SpatRaster. The function (re-)classifies groups of values to other values.

Classification can be based on ranges "from-to-becomes" or on specific values "is-becomes", or on "cuts".

With "from-to-becomes" or "is-becomes", classification is done with matrix `rcl`, in the row order of the matrix. Thus, if there are overlapping ranges or values, the first time a number is within a range determines the reclassification value.

With "cuts" the values are sorted, so that the order in which they are provided does not matter.

Usage

```
## S4 method for signature 'SpatRaster'
classify(x, rcl, include.lowest=FALSE, right=TRUE,
     othersNA=FALSE, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

`x`	SpatRaster
`rcl`	matrix for classification. This matrix must have 1, 2 or 3 columns. If there are three columns, the first two columns are "from" "to" of the input values, and the third column "becomes" has the new value for that range. The two column matrix ("is", "becomes") can be useful for re-classifying integer values. In that case, the `right` argument is automatically set to `NA`. A single column matrix or a vector is interpreted as a set of cuts. In that case the values are classified based on their location inbetween the cut-values
`include.lowest`	logical, indicating if a value equal to the lowest value in `rcl` (or highest value in the second column, for `right=FALSE`) should be included.
`right`	logical, indicating if the intervals should be closed on the right (and open on the left) or vice versa. The default is `TRUE`. A special case is to use right=NA. In this case both the left and right intervals are open
`othersNA`	logical. If `TRUE`, values that are not matched become `NA`. If `FALSE`, they retain their original value.
`filename`	character. Output filename. Optional
`overwrite`	logical. If `TRUE`, `filename` is overwritten
`wopt`	list. Options for writing files as in `writeRaster`
`...`	additional arguments. None implemented

Value

SpatRaster

Note

For model-based classification see predict

Examples

```
r <- rast(ncols=10, nrows=10)
values(r) <- (0:99)/99
# classify the values into three groups
# all values >= 0 and <= 0.25 become 1, etc.
m <- c(0, 0.25, 1,
       0.25, 0.5, 2,
   0.5, 1, 3)
rclmat <- matrix(m, ncol=3, byrow=TRUE)
rc1 <- classify(r, rclmat, include.lowest=TRUE)

# equivalent to
rc2 <- classify(r, c(0, 0.25, 0.5, 1), include.lowest=TRUE)

# is becomes
x <- round(r*5)
unique(x)
m <- rbind(c(1,100), c(2,200))
m
rcx1 <- classify(x, m)
unique(rcx1)
rcx2 <- classify(x, m, othersNA=TRUE)
unique(rcx2)
```

---

click *Query by clicking on a map*

Description

Click on a map (plot) to get values of a SpatRaster at that location; and optionally the coordinates and cell number of the location. For SpatVectors you need to click twice (draw a box).

Usage

```
## S4 method for signature 'SpatRaster'
click(x, n=Inf, id=FALSE, xy=FALSE, cell=FALSE, type="n", show=TRUE, ...)
```

Arguments

x	SpatRaster or SpatVector, or missing
n	number of clicks on the plot (map)
id	Logical. If TRUE, a numeric ID is shown on the map that corresponds to the row number of the output
xy	Logical. If TRUE, xy coordinates are included in the output

cell	Logical. If TRUE, cell numbers are included in the output
type	One of "n", "p", "l" or "o". If "p" or "o" the points are plotted; if "l" or "o" they are joined by lines. See ?locator
show	logical. Print the values after each click?
...	additional graphics parameters used if type != "n" for plotting the locations. See ?locator

Value

The value(s) of x at the point(s) clicked on (or touched by the box drawn).

Note

The plot only provides the coordinates for a spatial query, the values are read from the SpatRaster that is passed as an argument. Thus you can extract values from an object that has not been plotted, as long as it spatialy overlaps with with the extent of the plot.

Unless the process is terminated prematurely values at at most n positions are determined. The identification process can be terminated by clicking the second mouse button and selecting 'Stop' from the menu, or from the 'Stop' menu on the graphics window.

See Also

draw

Examples

```
r <-rast(system.file("exdata/test.tif", package="terra"))
plot(r)
click(r)
 # now click on the plot (map)
```

---

coerce *Coercion to other object types*

Description

Coercion to other object types

Usage

```
## S4 method for signature 'SpatRaster'
as.vector(x, mode='any')
## S4 method for signature 'SpatRaster'
as.matrix(x, wide=FALSE, ...)
## S4 method for signature 'SpatRaster'
as.data.frame(x, xy=FALSE, cells=FALSE, ...)
```

```
## S4 method for signature 'SpatRaster'
as.array(x, ...)
## S4 method for signature 'SpatRaster'
as.polygons(x, values=FALSE, na.rm=FALSE, ...)
## S4 method for signature 'SpatRaster'
as.points(x, values=FALSE, na.rm=FALSE, ...)
## S4 method for signature 'SpatVector'
as.lines(x, ...)
```

Arguments

x	SpatRaster
wide	logical
xy	logical
cells	logical
mode	this argument is ignored
values	logical; include cell values as attributes?
na.rm	logical; exclude cells with NAs?
...	additional arguments. None implemented

Value

vector, matrix, array, data.frame or SpatVector

Examples

```
r <- rast(ncol=2, nrow=2)
values(r) <- 1:ncell(r)

as.vector(r)
as.matrix(r)
as.matrix(r, wide=TRUE)
as.data.frame(r, xy=TRUE)
as.array(r)
as.polygons(r)
as.points(r)
```

---

collapse *Collapse cell values from a multi-layer SpatRaster*

Description

Use a single layer SpatRaster object to select cell values from different layers in a multi-layer SpatRaster and "collapse" it into a single layer. The values of the SpatRaster to select layers (y) should be between 1 and nlyr(x) (values outside this range are ignored); they are also truncated to integers.

See extract for extraction of values by cell, point, or otherwise.

Usage

```
## S4 method for signature 'SpatRaster'
collapse(x, y, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
y	SpatRaster
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

See Also

tapp, extract

Examples

```
r <- rast(ncol=10, nrow=10)
values(r) <- 1
s <- c(r, r+2, r+5)
set.seed(1)
values(r) <- round((runif(ncell(r)))*3)
x <- collapse(s, r)
```

---

compareGeom *Compare geometries of SpatRaster objects*

Description

Evaluate whether two SpatRaster objects have the same extent, number of rows and columns, projection, resolution, and origin (or a subset of these comparisons).

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster'
compareGeom(x, y, ..., lyrs=TRUE,
          crs=TRUE, warncrs=FALSE, ext=TRUE, rowcol=TRUE, res=FALSE)
```

Arguments

x	SpatRaster
y	SpatRaster
...	Additional SpatRaster objects
lyrs	logical. If TRUE, the number of layers is compared
crs	logical. If TRUE, coordinate reference systems are compared.
warncrs	logical. If TRUE, a warning is given if the crs is different (instead of an error)
ext	logical. If TRUE, bounding boxes are compared
rowcol	logical. If TRUE, number of rows and columns of the objects are compared
res	logical. If TRUE, resolutions are compared (redundant when checking extent and rowcol)

Examples

```
r1 <- rast()
r2 <- rast()
r3 <- rast()
compareGeom(r1, r2, r3)
nrow(r3) <- 10

compareGeom(r1, r3)
```

---

contour *Contour plot*

Description

Contour lines of a SpatRaster. Use add=TRUE to add the lines to the current plot. See contour for details.

if filled=TRUE, a new filled contour plot is made. See filled.contour for details.

as.contour returns the contour lines as a SpatVector.

Usage

```
## S4 method for signature 'SpatRaster'
contour(x, maxcells=100000, filled=FALSE, ...)
## S4 method for signature 'SpatRaster'
as.contour(x, maxcells=100000, ...)
```

Arguments

x	SpatRaster. Only the first layer is used
maxcells	maximum number of pixels used to create the contours
filled	logical. If TRUE, a filled.contour plot is made
...	any argument that can be passed to contour or filled.contour (graphics package)

See Also

plot

Examples

```
r <- rast(system.file("exdata/test.tif", package="terra"))
plot(r)
contour(r, add=TRUE)

v <- as.contour(r)
plot(r)
lines(v)

contour(r, filled=TRUE, nlevels=5)
```

---

cover *Cover (replace) NA values with values of another raster*

Description

Replace NA values (or another value) in SpatRaster (x) with the values of SpatRaster (y)

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster'
cover(x, y, value=NA, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
y	SpatRaster
value	numeric. The cell values in x to be replaced by the values in y
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

Examples

```
r1 <- r2 <- rast(ncols=36, nrows=18)
values(r1) <- 1:ncell(r1)
values(r2) <- runif(ncell(r2))
r2 <- classify(r2, cbind(-Inf, 0.5, NA))
r3 <- cover(r2, r1)
```

---

crop *Cut out a geographic subset*

Description

Cut out a part of a SpatRaster with a SpatExtent, or another object from which an extent can be obtained. You can only extract rectangular areas, but see mask for setting cell values within SpatRaster to NA.

Usage

```
## S4 method for signature 'SpatRaster,ANY'
crop(x, y, snap="near", filename="", overwrite=FALSE, wopt=list(), ...)
## S4 method for signature 'SpatVector,SpatVector'
crop(x, y, ...)
```

Arguments

x	SpatRaster or SpatVector
y	SpatExtent or other object that has a SpatExtent; or SpatVector if x is a SpatVector
snap	character. One of "near", "in", or "out"
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

Examples

```
r <- rast(xmin=0, xmax=10, ymin=0, ymax=10, nrows=25, ncols=25)
values(r) <- 1:ncell(r)
e <- ext(-5, 5, -5, 5)
rc <- crop(r, e)
```

---

crs *Get or set a coordinate reference system (projection)*

Description

Get or set the coordinate reference system (CRS) of a SpatRaster object.

Usage

```
## S4 method for signature 'SpatRaster'
crs(x)
## S4 method for signature 'SpatVector'
crs(x)
## S4 replacement method for signature 'SpatRaster'
crs(x, ...)<-value
## S4 replacement method for signature 'SpatVector'
crs(x, ...)<-value
```

Arguments

x	SpatRaster
value	character string describing a coordinate reference system in the PROJ.4 "+" format
...	additional arguments (none implemented)

Value

character or modified SpatRaster

Examples

```
r <- rast()
crs(r)
crs(r) <- "+proj=lcc +lat_1=48 +lat_2=33 +lon_0=-100 +ellps=WGS84"
crs(r)
```

density *Density plot*

Description

Create density plots of the cell values of a SpatRaster

Usage

```
## S4 method for signature 'SpatRaster'
density(x, maxcells=100000, plot=TRUE, main, ...)
```

Arguments

x	SpatRaster
maxcells	the maximum number of (randomly sampled) cells to be used for creating the plot
plot	if TRUE produce a plot, else return a density object
main	character. Caption of plot(s)
...	additional arguments passed to plot

Value

density plot (and a density object, returned invisibly if plot=TRUE)

Examples

```
logo <- rast(system.file("exdata/logo.tif", package="terra"))
density(logo)
```

dimensions *Dimensions of a SpatRaster or SpatVector*

Description

Get the number of rows (nrow), columns (ncol), cells (ncell), layers (nlyr), sources (nsrc) or the spatial resolution of a SpatRaster. Set the number of rows or columns or layers. When setting dimensions, all sources and values are dropped.

Usage

