Package ‘rgrass7’

September 28, 2018

Version 0.1-12
Date 2018-09-24
Title Interface Between GRASS 7 Geographical Information System and R
Description Interpreted interface between 'GRASS' 7 geographical information system and R, based on starting R from within the 'GRASS' 'GIS' environment, or running free-standing R in a temporary 'GRASS' location; the package provides facilities for using all 'GRASS' commands from the R command line. This package may not be used for 'GRASS' 6, for which 'spgrass6' should be used.

Depends R (>= 3.3.0), sp (>= 0.9), XML
Imports stats, utils, methods
Suggests rgdal (>= 1.0-6), RSQLite
SystemRequirements GRASS (>= 7)
License GPL (>= 2)

URL http://grass.osgeo.org/,
    https://r-forge.r-project.org/projects/spgrass/

    xml1.R

NeedsCompilation no

Author Roger Bivand [cre, aut],
    Rainer Krug [ctb],
    Markus Neteler [ctb],
    Sebastian Jeworutzki [ctb]

Maintainer Roger Bivand <Roger.Bivand@nhh.no>

Repository CRAN
Date/Publication 2018-09-28 08:20:03 UTC
R topics documented:

rgrass7-package .................................................. 2
execGRASS .......................................................... 3
gmeta ............................................................... 6
initGRASS ............................................................ 7
readRAST ............................................................. 9
readVECT ............................................................. 12

Index 16

| rgrass7-package | Interface between GRASS geographical information system and R |

Description

Interpreted interface between GRASS geographical information system, version 7, and R, based on starting R from within the GRASS environment, or on running R stand-alone and creating a throw-away GRASS environment from within R. The interface uses classes defined in the sp package to hold spatial data.

Details

Index:

readRAST       read GRASS raster files
writeRAST      write GRASS raster files
readVECT       read GRASS vector object files
writeVECT      write GRASS vector object files
gmeta          read GRASS metadata from the current LOCATION
gmeta2grd      create a GridTopology object from the GRASS region
vInfo          return vector geometry information
vColumns       return vector database columns information
vDataCount     return count of vector database rows
vect2neigh     return area neighbours with shared boundary length

Note that the examples now use the smaller subset North Carolina location: http://grass.osgeo.org/sampledata/north_carolina/nc_basic_spm_grass7.tar.gz

Author(s)

Roger Bivand

Maintainer: Roger Bivand <Roger.Bivand@nhh.no>
**Examples**

```r
if (nchar(Sys.getenv("GISRC")) > 0 &&
   read.dcf(Sys.getenv("GISRC"))[1, "LOCATION_NAME"] == "nc_basic_spm_grass7") {
   require(rgdal)
   elevation <- readRAST("elevation", ignore.stderr=TRUE, plugin=FALSE)
   summary(elevation)
   grd <- gmeta2grd(ignore.stderr=TRUE)
   seed
   pts <- spsample(elevation, 200, "random")
   sample <- SpatialPointsDataFrame(pts, data=over(pts, elevation))
   summary(sample)
   writeVECT(sample, "sp_dem", v.in.ogr_flags=c("overwrite", "o"), ignore.stderr=TRUE)
   bugsDF <- readVECT("schools", ignore.stderr=TRUE, mapset="PERMANENT")
   summary(bugsDF)
   vInfo("streams", ignore.stderr=TRUE)
   vColumns("streams", ignore.stderr=TRUE)
   vDataCount("streams", ignore.stderr=TRUE)
   streams <- readVECT("streams", type="line",
       remove.duplicates=FALSE, ignore.stderr=TRUE, plugin=FALSE)
   summary(streams)
}
```

---

**execGRASS Run GRASS commands**

**Description**

The functions provide an interface to GRASS commands run through `system`, based on the values returned by the `--interface` description flag using XML parsing. If required parameters are omitted, and have declared defaults, the defaults will be used.

**Usage**

```r
eexecGRASS(cmd, flags = NULL, ..., parameters = NULL, intern = NULL,
   ignore.stderr = NULL, Sys_ignore.stdout=FALSE, Sys_wait=TRUE,
   Sys_input=NULL, Sys_show.output.on.console=TRUE, Sys_minimized=FALSE,
   Sys_invisible=TRUE, echoCmd=NULL, redirect=FALSE, legacyExec=NULL)
doGRASS(cmd, flags = NULL, ..., parameters = NULL, echoCmd=NULL,
   legacyExec=NULL)
parseGRASS(cmd, legacyExec=NULL)
## S3 method for class 'GRASS_interface_desc'
print(x, ...)
getXMLencoding()
setXMLencoding(enc)
```
Arguments

- **cmd**: GRASS command name
- **flags**: character vector of GRASS command flags
- **parameters**: list of GRASS command parameters, used if GRASS parameters are not given as R arguments directly; the two methods for passing GRASS parameters may not be mixed. The storage modes of values passed must match those required in GRASS, so a single GRASS string must be a character vector of length 1, a single GRASS integer must be an integer vector of length 1 (may be an integer constant such as 10L), and a single GRASS float must be a numeric vector of length 1. For multiple values, use vectors of suitable length.
- **intern**: default NULL, in which case set internally from `get.useInternOption`; a logical (not 'NA') which indicates whether to make the output of the command an R object. Not available unless 'popen' is supported on the platform.
- **ignore.stderr**: default NULL, taking the value set by `set.ignore.stderrOption`, a logical indicating whether error messages written to 'stderr' should be ignored.

Other arguments to print method.

Details

`parseGRASS` checks to see whether the GRASS command has been parsed already and cached in this session; if not, it reads the interface description, parses it and caches it for future use. `doGRASS` assembles a proposed GRASS command with flags and parameters as a string, wrapping `parseGRASS`, and `execGRASS` is a wrapper for `doGRASS`, running the command through `system` (from 0.7-4, the
... argument is not used for passing extra arguments for system). The command string is termed proposed, because not all of the particular needs of commands are provided by the interface description, and no check is made for the existence of input objects. Support for multiple parameter values added with help from Patrick Caldon. Support for defaults and for direct use of GRASS parameters instead of a parameter list suggested by Rainer Krug.

Value

parseGRASS returns a GRASS_interface_desc object, doGRASS returns a character string with a proposed GRASS command - the