```
## S4 method for signature 'SpatRaster'
ncol(x)
## S4 method for signature 'SpatRaster'
nrow(x)
## S4 method for signature 'SpatRaster'
nlyr(x)
## S4 method for signature 'SpatRaster'
ncell(x)
## S4 method for signature 'SpatRaster'
size(x)
## S4 method for signature 'SpatRaster'
nsrc(x)
## S4 method for signature 'SpatRaster'
res(x)

ncol(x, ...) <- value
nrow(x, ...) <- value
nlyr(x, ...) <- value
res(x) <- value

## S4 method for signature 'SpatRaster'
xres(x)
## S4 method for signature 'SpatRaster'
yres(x)

## S4 method for signature 'SpatVector'
ncol(x)
## S4 method for signature 'SpatVector'
nrow(x)
## S4 method for signature 'SpatVector'
size(x, ...)
## S4 method for signature 'SpatVector'
length(x)
```

Arguments

x	SpatRaster or SpatVector
value	For ncol and nrow: positive integer. For res: one or two positive numbers
...	additional arguments. None implemented

Value

integer

See Also

ext

Examples

```
r <- rast()
ncell(r)
ncol(r)
nrow(r)
dim(r)
nsrc(r)

nrow(r) <- 18
ncol(r) <- 36
# equivalent to
dim(r) <- c(18, 36)

dim(r)
dim(r) <- c(10, 10, 5)
dim(r)

xres(r)
yres(r)
res(r)

res(r) <- 1/120
# different xres and yres
res(r) <- c(1/120, 1/60)
```

---

disaggregate *Disaggregate raster cells*

Description

Disaggregate a SpatRaster to create a new SpatRaster with a higher resolution (smaller cells). The values in the new SpatRaster are the same as in the larger original cells.

Usage

```
## S4 method for signature 'SpatRaster'
disaggregate(x, fact, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
fact	positive integer. Aggregation factor expressed as number of cells in each direction (horizontally and vertically). Or two integers (horizontal and vertical aggregation factor) or three integers (when also aggregating over layers)

filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

See Also

aggregate

Examples

```
r <- rast(ncols=10, nrows=10)
rd <- disaggregate(r, fact=c(10, 2))
ncol(rd)
nrow(rd)
values(r) <- 1:ncell(r)
rd <- disaggregate(r, fact=c(4, 2))
```

---

distance *Geographic distance*

Description

If x is a SpatRaster:

If y is missing this method computes the distance, for all cells that are NA in SpatRaster x to the nearest cell that is not NA. If argument grid=TRUE, the distance is computed using a path that goes through the centers of the 8 neighboring cells.

If y is a SpatVector, the distance to that SpatVector is computed for all cells. For lines and polygons this is done after rasterization (for now).

If x is a SpatVector:

If y is missing, a distance matrix between all object in x is computed. An distance matrix object of class "dist" is returned.

If y is a SpatVector the geographic distance between all objects is computed (and a matrix is returned). If both sets have the same number of points, and pairwise=TRUE, the distance between each pair of objects is computed, and a vector is returned.

Usage

```
## S4 method for signature 'SpatRaster,missing'
distance(x, y, grid=FALSE, filename="", overwrite=FALSE, wopt=list(), ...)

## S4 method for signature 'SpatRaster,SpatVector'
distance(x, y, filename="", overwrite=FALSE, wopt=list(), ...)

## S4 method for signature 'SpatVector,missing'
distance(x, y, ...)

## S4 method for signature 'SpatVector,SpatVector'
distance(x, y, pairwise=FALSE, ...)
```

Arguments

x	SpatRaster or SpatVector
y	missing or SpatVector
grid	logical. If TRUE, distance is computed using a path that goes through the centers of the 8 neighboring cells
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented
pairwise	logical. If TRUE and if x and y have the same size (number of rows), the pairwise distances are returned instead of the distances between all elements

Value

SpatRaster or numeric or matrix or distance matrix (object of class "dist")

The unit is in meters if the CRS is longlat and in map units (typically also meters) when it is not.

Examples

```
r <- rast(ncol=36,nrow=18)
v <- rep(NA, ncell(r))
v[500] <- 1
values(r) <- v
d <- distance(r)

plot(d / 100000)

p1 <- vect(rbind(c(0,0), c(90,30), c(-90,-30)), crs="+proj=longlat +datum=WGS84")
dp <- distance(r, p1)

d <- distance(p1)
d
```

```
as.matrix(d)

p2 <- vect(rbind(c(30,-30), c(25,40), c(-9,-3)), crs="+proj=longlat +datum=WGS84")
dd <- distance(p1, p2)
dd
pd <- distance(p1, p2, pairwise=TRUE)
pd
pd == diag(dd)
```

---

draw *Draw a polygon, line, extent, or points*

Description

Draw on a plot (map) to get a SpatVector or SpatExtent object for later use. After calling the function, start clicking on the map. To finish, right-click and select "stop".

Usage

```
## S4 method for signature 'character'
draw(x="extent", col="red", lwd=2, ...)
```

Arguments

x	character. The type of object to draw. One of "extent", "polygon", "line", or "points"
col	the color to be used
lwd	the width of the lines to be drawn
...	additional arguments passed to locator

Value

SpatVector or SpatExtent

See Also

locator

extend *Extend*

Description

Extend returns an SpatRaster object with a larger spatial extent. See crop if you (also) want to remove rows or columns.

There is also an extend method for Extent objects to enlarge (or app) an Extent. You can also use algebraic notation to do that (see examples)

Usage

```
## S4 method for signature 'SpatRaster'
extend(x, y, filename="", overwrite=FALSE, wopt=list(), ...)

## S4 method for signature 'SpatExtent'
extend(x, y, ...)
```

Arguments

x	SpatRaster or SpatExtent
y	If x is a SpatRaster, y should be a SpatExtent, or an object from which it can be extracted (such as SpatRaster and Spatvector objects). Alternatively, you can provide a numeric vector of length 2 indicating the number of rows and columns that need to be added (or a single number when the number of rows and columns is equal) If x is an Extent object, y should be a numeric vector of 1, 2, or 4 elements
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster or SpatExtent

See Also

crop, merge

Examples

```
r <- rast(xmin=-150, xmax=-120, ymin=30, ymax=60, ncol=36, nrow=18)
values(r) <- 1:ncell(r)
e <- ext(-180, -100, 40, 70)
re <- extend(r, e)

# extend with a number of rows and columns (at each side)
re2 <- extend(r, c(2,10))

# Extent object
e <- ext(r)
e
extend(e, 10)
extend(e, 10, -10, 0, 20)
```

`extent` *Create, get or set a SpatExtent*

Description

Get a SpatExtent of a SpatRaster, or coordinates from such an object. Or create a SpatExtent from a vector (length=4; order= xmin, xmax, ymin, ymax)

Usage

```
## S4 method for signature 'SpatRaster'
ext(x, ...)
## S4 replacement method for signature 'SpatRaster,SpatExtent'
ext(x)<-value
```

Arguments

`x`	SpatRaster
`value`	SpatExtent
`...`	additional arguments. None implemented

Value

SpatExtent or numeric coordinate

Examples

```
r <- rast()
as.character(r)
as.character(ext(r))
```

extract *Extract values from a SpatRaster*

Description

Extract values from a SpatRaster for the locations of a SpatVector

Usage

```
## S4 method for signature 'SpatRaster,SpatVector'
extract(x, y, fun=NULL, ..., method="simple", drop=FALSE)
## S4 method for signature 'SpatRaster,matrix'
extract(x, y, ...)
```

Arguments

x	SpatRaster
y	SpatVector (for points, lines, polygons), or 2-column matrix (x, y; for points)
fun	character. function to summarize the data. Currently ignored
...	additional arguments. None implemented
method	character. method for extracting values with points. The default is "simple", the alternative is "bilinear"
drop	boolean. If TRUE, the output is simplified

Value

data.frame

See Also

values

Examples

```
r <- rast(ncol=5, nrow=5, xmin=0, xmax=5, ymin=0, ymax=5)
values(r) <- 1:25
xy <- rbind(c(0.5,0.5), c(2.5,2.5))
p <- vect(xy)

extract(r, xy)
extract(r, p)

r[1,]
r[5]
r[,5]

r[c(0:2, 99:101)]
```

```
f <- system.file("exdata/test.tif", package="terra")
r <- rast(f)

xy <- cbind(179000, 330000)
xy <- rbind(xy-100, xy, xy+1000)
extract(r, xy)

p <- vect(xy)
g <- geom(p)
g

extract(r, p)

x <- r + 10
extract(x, p)

i <- cellFromXY(r, xy)
x[i]
r[i]

y <- c(x,x*2,x*3)
y[i]

# extract with a polygon
f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
z <- rast(v)
values(z) <- 1:100
e <- extract(z, v)
e[1:2]
rapply(e, mean)

x <- c(z, z*2, z/3)
e <- extract(x, v)
matrix(rapply(e, mean), ncol=nlyr(x), byrow=TRUE)
```

`factors` *Factors*

Description

These functions allow for defining a SpatRaster layer as a categorical variable. The cell values are an index (id), whereas the actual values are stored seperately, in a table (sometimes called "Raster Attribute Table"). This table is a data.frame. The first column in the RAT ("ID") has the unique cell values of the layer; this column should normally not be changed. The other columns can be of any basic type (factor, character, integer, numeric or logical).

Function 'levels' returns the RAT for inspection. It can be modified and set using `levels <-value`.

`as.factor` and `ratify` create a layer with a RAT table. `deratify` creates a single layer for a (or each) variable in the RAT table.

Usage

```
is.factor(x)
as.factor(x)
levels(x)
## S4 method for signature 'SpatRaster'
rats(x)
setRat(x, rat)
```

Arguments

x	SpatRaster
rat	list

Value

SpatRaster; list (levels); boolean (is.factor)

Examples

```
set.seed(0)
r <- rast(nrow=10, ncol=10)
values(r) <- runif(ncell(r)) * 10
is.factor(r)

r <- round(r)
f <- as.factor(r)
is.factor(f)
```

---

flip *Flip a raster*

Description

Flip the values of a SpatRaster by inverting the order of the rows (`vertical=TRUE`) or the columns (`vertical=FALSE`).

Usage

```
## S4 method for signature 'SpatRaster'
flip(x, vertical=TRUE, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
vertical	logical. If `TRUE`, values are flipped by rows, if `FALSE`, values are flipped by columns
filename	character. Output filename. Optional

overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

See Also

transpose, rotate

Examples

```
r <- rast(nrow=18, ncol=36)
m <- matrix(1:ncell(r), nrow=18)
values(r) <- as.vector(t(m))
rx <- flip(r, vertical=FALSE)
values(r) <- as.vector(m)
ry <- flip(r, vertical=TRUE)
```

---

focal *Focal values*

Description

Calculate focal ("moving window") values for the neighborhood of focal cells using a matrix of weights, perhaps in combination with a function.

Usage

```
## S4 method for signature 'SpatRaster'
focal(x, w=3, na.rm=TRUE, fillvalue=NA, fun="sum",
            filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
w	matrix of weights (the moving window), e.g. a 3 by 3 matrix with values 1; see Details. The matrix does not need to be square, but the sides must be odd numbers. If you need even sides, you can add a column or row with weights of zero
na.rm	logical. If TRUE, NA will be removed from focal computations. The result will only be NA if all focal cells are NA. Except for some special cases (weights of 1, functions like min, max, mean), using na.rm=TRUE is generally not a good idea in this function because it will unbalance the effect of the weights
fillvalue	numeric. The value of the cells of the padded rows and columns

fun	function (optional). The function fun should take multiple numbers, and return a single number. For example mean, modal, min or max. It should also accept a na.rm argument (or ignore it, e.g. as one of the 'dots' arguments. For example, length will fail, but function(x,...){na.omit(length(x))} works.
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Details

focal uses a matrix of weights for the neighborhood of the focal cells. The default function is sum. It is computationally much more efficient to adjust the weights-matrix than to use another function through the fun argument. Thus while the following two statements are equivalent (if there are no NA values), the first one is faster than the second one:

```
a <-focal(x,w=matrix(1/9,nc=3,nr=3))
```

```
b <-focal(x,w=matrix(1,3,3),fun=mean)
```

There is, however, a difference if NA values are considered. One can use the na.rm=TRUE option which may make sense when using a function like mean. However, the results would be wrong when using a weights matrix.

Laplacian filter: `filter=matrix(c(0,1,0,1,-4,1,0,1,0),nrow=3)`

Sobel filter: `filter=matrix(c(1,2,1,0,0,0,-1,-2,-1) / 4,nrow=3)`

Value

SpatRaster

Examples

```
r <- rast(ncols=36, nrows=18, xmn=0)
values(r) <- 1:ncell(r)

x <- focal(r, w=3, fun=mean)
```

---

freq *Frequency table*

Description

Frequency table of the values of a SpatRaster. NAs are not counted.

Usage

```
## S4 method for signature 'SpatRaster'
freq(x, bylayer=TRUE, ...)
```

Arguments

x	SpatRaster
bylayer	logical. If TRUE tabulation is done by layer
...	additional arguments (none implemented)

Value

matrix with 2 columns (value, count) or, if bylayer=TRUE three layers (layer, value, count).

See Also

freq

Examples

```
r <- rast(nrow=10, ncol=10)
set.seed(2)
values(r) <- sample(5, ncell(r), replace=TRUE)

freq(r, FALSE)
freq(r)

x <- c(r, r/2)
freq(x, FALSE)
freq(x)

freq(round(x))
```

---

geom *Get the geometry (coordinates) of a SpatVector*

Description

Get the geometry of a SpatVector. This is a five-column matrix: the vector object ID, the IDs for the parts of each object (e.g. five polygons that together are one spatial object), the x (longitude) and y (latitude) coordinates, and a flag indicating whether the part is a "hole" (only relevant for polygons).

Usage

```
## S4 method for signature 'SpatVector'
geom(x, ...)
```

Arguments

x	SpatVector
...	additional arguments. None implemented

Value

matrix

See Also

See xyFromCell to get the coordinates of the cells of a SpatRaster

Examples

```
x1 <- rbind(c(-175,-20), c(-140,55), c(10, 0), c(-140,-60))
x2 <- rbind(c(-10,0), c(140,60), c(160,0), c(140,-55))
x3 <- rbind(c(-125,0), c(0,60), c(40,5), c(15,-45))
x4 <- rbind(c(80,0), c(105,13), c(120,2), c(105,-13))
z <- rbind(cbind(object=0, part=0, x1), cbind(object=1, part=0, x2),
           cbind(object=2, part=0, x3), cbind(object=2, part=1,  x4))
colnames(z)[3:4] <- c('x', 'y')
z <- data.frame(z)
z$hole <- 0
z$hole[z$object==2 & z$part==1] <- 1

p <- vect(z)
geom(p)

f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
g <- geom(v)
head(g)
```

---

geomtype *Geometry type of a SpatVector*

Description

Get the geometry type (points, lines, or polygons) of a SpatVector

Usage

```
## S4 method for signature 'SpatVector'
geomtype(x, ...)
```

Arguments

x	SpatVector
...	additional arguments. None implemented

Value

character

`global` *global statistics*

Description

Compute global statistics, that is summarized values of an entire SpatRaster.

If `x` is very large `global` will fail, except when `fun` is one of "mean", "min", "max", or "sum".

Usage

```
## S4 method for signature 'SpatRaster'
global(x, fun="mean", ...)
```

Arguments

`x`	SpatRaster
`fun`	function to be applied to summarize the values by zone. Either as character:"mean", "min", "max", "sum"; or, for relatively small SpatRasters, a proper function
`...`	additional arguments passed on to `fun`

Value

A `data.frame` with a row for each layer

See Also

`zonal` for "zonal" statistics, and `app` or `Summary-methods` for "local" statistics, and `extract` for summarizing values for polygons. Also see `focal` for "focal" or "moving window" operations.

Examples

```
r <- rast(ncols=10, nrows=10)
values(r) <- 1:ncell(r)
global(r, "sum")
global(r, "mean", na.rm=TRUE)
```

head and tail *Show the head or tail of a Spat* object*

Description

Show the head (first values) or tail (last values) of a SpatRaster or of the attributes of a SpatVector.

Usage

```
head(x, ...)
tail(x, ...)
```

Arguments

x	SpatRaster or SpatVector
...	additional arguments passed on to other methods

Value

matrix (SpatRaster) or data.frame (SpatVector)

See Also

show, geom

Examples

```
r <- rast(nrow=25, ncol=25)
values(r) <- 1:ncell(r)
head(r)
tail(r)
```

hist *Histogram*

Description

Create a histogram of the values of a SpatRaster. For large datasets a sample of `maxcell` is used.

Usage

```
## S4 method for signature 'SpatRaster'
hist(x, layer, maxcell=1000000, plot=TRUE, main, ...)
```

Arguments

x	SpatRaster
layer	integer (or character) to indicate layer number (or name). Can be used to subset the layers to plot in a multilayer SpatRaster
maxcell	integer. To regularly sample very large objects
plot	logical. Plot the histogram or only return the histogram values
main	character. Main title(s) for the plot. Default is the value of names
...	additional arguments. See hist

Value

This function is principally used for plotting a histogram, but it also returns an object of class "histogram" (invisibly if plot=TRUE).

See Also

pairs,boxplot

Examples

```
r1 <- r2 <- rast(nrows=50, ncols=50)
values(r1) <- runif(ncell(r1))
values(r2) <- runif(ncell(r1))
rs <- r1 + r2
rp <- r1 * r2

opar <- par(no.readonly =TRUE)
par(mfrow=c(2,2))
plot(rs, main='sum')
plot(rp, main='product')
hist(rs)
a <- hist(rp)
a
x <- c(rs, rp, sqrt(rs))
hist(x)
par(opar)
```

---

ifel *ifelse for SpatRaster objects*

Description

This functions like ifelse but this one works for SpatRaster objects. This is "syntactic sugar", that is, it can be easier to express what is desired; but there is no real need for it, as you can achieve the same things with combinations of app, reclassify, mask, and cover.

ifel is R equivalent to the Con method in ArcGIS (arcpy).

Usage

```
## S4 method for signature 'SpatRaster'
ifel(test, yes, no, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

test	SpatRaster
yes	SpatRaster or numeric
no	SpatRaster or numeric
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

Examples

```
r <- rast(nrows=5, ncols=5)
values(r) <- -10:14

x <- ifel(r > 1, 1, r)
# same as
a <- classify(r, cbind(1, Inf, 1))
b <- app(r, fun=function(i) {i[i > 1] <- 1; i})
d <- clamp(r, -Inf, 1)

y <- ifel(r > 1, 1, ifel(r < -1, -1, r))

z <- ifel(r > -2 & r < 2, 100, 0)

k <- ifel(r > 0, r+10, ifel(r < 0, r-10, 3))
```

---

initialize *Initialize a SpatRaster with values*

Description

Create a SpatRaster with values reflecting a cell property: 'x', 'y', 'col', 'row', 'cell' or 'chess'. Alternatively, a function can be used. In that case, cell values are initialized without reference to pre-existing values. E.g., initialize with a random number (fun=runif). While there are more direct ways of achieving this for small objects (see examples) for which a vector with all values can be created in memory, the init function will also work for Raster* objects with many cells.

Usage

```
## S4 method for signature 'SpatRaster'
init(x, fun, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
fun	function to be applied. This must be a either a single number, a function, or one of a set of character values. A function must take the number of cells as a single argument to return a vector of values with a length equal to the number of cells, such as `fun=runif`. Allowed character values are 'x', 'y', 'row', 'col', 'cell', and 'chess' to get the x or y coordinate, row, col or cell number or a chessboard pattern (alternating 0 and 1 values)
filename	character. Output filename. Optional
overwrite	logical. If `TRUE`, `filename` is overwritten
wopt	list. Options for writing files as in `writeRaster`
...	additional arguments. None implemented

Value

SpatRaster

Examples

```
r <- rast(ncol=10, nrow=5, xmn=0, xmx=10, ymn=0, ymx=5)
x <- init(r, fun="cell")
y <- init(r, fun=runif)

# initialize with a single value
z <- init(r, fun=8)
```

---

interpolate *Interpolate*

Description

Make a RasterLayer with interpolated values using a fitted model object of classes such as "gstat" (gstat package) or "Krige" (fields package). That is, these are models that have location ("x" and "y", or "longitude" and "latitude") as independent variables. If x and y are the only independent variables provide an empty (no associated data in memory or on file) SpatRaster for which you want predictions. If there are more spatial predictor variables provide these as a Raster* object in the first argument of the function. If you do not have x and y locations as implicit predictors in your model you should use `predict` instead.

Usage

```
## S4 method for signature 'SpatRaster'
interpolate(object, model, fun=predict, ...,
       xyNames=c("x", "y"), factors=NULL, const=NULL, index = NULL,
   na.rm=FALSE, filename="", overwrite=FALSE, wopt=list())
```

Arguments

object	SpatRaster
model	model object
fun	function. Default value is "predict", but can be replaced with e.g. "predict.se" (depending on the class of the model object)
...	additional arguments passed to fun
xyNames	character. variable names that the model uses for the spatial coordinates. E.g., c("longitude","latitude")
factors	list with levels for factor variables. The list elements should be named with names that correspond to names in object such that they can be matched. This argument may be omitted for many models as the predict function will extract the levels from the model object
const	data.frame. Can be used to add a constant for which there is no Raster object for model predictions. This is particularly useful if the constant is a character-like factor value
index	positive integer or NULL. Allows for selecting of the variable returned if the model returns multiple variables
na.rm	logical. If TRUE, cells with NA values in the predictors are removed from the computation. This option prevents errors with models that cannot handle NA values. In most other cases this will not affect the output. An exception is when predicting with a model that returns predicted values even if some (or all!) variables are NA
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster

Value

SpatRaster

See Also

predict

`isLonLat` *Check for longitude/latitude crs*

Description

Test whether a SpatRaster or SpatVector has a longitude/latitude coordinate reference system (CRS). `couldBeLonLat` returns `TRUE` if the CRS is unknown (`""`) but the x coordinates are within -181 and 181 and the y coordinates are within -90.1 and 90.1.

Usage

```
## S4 method for signature 'SpatRaster'
isLonLat(x, ...)
## S4 method for signature 'SpatVector'
isLonLat(x, ...)
## S4 method for signature 'SpatRaster'
isGlobalLonLat(x, ...)
## S4 method for signature 'SpatRaster'
couldBeLonLat(x, warn=TRUE, ...)
## S4 method for signature 'SpatVector'
couldBeLonLat(x, warn=TRUE, ...)
```

Arguments

`x`	SpatRaster or SpatVector object
`warn`	logical. If `TRUE`, a warning is given if the CRS is unknown or when the CRS is longitude/latitude but the coordinates do not match that
`...`	additional arguments. None implemented

Value

Logical

Examples

```
r <- rast()
isLonLat(r)
crs(r) <- "+proj=lcc +lat_1=48 +lat_2=33 +lon_0=-100 +ellps=WGS84"
isLonLat(r)
```

`local` *Local statistics*

Description

Compute cell (pixel) level "local" statistics across layers, and most return a single layer (the exception is `range`).

The following summary methods are available for SpatRaster: `any,all,max,min,mean,prod,range,stdev,sum`

See `app` to compute statistics that are not included here (e.g. median) or any other custom functions.

See `modal` to compute the mode.

Because generic functions are used, the method applied is chosen based on the first argument: "`x`". This means that if `r` is a SpatRaster, `mean(r,5)` will work, but `mean(5,r)` will not work.

The `mean` method has an argument "trim" that is ignored.

The `stdev` mehtod returns the population standard deviation, computed as such:

`f <-function(x) sqrt(sum((x-mean(x))^2) / length(x))`

This is different than the sample standard deviation retunred by `sd` (which uses `n-1` as denominator). Function `f` above is equivalent to function `g` below

`g <-function(x) sqrt(sum((x-mean(x))^2) / length(x))`

Usage

```
## S4 method for signature 'SpatRaster'
min(x, ..., na.rm=FALSE)
## S4 method for signature 'SpatRaster'
max(x, ..., na.rm=FALSE)
## S4 method for signature 'SpatRaster'
range(x, ..., na.rm=FALSE)
## S4 method for signature 'SpatRaster'
mean(x, ..., trim=NA, na.rm=FALSE)
## S4 method for signature 'SpatRaster'
stdev(x, ..., na.rm=FALSE)
```

Arguments

`x`	SpatRaster
`...`	additional argument of the same type as `x` or numeric
`trim`	ignored
`na.rm`	logical. If `TRUE`, `NA` values are ignored. If `FALSE`, `NA` is returned if `x` has any `NA` values

Value

SpatRaster

See Also

Math-methods,modal

Examples

```
set.seed(0)
r <- rast(nrow=10, ncol=10, nlyr=3)
values(r) <- runif(size(r))

x <- mean(r)
x <- sum(r, r[[2]], 3)
max(r)
max(r, 0.5)

y <- stdev(r)
# not the same as
yy <- app(r, sd)

z <- stdev(r, r*2)
```

---

mask *Mask values in a SpatRaster*

Description

Create a new SpatRaster that has the same values as SpatRaster x, except for the cells that are NA (or another maskvalue in another SpatRaster (the 'mask') or not covered by the objects of a SpatVector. These cells become NA (or another updatevalue).

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster'
mask(x, mask, inverse=FALSE, maskvalue=NA,
   updatevalue=NA, filename="", overwrite=FALSE, wopt=list(), ...)
## S4 method for signature 'SpatRaster,SpatVector'
mask(x, mask, inverse=FALSE, updatevalue=NA,
   filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
mask	SpatRaster
inverse	logical. If TRUE, areas on mask that are _not_ the maskvalue are masked
maskvalue	numeric. The value in mask that indicates the cells of x that should become updatevalue (default = NA)
updatevalue	numeric. The value that cells of x should become if they are not covered by mask (and not NA)

filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster object

See Also

crop

Examples

```
r <- rast(ncol=10, nrow=10)
m <- rast(ncol=10, nrow=10)
values(r) <- 1:100
x <- runif(ncell(r))
x[x < 0.5] <- NA
values(m) <- x
mr <- mask(r, m)
```

---

math *Mathematical operations with SpatRaster objects*

Description

Standard arithmetic operators for computations with SpatRaster objects and numeric values. The following operators and methods are available for SpatRaster:

Arith: +,-,*,/,^,%%,%/%

Compare: ==,!=,>,<,<=,>=

Summary: "max","min","range","prod","sum","any","all"

Math: "abs","sign","sqrt","ceiling","floor","trunc","cummax","cummin","cumprod","cumsum","log","log

Math2: "round","signif"

If multiple SpatRaster objects are used, these must have the same extent and resolution. Operators are applied on a cell by cell basis.

For **SpatExtent** the following methods have been implemented: "round","floor","ceil","=="

Value

SpatRaster

Note

You can use the somewhat more flexible app method instead of the Math-methods.

Examples

```
r1 <- rast(ncols=10, nrows=10)
values(r1) <- runif(ncell(r1))
r2 <- rast(r1)
values(r2) <- 1:ncell(r2) / ncell(r2)
r3 <- r1 + r2
r2 <- r1 / 10
r3 <- r1 * (r2 - 1 / r2)


b <- c(r1, r2, r3)
b2 <- b * 10
s <- sqrt(b2)
round(s, 1)
```

`merge` *Merge SpatRaster or SpatExtent objects, or a SpatVector with a data.frame*

Description

Merge SpatRasters to form a new SpatRaster object with a larger spatial extent. If objects overlap, the values get priority in the same order as the arguments. See `classify` to merge a `SpatRaster` and a `data.frame`. You can also merge SpatExtent objects.

There is a also a method for merging SpatVector with a data.frame.

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster'
merge(x, y, ..., filename="", overwrite=FALSE, wopt=list())
## S4 method for signature 'SpatExtent,SpatExtent'
merge(x, y, ...)

## S4 method for signature 'SpatVector,data.frame'
merge(x, y, ...)
```

Arguments

`x`	SpatRaster or SpatExtent object
`y`	object of same class as `x`
`...`	if `x` is a SpatRaster or SpatVector: additional objects of the same class as `x`. If `x` is a SpatVector, the same arguments as in `merge`
`filename`	character. Output filename. Optional
`overwrite`	logical. If `TRUE`, `filename` is overwritten
`wopt`	list. Options for writing files as in `writeRaster`

Details

The SpatRaster objects must have the same origin and spatial resolution. In areas where the SpatRaster objects overlap, the values of the SpatRaster that is last in the sequence of arguments will be retained.

Value

SpatRaster or SpatExtent

Examples

```
x <- rast(xmin=-110, xmax=-50, ymin=40, ymax=70, ncols=60, nrows=30)
y <- rast(xmin=-80, xmax=-20, ymax=60, ymin=30)
res(y) <- res(x)
values(x) <- 1:ncell(x)
values(y) <- 1:ncell(y)
mr <- merge(x, y)
plot(mr)
mr <- merge(y, x)

# if you have many SpatRaster objects in a list
# you can use do.call:
s <- list(x, y)
# add arguments such as filename
s$filename <- ""
m <- do.call(merge, s)

##
# SpatVector with data.frame
f <- system.file("exdata/lux.shp", package="terra")
p <- vect(f)
dfr <- data.frame(District=p$NAME_1, Canton=p$NAME_2, Value=round(runif(length(p), 100, 1000)))
dfr <- dfr[1:5, ]
pm <- merge(p, dfr, all.x=TRUE, by.x=c('NAME_1', 'NAME_2'), by.y=c('District', 'Canton'))
pm
values(pm)
```

---

modal *modal value*

Description

Compute the mode for each cell across the layers of a SpatRaster. The mode, or modal value, is the most frequent value in a set of values.

Usage

```
## S4 method for signature 'SpatRaster'
modal(x, ..., ties="first", na.rm=FALSE, filename="", overwrite=FALSE, wopt=list())
```

Arguments

x	SpatRaster
...	additional argument of the same type as x or numeric
ties	character. Indicates how to treat ties. Either "random", "lowest", "highest", "first", or "NA"
na.rm	logical. If TRUE, NA values are ignored. If FALSE, NA is returned if x has any NA values
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster

Value

SpatRaster

Examples

```
r <- rast(system.file("exdata/logo.tif", package="terra"))
r <- c(r/2, r, r*2)
m <- modal(r)
```

---

names *Names of Spat objects*

Description

Get or set the names of the layers of a SpatRaster or the attributes of a SpatVector.

Usage

```
## S4 method for signature 'SpatRaster'
names(x)
## S4 replacement method for signature 'SpatRaster'
names(x)<-value
## S4 method for signature 'SpatVector'
names(x)
## S4 replacement method for signature 'SpatVector'
names(x)<-value
```

Arguments

x	SpatRaster or SpatVector
value	character (vector)

Value

character

Examples

```
s <- rast(ncols=5, nrows=5, nlyr=3)
nlyr(s)
names(s)
names(s) <- c("a", "b", "c")
names(s)

# space is not valid
names(s)[2] <- "hello world"
names(s)

# two invalid names
names(s) <- c("a", "  a  ", "3")
names(s)

# SpatVector names
f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
names(v)
names(v) <- paste0(substr(names(v), 1, 2), "_", 1:ncol(v))
names(v)
```

---

overlay *Overlay SpatRaster objects*

Description

Create a new SpatRaster object, by combining two or more SpatRaster objects.

You should supply a function `fun` to determine how the SpatRasters are combined. The number of arguments in the function must match the number of SpatRaster objects (or take any number). For example, if you combine two SpatRaster objects you could use multiply: `fun=function(x,y){return(x*y)}` percentage: `fun=function(x,y){return(100 * x / y)}`. If you combine three layers you could use `fun=function(x,y,z){return((x + y) * z)}`

Note that the function work for vectors (not only for single numbers). That is, it must return the same number of elements as its input vectors.

The function must take each layer as an argument.

Use app for functions such as `sum`, that take `n` arguments.

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster'
overlay(x, y, fun, ..., filename="", overwrite=FALSE, wopt=list())
```

Arguments

x	SpatRaster
y	SpatRaster
fun	function to be applied. The first two arguments the function should expect must be numerical vectors
...	additional arguments to be passed to fun
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster

Value

SpatRaster

See Also

app,math

Examples

```
r1  <- rast(ncol=10, nrow=10)
r2  <- rast(ncol=10, nrow=10)
values(r1) <- 1:ncell(r1)
values(r2) <- 10
x <- overlay(r1, r2, fun=function(x,y){return(x/y)})
```

---

pack *pack a Spat* object*

Description

Pack a Spat* object so that it can be saved as an R object to disk, or passed over a connection that serializes (e.g. using a computer cluster).

Usage

```
## S4 method for signature 'SpatRaster'
pack(x, ...)
## S4 method for signature 'SpatVector'
pack(x, ...)
```

Arguments

x	SpatVector or SpatRaster
...	additional arguments. None implemented

Value

Packed* object

`pairs` *Pairs plot (matrix of scatterplots)*

Description

Pair plots of layers in a SpatRaster. This is a wrapper around graphics function `pairs`.

Usage

```
## S4 method for signature 'SpatRaster'
pairs(x, hist=TRUE, cor=TRUE, use="pairwise.complete.obs", maxcells=100000, ...)
```

Arguments

`x`	SpatRaster
`hist`	logical. If TRUE a histogram of the values is shown on the diagonal
`cor`	logical. If TRUE the correlation coefficient is shown in the upper panels
`use`	argument passed to the `cor` function
`maxcells`	integer. Number of pixels to sample from each layer of large Raster objects
`...`	additional arguments (graphical parameters)

See Also

`boxplot`,`hist`

Examples

```
r <-rast(system.file("exdata/test.tif", package="terra"))
s <- c(r*1, 1/r, sqrt(r))
pairs(s)

# to make indvidual histograms:
hist(r)
# or scatter plots:
plot(r, 1/r)
```

persp *Perspective plot*

Description

Perspective plot of a SpatRaster. This is an implementation of a generic function in the graphics package.

Usage

```
## S4 method for signature 'SpatRaster'
persp(x, maxcells=100000, ...)
```

Arguments

x	SpatRaster. Only the first layer is used
maxcells	integer > 0. Maximum number of cells to use for the plot. If `maxpixels < ncell(x)`, `spatSample(method="regular")` is used before plotting
...	Any argument that can be passed to persp (graphics package)

See Also

persp, contour, plot

Examples

```
r <- rast(system.file("exdata/test.tif", package="terra"))
persp(r)
```

plot *Plot a SpatRaster or SpatVector*

Description

Plot (that is, make a map of) the values of a SpatRaster or SpatVector

Usage

```
## S4 method for signature 'SpatRaster,numeric'
plot(x, y, col, maxcell=100000, leg.mar=NULL,
   leg.levels=5, leg.shrink=c(0,0), leg.main=NULL, leg.main.cex=1, leg.ext=NULL,
   digits, useRaster=TRUE, zlim, xlab="", ylab="", axes=TRUE, add=FALSE, ...)

## S4 method for signature 'SpatRaster,missing'
```

```
plot(x, y, maxcell=100000, nc, nr, main,
   maxnl=16, add=FALSE, ...)
## S4 method for signature 'SpatRaster,SpatRaster'
plot(x, y, maxcell=100000, nc, nr,
   maxnl=16, gridded=FALSE, ncol=25, nrow=25, ...)

## S4 method for signature 'SpatVector,missing'
plot(x, y, col=NULL, xlab="", ylab="",
   axes=TRUE, add=FALSE, border="black", ...)

## S4 method for signature 'SpatVector,character'
plot(x, y, col=topo.colors(100), xlab="", ylab="",
   axes=TRUE, add=FALSE, leg.ext=NULL, leg.type=NULL, leg.levels=5, digits, ...)

## S4 method for signature 'SpatExtent,missing'
plot(x, y, col=NULL, xlab="", ylab="", axes=TRUE, ...)

## S4 method for signature 'SpatRaster'
image(x, y, maxcell=100000, xlab="", ylab="", ...)

## S4 method for signature 'SpatVector'
points(x, col=NULL, ...)
## S4 method for signature 'SpatVector'
lines(x, col=NULL, ...)
## S4 method for signature 'SpatExtent'
points(x, col=NULL, ...)
## S4 method for signature 'SpatExtent'
lines(x, col=NULL, ...)
```

Arguments

x	SpatRaster or SpatVector
y	missing or positive integer indicating the layer(s) to be plotted, or the name of the layer to be mapped
col	character. Colors
maxcell	positive integer. Maximum number of cells to use for the plot
leg.mar	positive integer. Size of margin for the legend
leg.levels	positive integer. Number of labels on the legend
leg.shrink	numeric. Relative shrinkage of legend from top and bottom
leg.main	character. Legend title
leg.main.cex	character. Legend title size
leg.ext	SpatExtent or numeric vector indicating the area where to plot the legend. The vector should have a lenght of 4 (xmin, xmax, ymin, ymax)
leg.type	character. Legend type
digits	integer >= 0

zlim	range of allowed values
xlab	character
ylab	character
axes	logical
useRaster	logical; If TRUE plotting is faster because a bitmap raster is used to plot the image instead of polygons. See image
nc	positive integer. Optional. The number of columns to divide the plotting device in (when plotting multiple layers)
nr	positive integer. Optional. The number of rows to divide the plotting device in (when plotting multiple layers)
maxnl	postive integer. Maximum number of layers to plot (for a multi-layer object)
main	character. Main plot title
gridded	logical. If TRUE the scatterplot is gridded (counts by cells)
ncol	positive integer. Number of columns for gridding
nrow	positive integer. Number of rows for gridding
add	logical. If TRUE add the object to the current plot (for points, lines, and polygons this is an alternative to the lines or points methods
border	character to select the border (polygon outline) color
...	additional graphical arguments

Examples

```
f <- system.file("exdata/test.tif", package="terra")
r <- rast(f)
plot(r)

d <- (r > 400) + (r > 600)
plot(d, leg.main="Title")
plot(d, leg.ext=c(178000,178200,332800,333250),
leg.main=c("Title     \n\n"))

# character labels
x <- round(r/1000)
x <- as.factor(x)
levels(x) <- c("earth", "wind", "fire")
plot(x)


f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
r <- rast(v)
values(r) <- 1:ncell(r)
plot(r)
lines(v)
points(v)
```

plotRGB *Red-Green-Blue plot of a multi-layered SpatRaster*

Description

Make a Red-Green-Blue plot based on three layers in a SpatRaster. The layers (sometimes referred to as "bands" because they may represent different bandwidths in the electromagnetic spectrum) are combined such that they represent the red, green and blue channel. This function can be used to make "true" (or "false") color images from Landsat and other multi-spectral satellite images.

Usage

```
## S4 method for signature 'SpatRaster'
plotRGB(x, r=1, g=2, b=3, scale, maxcell=500000, stretch=NULL,
ext=NULL, interpolate=FALSE, colNA="white", alpha, bgalpha, addfun=NULL, zlim=NULL,
zlimcol=NULL, axes=FALSE, xlab="", ylab="", asp=NULL, add=FALSE, margins=FALSE, ...)
```

Arguments

x	SpatRaster
r	integer. Index of the Red channel, between 1 and nlyr(x)
g	integer. Index of the Green channel, between 1 and nlyr(x)
b	integer. Index of the Blue channel, between 1 and nlyr(x)
scale	integer. Maximum (possible) value in the three channels. Defaults to 255 or to the maximum value of x if that is known and larger than 255
maxcell	integer > 0. Maximum number of pixels to use
stretch	character. Option to stretch the values to increase the contrast of the image: "lin" or "hist"
ext	An Extent object to zoom in to a region of interest (see drawExtent)
interpolate	logical. If TRUE, interpolate the image when drawing
colNA	color for the background (NA values)
alpha	transparency. Integer between 0 (transparent) and 255 (opaque)
bgalpha	Background transparency. Integer between 0 (transparent) and 255 (opaque)
addfun	Function to add additional items such as points or polygons to the plot (map). See plot
zlim	numeric vector of length 2. Range of values to plot (optional)
zlimcol	If NULL the values outside the range of zlim get the color of the extremes of the range. If zlimcol has any other value, the values outside the zlim range get the color of NA values (see colNA)
axes	logical. If TRUE axes are drawn (and arguments such as main="title" will be honored)
xlab	character. Label of x-axis

ylab	character. Label of y-axis
asp	numeric. Aspect (ratio of x and y. If NULL, and appropriate value is computed to match data for the longitude/latitude coordinate reference system, and 1 for planar coordinate reference systems
add	logical. If TRUE add values to current plot
margins	logical. If TRUE standard whitespace margins are used. If FALSE, graphics::par(plt=c(0,1,0,1)) is used
...	graphical parameters as in plot or rasterImage

See Also

plot

Examples

```
b <- rast(system.file("exdata/logo.tif", package="terra"))
plotRGB(b)
plotRGB(b, 3, 2, 1)
plotRGB(b, 3, 2, 1, stretch='hist')
```

---

predict *Spatial model predictions*

Description

Make a SpatRaster object with predictions from a fitted model object (for example, obtained with glm or randomForest). The first argument is a SpatRaster object with the predictor variables. The names in the Raster object should exactly match those expected by the model. Any regression like model for which a predict method has been implemented (or can be implemented) can be used.

This approach of using model predictions is commonly used in remote sensing (for the classification of satellite images) and in ecology, for species distribution modeling.

Usage

```
## S4 method for signature 'SpatRaster'
predict(object, model, fun=predict, ..., factors=NULL,
   const=NULL, na.rm=FALSE, index=NULL, filename="", overwrite=FALSE, wopt=list())
```

Arguments

object	SpatRaster
model	fitted model of any class that has a "predict" method (or for which you can supply a similar method as fun argument. E.g. glm, gam, or randomForest
fun	function. Default value is predict, but can be replaced with e.g. predict.se (depending on the type of model), or your own custom function

...	additional arguments for fun
const	data.frame. Can be used to add a constant value as a predictor variable so that you do not need to make a SpatRaster layer for it
factors	list with levels for factor variables. The list elements should be named with names that correspond to names in object such that they can be matched. This argument may be omitted for standard models such as "glm" as the predict function will extract the levels from the model object, but it is necessary in some other cases (e.g. cforest models from the party package)
na.rm	logical. If TRUE, cells with NA values in the predictors are removed from the computation. This option prevents errors with models that cannot handle NA values. In most other cases this will not affect the output. An exception is when predicting with a model that returns predicted values even if some (or all!) variables are NA
index	integer. To select subset of output variables
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster

Value

SpatRaster

Examples

```
logo <- rast(system.file("exdata/logo.tif", package="terra"))
names(logo) <- c("red", "green", "blue")
p <- matrix(c(48, 48, 48, 53, 50, 46, 54, 70, 84, 85, 74, 84, 95, 85,
   66, 42, 26, 4, 19, 17, 7, 14, 26, 29, 39, 45, 51, 56, 46, 38, 31,
   22, 34, 60, 70, 73, 63, 46, 43, 28), ncol=2)

a <- matrix(c(22, 33, 64, 85, 92, 94, 59, 27, 30, 64, 60, 33, 31, 9,
   99, 67, 15, 5, 4, 30, 8, 37, 42, 27, 19, 69, 60, 73, 3, 5, 21,
   37, 52, 70, 74, 9, 13, 4, 17, 47), ncol=2)

xy <- rbind(cbind(1, p), cbind(0, a))

# extract predictor values for points
e <- extract(logo, xy[,2:3])

# combine with response
v <- data.frame(cbind(pa=xy[,1], e))

#build a model, here with glm
model <- glm(formula=pa~., data=v)

#predict to a raster
r1 <- predict(logo, model)

plot(r1)
```

```
points(p, bg='blue', pch=21)
points(a, bg='red', pch=21)

# logistic regression
model <- glm(formula=pa~., data=v, family="binomial")
r1log <- predict(logo, model, type="response")

# use a modified function to get the probability and standard error
# from the glm model. The values returned by "predict" are in a list,
# and this list needs to be transformed to a matrix

predfun <- function(model, data) {
  v <- predict(model, data, se.fit=TRUE)
  cbind(p=as.vector(v$fit), se=as.vector(v$se.fit))
}

r2 <- predict(logo, model, fun=predfun)

# principal components of a SpatRaster
# here using sampling to simulate an object too large
# to feed all its values to prcomp

sr <- values(spatSample(logo, 100, as.raster=TRUE))
pca <- prcomp(sr)

x <- predict(logo, pca)
plot(x)
```

`project` *Change the coordinate reference system*

Description

Change the coordinate reference system ("project") of a SpatVector or SpatRaster. For a SpatRaster, you have more control with the `warp` method.

Usage

```
## S4 method for signature 'SpatVector'
project(x, crs, ...)
## S4 method for signature 'SpatRaster'
project(x, crs, method="bilinear", filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

`x`	SpatVector
`crs`	character. Description of output coordinate reference system in PROJ.4 notation

method	character. Method used to compute values. "bilinear" for bilinear interpolation, or "ngb" for "nearest neighbor"
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatVector or SpatRaster

See Also

warp

Examples

```
f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
crs <- "+proj=moll +lon_0=0 +x_0=0 +y_0=0 +ellps=WGS84 +datum=WGS84"
p <- project(v, crs)
p

a <- rast(ncol=40, nrow=40, xmin=-110, xmax=-90, ymin=40, ymax=60, crs="+proj=longlat +datum=WGS84")
values(a) = 1:ncell(a)
crs="+proj=lcc +lat_1=48 +lat_2=33 +lon_0=-100 +datum=WGS84"
b <- project(a, crs)
```

---

quantile *SpatRaster local quantiles*

Description

Compute quantiles for each cell across the layers of a SpatRaster

Usage

```
## S4 method for signature 'SpatRaster'
quantile(x, probs=seq(0, 1, 0.25), na.rm=FALSE,
    filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
probs	numeric vector of probabilities with values in [0,1]
na.rm	logical. If TRUE, NA's are removed from x before the quantiles are computed
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments, none implemented

Value

A vector of quantiles

See Also

app, local

Examples

```
r <- rast(system.file("exdata/logo.tif", package="terra"))
r <- c(r/2, r, r*2)
q <- quantile(r)
q

# same as (but faster than)
qa <- app(r, quantile)
```

---

range *Get or compute the minimum and maximum cell values*

Description

The minimum and maximum value of a SpatRaster are returned or computed (from a file on disk if necessary) and stored in the object.

Usage

```
## S4 method for signature 'SpatRaster'
minmax(x)
## S4 method for signature 'SpatRaster'
setMinMax(x)
```

Arguments

x	SpatRaster

Value

setMinMax: nothing. Used for the side-effect of computing the minimum and maximum values of a SpatRaster

minmax: numeric matrix of minimum and maximum cell values by layer

Examples

```
r <- rast(system.file("exdata/test.tif", package="terra"))
minmax(r)
```

---

rast *Create a SpatRaster*

Description

Methods to create a SpatRaster. These objects can be created from scratch or from a file.

A SpatRaster represents a spatially referenced surface divided into three dimensional cells (rows, columns, and layers).

When a SpatRaster is created from a file, it does (initially) not contain any cell (pixel) values in memory (RAM), it only has the parameters that describe the SpatRaster. You can access cell-values with values.

Usage

```
## S4 method for signature 'character'
rast(x, ...)

## S4 method for signature 'missing'
rast(x, nrows=180, ncols=360, nlyrs=1, xmin=-180, xmax=180,
                                ymin=-90, ymax=90, crs, extent, resolution, ...)

## S4 method for signature 'SpatRaster'
rast(x, nlyrs=nlyr(x), ...)

## S4 method for signature 'matrix'
rast(x, crs=NA, type="", ...)

## S4 method for signature 'SpatVector'
rast(x, nrows=10, ncols=10, nlyrs=1, ...)

## S4 method for signature 'SpatExtent'
rast(x, nrows=10, ncols=10, nlyrs=1, crs="", ...)

## S4 method for signature 'Raster'
rast(x, ...)
```

Arguments

x	filename (character), Extent, SpatRaster, matrix, array, or Raster* object (from the "raster" package). Can also be missing
nrows	positive integer. Number of rows
ncols	positive integer. Number of columns
nlyrs	positive integer. Number of layers
xmin	minimum x coordinate (left border)
xmax	maximum x coordinate (right border)
ymin	minimum y coordinate (bottom border)
ymax	maximum y coordinate (top border)
extent	object of class SpatExtent. If present, the arguments xmin, xmax, ymin and ymax are ignored
crs	character. PROJ.4 type description of a Coordinate Reference System (map projection). If this argument is missing, and the x coordinates are within -360 .. 360 and the y coordinates are within -90 .. 90, "+proj=longlat +datum=WGS84" is used
resolution	numeric vector of length 1 or 2 to set the resolution (see res). If this argument is used, arguments ncol and nrow are ignored
type	character. If the value is not "xyz", the raster has the same number of rows and colums as the matrix. If the value is "xyz", the matrix must have at least two columns, the first with x (or longitude) and the second with y (or latitude) coordinates that represent the centers of raster cells. The additional columns are the values associated with the raster cells.
...	additional arguments. None implemented

Value

SpatRaster

Examples

```
# Create a SpatRaster from scratch
x <- rast(nrow=108, ncol=21, xmin=0, xmax=10)

# Create a SpatRaster from a file
f <- system.file("exdata/test.tif", package="terra")
r <- rast(f)

s <- rast(system.file("exdata/logo.tif", package="terra"))

# Create a skeleton with no associated cell values
rast(s)
```

rasterize *Rasterize vector data*

Description

Transfer vector data to a raster

Usage

```
## S4 method for signature 'SpatVector,SpatRaster'
rasterize(x, y, field=1:nrow(x), background=NA,
    update=FALSE, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatVector
y	SpatRaster
field	numeric. The values to be rasterized. Either a single number, or a vector with the same length as x
background	numeric. Value for cells that are not covered by a polygon
update	logical. If TRUE the value in x are used except for cells covered by y
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

Examples

```
f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
r <- rast(v, ncol=75, nrow=100)
x <- rasterize(v, r)

plot(x)
lines(v)
```

`read and write` *Read from, or write to, file*

Description

Methods to read from or write chunks of values to or from a file. These are low level methods for programmers. Use writeRaster if you want to save an entire SpatRaster to file in one step. It is much easier to use.

To write chunks, begin by opening a file with `writeStart`, then write values to it in chunks. When writing is done close the file with `writeStop`.

Usage

```
## S4 method for signature 'SpatRaster'
readStart(x, ...)
## S4 method for signature 'SpatRaster'
readStop(x)
## S4 method for signature 'SpatRaster'
readValues(x, row=1, nrows=nrow(x), col=1, ncols=ncol(x), mat=FALSE, dataframe=FALSE, ...)

## S4 method for signature 'SpatRaster,character'
writeStart(x, filename="", overwrite=FALSE, wopt=list(), ...)
## S4 method for signature 'SpatRaster'
writeStop(x)
## S4 method for signature 'SpatRaster,vector'
writeValues(x, v, start)
```

Arguments

`x`	SpatRaster
`filename`	Character. Output filename. Optional
`v`	vector with cell values to be written
`start`	integer. Row number (counting starts at 1) from where to start writing `v`
`row`	positive integer. Row number to start from, should be between 1 and nrow(x)
`nrows`	positive integer. How many rows? Default is 1
`col`	positive integer. Column number to start from, should be between 1 and ncol(x)
`ncols`	positive integer. How many columns? Default is the number of columns left after the start column
`mat`	logical. If `TRUE`, values are returned as a matrix instead of as a vector, except when dataframe is `TRUE`
`dataframe`	logical. If `TRUE`, values are returned as a data.frame instead of as a vector (also if matrix is `TRUE`)
`overwrite`	logical. If `TRUE`, `filename` is overwritten
`wopt`	list. Options for writing files as in `writeRaster`
`...`	additional arguments. None implemented

Value

`readValues` returns a vector, matrix, or data.frame

`writeStart` returns a list that can be used for processing the file in chunks.

The other methods invisibly return a logical value indicating whether they were succesful or not. Their purpose is the side-effect of opening or closing files.

---

rotate *Rotate a SpatRaster along longitude*

Description

Rotate a SpatRaster that has x coordinates (longitude) from 0 to 360, to standard coordinates between -180 and 180 degrees (or vice-versa). Longitude between 0 and 360 is frequently used in global climate models.

Usage

```
## S4 method for signature 'SpatRaster'
rotate(x, left=TRUE, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

`x`	Raster* object
`left`	logical. If `TRUE`, rotate to the left, else to the right
`filename`	character. Output filename. Optional
`overwrite`	logical. If `TRUE`, `filename` is overwritten
`wopt`	list. Options for writing files as in `writeRaster`
`...`	additional arguments. None implemented

Value

SpatRaster

Examples

```
x <- rast(nrow=9, ncol=18, nl=3, xmin=0, xmax=360)
v <- rep(as.vector(t(matrix(1:ncell(x), nrow=9, ncol=18))), 3)
values(x) <- v
z <- rotate(x)
```

`select` *Geometric subsetting*

Description

Geometrically subset SpatRaster or SpatVector (to be done) by drawing on a plot (map).

Usage

```
## S4 method for signature 'SpatRaster'
select(x, ...)
```

Arguments

`x`	SpatRaster or SpatVector
`...`	additional arguments passed on to `draw`

Value

SpatRaster or SpatVector

See Also

`click`,`crop`

Examples

```
# select a subset of a RasterLayer
r <- rast(nrow=10, ncol=10)
values(r) <- 1:ncell(r)
plot(r)
s <- select(r) # now click on the map twice

# plot the selection on a new canvas:
x11()
plot(s)
```

shift *Shift*

Description

Shift the location of a SpatRaster or SpatVector object

Usage

```
## S4 method for signature 'SpatRaster'
shift(x, dx=0, dy=0, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster or SpatVector
dx	numeric. The shift in horizontal direction
dy	numeric. The shift in vertical direction
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

Same object type as x

See Also

flip, rotate

Examples

```
r <- rast(xmn=0, ymn=0, xmx=1, ymx=1)
r <- shift(r, dx=1, dy=-1)
```

slope *Compute slopes*

Description

Compute slopes from elevation data. The elevation values should be in map units (typically meter) for projected (planar) raster data. They should be in meters when the coordinate reference system (CRS) is longitude/latitude.

Usage

```
## S4 method for signature 'SpatRaster'
slope(x, neighbors=8, unit="degrees", filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	Single layer SpatRaster with elevation values. Values should have the same unit as the map units, or in meters when the crs is longitude/latitude
unit	Character. 'degrees' or 'tangent'
neighbors	Integer. Indicating how many neighboring cells to use to compute slope for any cell. Either 8 (queen case) or 4 (rook case)
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Details

When neighbors=4 slope and aspect are computed according to Fleming and Hoffer (1979) and Ritter (1987). When neigbors=8, slope and aspect are computed according to Horn (1981). The Horn algorithm may be best for rough surfaces, and the Fleming and Hoffer algorithm may be better for smoother surfaces (Jones, 1997; Burrough and McDonnell, 1998).

If slope = 0, aspect is set to 0.5*pi radians (or 90 degrees if unit="degrees"). When computing slope or aspect, the crs of SpatRaster x must be known (may not be NA), to be able to safely differentiate between planar and longitude/latitude data.

References

Burrough, P., and R.A. McDonnell, 1998. Principles of Geographical Information Systems. Oxford University Press.

Fleming, M.D. and Hoffer, R.M., 1979. Machine processing of landsat MSS data and DMA topographic data for forest cover type mapping. LARS Technical Report 062879. Laboratory for Applications of Remote Sensing, Purdue University, West Lafayette, Indiana.

Horn, B.K.P., 1981. Hill shading and the reflectance map. Proceedings of the IEEE 69:14-47

Jones, K.H., 1998. A comparison of algorithms used to compute hill slope as a property of the DEM. Computers & Geosciences 24: 315-323

Ritter, P., 1987. A vector-based slope and aspect generation algorithm. Photogrammetric Engineering and Remote Sensing 53: 1109-1111

`sources` *Data sources of a SpatRaster*

Description

Get the data sources of a SpatRaster and the number of layers by source. Sources are either files (or similar resources) or "", meaning that they are in memory. You an use hasValues to check if a in-memory layer these actualy have values.

Usage

```
## S4 method for signature 'SpatRaster'
sources(x, ...)
## S4 method for signature 'SpatRaster'
hasValues(x, ...)
```

Arguments

`x`	SpatRaster
`...`	additional arguments. None implemented

Value

data.frame with the source names and the number of layers by source

Examples

```
f <- system.file("exdata/test.tif", package="terra")
r <- rast(f)
s <- rast(r)
values(s) <= 1:ncell(s)
rs <- c(r,r,s,r)
sources(rs)
hasValues(r)
x <- rast()
hasValues(x)
```

SpatDataFrame-class *C++ classes*

Description

C++ classes and methods. Not for end-users.

SpatExtent-class *Class "SpatExtent"*

Description

Objects of class SpatExtent are used to define the spatial extent (extremes) of objects of the SpatRaster class.

Objects from the Class

You can use the `ext` function to create SpatExtent objects, or to extract them from SpatRaster objects.

Slots

`ptr:` pointer to the C++ class

Methods

show display values of a SpatExtent object

Examples

```
e <- ext(-180, 180, -90, 90)
e
```

SpatOptions-class *Inspect of set general options for "terra"*

Description

Class and methods for showing and setting options.

Usage

```
terraOptions(...)
```

Arguments

...	additional arguments. Can be use to set options (see examples). When empty, the current options are shown

Details

The following options are available.

memfrac - value between 0.1 and 0.8. The fraction of RAM that may be used by the program.

tempdir - directory where temporary files are written. The default what is returned by tempdir().

datatype - default data type. See writeRaster

filetype - default file type. See writeRaster

progress - non-negative integer. A progress bar is shown if the number of chunks in which the data is processed is larger than this number. No progress bar is shown if the value is zero

Examples

```
terraOptions()
terraOptions(memfrac=0.5, tempdir = "c:/temp")
terraOptions(progress=10)
terraOptions()
```

SpatRaster-class *SpatRaster class*

Description

A raster is a database organized as a rectangular grid that is sub-divided into rectangular cells of equal area (in terms of the units of the coordinate reference system) and layers.

An object of the SpatRaster class can point to one or more files on disk that holds the values of the raster cells, or hold these values in memory. Or it can not have any associated values at all.

Objects can be created with the rast function.

This package defines the SpatRaster "S4 class" to manipulate such data. The R object only contains a pointer to the C++ class "Rcpp_SpatRaster".

Objects from the Class

Objects can be created with the `rast` function.

Slots

Slots for SpatRaster objects

`ptr:` pointer to the C++ class

Examples

```
rast()
```

`spatSample` *Take a regular sample*

Description

Take a regular sample of a SpatRaster. Either get cell values, or a new SpatRaster with the same extent, but fewer cells.

Usage

```
## S4 method for signature 'SpatRaster'
spatSample(x, size, method="regular", replace=FALSE, as.raster=FALSE, ...)
```

Arguments

`x`	SpatRaster
`size`	numeric. The sample size
`method`	character. Should be "regular" or "random"
`replace`	logical. If `TRUE`, sampling is with replacement (if `method="random"`
`as.raster`	logical. If `TRUE` and `method="regular"`, a SpatRaster is returned
`...`	additional arguments. None implemented

Value

numeric or SpatRaster

Examples

```
f <- system.file("exdata/test.tif", package="terra")
r <- rast(f)
s <- spatSample(r, 10, as.raster=TRUE)
spatSample(r, 10)
spatSample(r, 10, "random")
```

SpatVector-class *Class "SpatVector"*

Description

Objects of class SpatVector.

Objects from the Class

You can use the vect method to create SpatVector objects.

Slots

ptr: pointer to the C++ class

Methods

show display values of a SpatVector

subset *Subset of a SpatRaster*

Description

Select a subset of layers from a SpatRaster.

Usage

```
## S4 method for signature 'SpatRaster'
subset(x, subset, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
subset	integer or character. Should indicate the layers (represented as integer or by their names)
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

Examples

```
s <- rast(system.file("exdata/logo.tif", package="terra"))
subset(s, 2:3)
subset(s, c(3,2,3,1))
#equivalent to
s[[ c(3,2,3,1) ]]
```

---

subset-vector *Subset of a SpatVector*

Description

Select a subset of variables or records from a SpatVector.

Usage

```
## S4 method for signature 'SpatVector'
subset(x, subset, drop=FALSE, ...)
```

Arguments

x	SpatVector
subset	logical expression indicating elements or rows to keep: missing values are taken as false
drop	logical. If TRUE, the geometries will be dropped, and a data.frame is returned
...	additional arguments. None implemented

Value

SpatVector or, if drop=TRUE, a data.frame.

Examples

```
f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
v[2:3,]
v[,2:3]
subset(v, v$NAME_1 == "Diekirch")
```

tapp *Apply a function to subsets of layers of a SpatRaster*

Description

Apply a function to subsets of layers of a SpatRaster (similar to tapply and link[stats]{aggregate}). The layers are combined are combind based on the index.

The function used should return a single value, and the number of layers in the output SpatRaster equals the number of unique values in index.

For example, if you have a SpatRaster with 6 layers, you can use index=c(1,1,1,2,2,2) and fun=sum. This will return a SpatRaster with two layers. The first layer is the sum of the first three layers in the input SpatRaster, and the second layer is the sum of the last three layers in the input SpatRaster. index are recycled such that index=c(1,2) would also return a SpatRaster with two layers (one based on the odd layers (1,3,5), the other based on the even layers (2,4,6)).

See app or Summary-methods if you want to use a more efficient function that returns multiple layers based on **all** layers in the SpatRaster object.

Usage

```
## S4 method for signature 'SpatRaster'
tapp(x, index, fun, ..., filename="", overwrite=FALSE, wopt=list())
```

Arguments

x	SpatRaster
index	factor or numeric (integer). Vector of length nlayers(x) (shorter vectors are recycled) grouping the input layers
fun	function to be applied
...	additional arguments passed to fun
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster

Value

SpatRaster

See Also

app, Summary-methods

Examples

```
r <- rast(ncol=10, nrow=10)
values(r) <- 1:ncell(r)
s <- c(r, r, r, r, r, r)
s <- s * 1:6
b1 <- tapp(s, index=c(1,1,1,2,2,2), fun=sum)
b1
b2 <- tapp(s, c(1,2,3,1,2,3), fun=sum)
b2
```

`text` *Add labels to a map*

Description

Plots labels, that is a textual (rather than color) representation of values, on top an existing plot (map).

Usage

```
## S4 method for signature 'SpatRaster'
text(x, labels, digits=0, halo=FALSE, ...)

## S4 method for signature 'SpatVector'
text(x, labels, halo=FALSE, ...)
```

Arguments

`x`	SpatRaster or SpatVector
`labels`	character. Optional. Vector of labels with `length(x)` or a variable name from `names(x)`
`digits`	integer. how many digits should be used?
`halo`	logical. If `TRUE` a "halo" is printed around the text. If `TRUE`, additional arguments `hc="white"` and `hw=0.1` can be modified to set the colour and width of the halo
`...`	additional arguments to pass to graphics function `text`

See Also

`text`,`plot`

Examples

```
r <- rast(nrows=4, ncols=4)
values(r) <- 1:ncell(r)
plot(r)
text(r)
```

```
plot(r)
text(r, halo=TRUE, hc="blue", col="white", hw=0.2)

plot(r, col=rainbow(16))
text(r, col=c("black", "white"), vfont=c("sans serif", "bold"), cex=2)
```

`tmpFiles` *Temporary files*

Description

List and optionally remove temporary files created by the terra package. These files are created when an output SpatRaster may be too large to store in memory (RAM). This can happen when no filename is provided to a function and in functions where you cannot provide a filename.

Temporary files are automatically removed at the end of each session that ends normally. You can use `tmpFiles` to see the files in the current sessions, from other (perhaps old) sessions, and remove all the temporary files.

Usage

```
tmpFiles(old=FALSE, remove=FALSE)
orphanTmpFiles()
staleTmpFiles()
```

Arguments

`old`	logical. If `TRUE`, temporary files from other (possibly aborted) sessions are included
`remove`	logical. If `TRUE`, temporary files are removed

Value

character

See Also

`rasterOptions`, `tempfile`

Examples

```
tmpFiles()
```

transpose *Transpose*

Description

Transpose a SpatRaster

Usage

```
## S4 method for signature 'SpatRaster'
t(x)
## S4 method for signature 'SpatRaster'
transpose(x, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

See Also

flip,rotate

Examples

```
r <- rast(nrow=18, ncol=36)
values(r) <- 1:ncell(r)
tr1 <- t(r)
tr2 <- transpose(r)
ttr <- transpose(tr2)
```

trim *Trim a SpatRaster*

Description

Trim (shrink) a SpatRaster by removing outer rows and columns that are NA.

Usage

```
## S4 method for signature 'SpatRaster'
trim(x, padding=0, filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
padding	integer. Number of outer rows/columns to keep
filename	character. Output filename. Optional
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

Examples

```
r <- rast(ncol=10, nrow=10, xmin=0,xmax=10,ymin=0,ymax=10)
v <- rep(NA, ncell(r))
v[c(12,34,69)] <- 1:3
values(r) <- v
s <- trim(r)
```

unique *Unique values*

Description

This function returns the unique values in a SpatRaster.

Usage

```
## S4 method for signature 'SpatRaster'
unique(x, incomparables=FALSE, ...)
```

Arguments

x	SpatRaster
incomparables	logical. If TRUE, the unique values are determined for all layers together, and the result is a matrix. If FALSE, each layer is evaluated separately, and a list is returned.
...	additional arguments. none implemented

Value

vector or matrix

Examples

```
r <- rast(ncol=5, nrow=5)
values(r) <- rep(1:5, each=5)
unique(r)
s <- c(r, round(r/3))
unique(s)
unique(s,TRUE)
```

---

values *Get cell values*

Description

values returns all cell values of a SpatRaster (a matrix), or all the attributes of a SpatVector (a data.frame).

Usage

```
## S4 method for signature 'SpatRaster'
values(x, mat=TRUE, ...)
## S4 replacement method for signature 'SpatRaster,ANY'
values(x, ...)<-value
## S4 method for signature 'SpatVector'
values(x, ...)
## S4 replacement method for signature 'SpatVector,data.frame'
values(x, ...)<-value
```

Arguments

x	SpatRaster or SpatVector
mat	logical. If TRUE, values are returned as a matrix instead of as a vector
value	For SpatRaster: matrix or numeric, the length must match the total number of cells (ncell(x) * nlyr(x)), or be a single value. For SpatVector: data.frame
...	additional arguments. none implemented

Details

If x is a SpatRaster:

If matrix=TRUE, a matrix is returned in which the values of each layer are represented by a column (with ncell(x) rows). The values per layer are in cell-order, that is, from top-left, to top-right and then down by row. Use as.matrix for an alternative matrix representation where the number of rows and columns matches that of x, if x has a single layer. If matrix=FALSE, the values are returned as a vector. In cell-order by layer.

If x is a SpatVector: a data.frame

Value

matrix or vector

Examples

```
r <- rast(system.file("exdata/test.tif", package="terra"))
r
v <- values(r)
values(r) <- v * 10
r

f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
x <- values(v)
x
values(v) <- x[,1:2]
v
```

---

vect *Create SpatVector objects*

Description

Create a new SpatVector

Usage

```
## S4 method for signature 'data.frame'
vect(x, type="points", atts=NULL, crs=NA, ...)
```

Arguments

x	character (filename), data.frame, matrix, or missing. If x is a filename, all other arguments are ignored
type	character. Geometry type. Must be "points", "lines", or "polygons"
atts	data.frame with the attributes. The number of rows must match the number of geometrical elements

crs	the coordinate reference system (PROJ4 notation)
...	additional matrices and/or lists with matrices

Value

SpatVector object

Examples

```
f <- system.file("exdata/lux.shp", package="terra")
f
v <- vect(f)
v

x1 <- rbind(c(-180,-20), c(-140,55), c(10, 0), c(-140,-60))
x2 <- rbind(c(-10,0), c(140,60), c(160,0), c(140,-55))
x3 <- rbind(c(-125,0), c(0,60), c(40,5), c(15,-45))
hole <- rbind(c(80,0), c(105,13), c(120,2), c(105,-13))

z <- rbind(cbind(object=1, part=1, x1, hole=0), cbind(object=2, part=1, x3, hole=0),
cbind(object=3, part=1, x2, hole=0), cbind(object=3, part=1, hole, hole=1))
colnames(z)[3:4] <- c('x', 'y')
z <- data.frame(z)

p <- vect(z, "polygons")
p

z[z$hole==1, "object"] <- 4
lns <- vect(z[,1:4], "lines")
plot(p)
lines(lns, col='red')
```

---

vector-attributes *Get or replace attribute values of a SpatVector*

Description

Replace values of a SpatVector.

Usage

```
## S4 method for signature 'SpatVector'
x$name
## S4 replacement method for signature 'SpatVector'
x$name<-value
```

Arguments

x	SpatVector
name	character
value	vector

Value

vector

See Also

values

Examples

```
f <- system.file("exdata/lux.shp", package="terra")
v <- vect(f)
v$NAME_1
v$ID_1 <- LETTERS[1:12]
v$new <- sample(12)
values(v)
```

---

warp *Transfer values of a SpatRaster to another one with a different geometry*

Description

Warp transfers values between SpatRaster objects that do not align (have a different origin and/or coordinate reference system (crs)). This is also known as "resample" and/or "reproject". If the origin and crs are the same, you should consider using these other functions instead: aggregate, disaggregate, crop.

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster'
warp(x, y, method="bilinear",
       filename="", overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster to be warped
y	SpatRaster that x should be warped to
method	character. Method used to compute values. "bilinear" for bilinear interpolation, or "ngb" for "nearest neighbor"
filename	character. Output filename. Optional

overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files as in writeRaster
...	additional arguments. None implemented

Value

SpatRaster

See Also

aggregate, disaggregate, crop, project,

Examples

```
r <- rast(nrow=3, ncol=3, xmin=0, xmax=10, ymin=0, ymax=10)
values(r) <- 1:ncell(r)
s <- rast(nrow=25, ncol=30, xmin=1, xmax=11, ymin=-1, ymax=11)
x <- warp(r, s, method='bilinear')

opar <- par(no.readonly =TRUE)
par(mfrow=c(1,2))
plot(r)
plot(x)
par(opar)

# warp to different CRS
a <- rast(ncol=40, nrow=40, xmin=-110, xmax=-90, ymin=40, ymax=60,
          crs="+proj=longlat +datum=WGS84")
values(a) = 1:ncell(a)
b <- rast(ncol=94, nrow=124, xmin=-944881, xmax=935118, ymin=4664377, ymax=7144377,
     crs="+proj=lcc +lat_1=48 +lat_2=33 +lon_0=-100 +datum=WGS84")
w <- warp(a, b)
```

---

writeRaster *Write raster data to a file*

Description

Write a SpatRaster object to a file.

Usage

```
## S4 method for signature 'SpatRaster,character'
writeRaster(x, filename,
    overwrite=FALSE, wopt=list(), ...)
```

Arguments

x	SpatRaster
filename	character. Output filename
overwrite	logical. If TRUE, filename is overwritten
wopt	list. Options for writing files. See Details
...	additional arguments. None implemented

Details

In many methods, options for writing raster files to disk can be provided with the wopt argument. wopt should be a named list. The following names are valid: filetype, datatype, gdal, tempdir, progress, memfrac and names.

Values for filetype are GDAL driver names. If this argument is not supplied, the driver is derived from the filename.

Values for datatype are "INT2U", "INT2S", "INT4U", "INT4S", "FLT4S", "FLT8S". The first three letters indicate whether it is an integer (whole numbers) of float (decimal numbers), the fourth character indicates the number of bytes used (allowing for large numbers, and/or more precision), and the "S" or "U" indicate whether the values are signed (both negative and positive) or unsigned (postive values only).

Values for gdal are GDAL driver specific datasource creation options. See the GDAL documentation. For example: gdal=c("COMPRESS=LZW","TFW=YES")

Value

SpatRaster (invisibly). This function is used for the side-effect of writing values to a file.

---

writeVector *Write vector data to a file*

Description

Write a SpatVector object to a file.

Usage

```
## S4 method for signature 'SpatVector,character'
writeVector(x, filename, overwrite=FALSE, ...)
```

Arguments

x	SpatVector
filename	character. Output filename
overwrite	logical. If TRUE, filename is overwritten
...	additional arguments. None implemented

Value

SpatVector (invisibly). This function is used for the side-effect of writing values to a file.

`xmin` *Get or set single values of an extent*

Description

Get or set single values of an extent. Values can be set for a SpatExtent or SpatRaster, but not for a SpatVector)

Usage

```
## S4 method for signature 'SpatExtent'
xmin(x)
## S4 method for signature 'SpatExtent'
xmax(x)
## S4 method for signature 'SpatExtent'
ymin(x)
## S4 method for signature 'SpatExtent'
ymax(x)
## S4 method for signature 'SpatRaster'
xmin(x)
## S4 method for signature 'SpatRaster'
xmax(x)
## S4 method for signature 'SpatRaster'
ymin(x)
## S4 method for signature 'SpatRaster'
ymax(x)
## S4 method for signature 'SpatVector'
xmin(x)
## S4 method for signature 'SpatVector'
xmax(x)
## S4 method for signature 'SpatVector'
ymin(x)
## S4 method for signature 'SpatVector'
ymax(x)

xmin(x, ...) <- value
xmax(x, ...) <- value
ymin(x, ...) <- value
ymax(x, ...) <- value
```

Arguments

`x`	SpatRaster, SpatExtent, or SpatVector
`value`	numeric
`...`	additional arguments. None implemented

Value

SpatExtent or numeric coordinate

Examples

```
r <- rast()
ext(r)
ext(c(0, 20, 0, 20))

xmin(r)
xmin(r) <- 0
xmin(r)
```

---

xyRowColCell *Coordinates from a row, column or cell number and vice versa*

Description

Get coordinates of the center of raster cells for a row, column, or cell number of a SpatRaster object. Or get row, column, or cell numbers from coordinates or from each other.

Cell numbers start at 1 in the upper left corner, and increase from left to right, and then from top to bottom. The last cell number equals the number of cells of the SpatRaster object. row numbers start at 1 at the top, column numbers start at 1 at the left.

Usage

```
## S4 method for signature 'SpatRaster,numeric'
xFromCol(object, col)
## S4 method for signature 'SpatRaster,numeric'
yFromRow(object, row)
## S4 method for signature 'SpatRaster,numeric'
xyFromCell(object, cell, ...)
## S4 method for signature 'SpatRaster,numeric'
xFromCell(object, cell)
## S4 method for signature 'SpatRaster,numeric'
yFromCell(object, cell)
## S4 method for signature 'SpatRaster,numeric'
colFromX(object, x)
## S4 method for signature 'SpatRaster,numeric'
rowFromY(object, y)
## S4 method for signature 'SpatRaster,numeric,numeric'
cellFromRowCol(object, row, col, ...)
## S4 method for signature 'SpatRaster,numeric,numeric'
cellFromRowColCombine(object, row, col, ...)
## S4 method for signature 'SpatRaster,numeric'
rowFromCell(object, cell)
## S4 method for signature 'SpatRaster,numeric'
```

```
colFromCell(object, cell)
## S4 method for signature 'SpatRaster,numeric'
rowColFromCell(object, cell)
## S4 method for signature 'SpatRaster,matrix'
cellFromXY(object, xy)
```

Arguments

object	SpatRaster
cell	integer. cell number(s)
col	integer. column number(s)
row	integer row number(s)
x	x coordinate(s)
y	y coordinate(s)
xy	matrix of x and y coordinates
...	additional arguments. None implemented

Details

Cell numbers start at 1 in the upper left corner, and increase from left to right, and then from top to bottom. The last cell number equals the number of cells of the SpatRaster (see ncell).

Value

xFromCol, yFromCol, xFromCell, yFromCell: vector of x or y coordinates

xyFromCell: matrix(x,y) with coordinate pairs

colFromX, rowFromY, cellFromXY, cellFromRowCol, rowFromCell, colFromCell: vector of row, column, or cell numbers

rowColFromCell: matrix of row and column numbers

Examples

```
r <- rast()

xFromCol(r, c(1, 120, 180))
yFromRow(r, 90)
xyFromCell(r, 10000)
xyFromCell(r, c(0, 1, 32581, ncell(r), ncell(r)+1))

cellFromRowCol(r, 5, 5)
cellFromRowCol(r, 1:2, 1:2)
cellFromRowCol(r, 1, 1:3)

# all combinations
cellFromRowColCombine(r, 1:2, 1:2)

colFromX(r, 10)
rowFromY(r, 10)
cellFromXY(r, cbind(c(10,5), c(15, 88)))
```

zonal *Zonal statistics*

Description

Compute zonal statistics, that is summarized values of a SpatRaster for each "zone" defined by another SpatRaster.

If fun is a true function, zonal may fail for very large SpatRaster objects, except for the functions ("mean", "min", "max", or "sum").

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster'
zonal(x, z, fun=mean, ...)
```

Arguments

x	SpatRaster
z	SpatRaster with values representing zones
fun	function to be applied to summarize the values by zone. Either as character: "mean", "min", "max", "sum", or, for relatively small SpatRasters, a proper function
...	additional arguments passed to fun

Value

A data.frame with a value for each zone (unique value in zones)

See Also

See global for "global" statistics (i.e., all of x is considered a single zone), app for local statistics, and extract for summarizing values for polygons

Examples

```
r <- rast(ncols=10, nrows=10)
values(r) <- 1:ncell(r)
z <- rast(r)
values(z) <- rep(1:4, each=25)
zonal(r, z, "sum", na.rm=TRUE)
```

zoom *Zoom in on a map*

Description

Zoom in on a map (plot) by providing a new extent, by default this is done by clicking twice on the map.

Usage

```
## S4 method for signature 'SpatRaster'
zoom(x, ext=draw(), maxcell=10000, layer=1, new=TRUE, ...)
```

Arguments

x	SpatRaster
ext	SpatExtent
maxcell	Maximum number of cells used for the map
layer	Positive integer to select the layer to be used
new	Logical. If TRUE, the zoomed in map will appear on a new device (window)
...	additional paramters for plot

Value

SpatExtent (invisibly)

See Also

draw, plot

Index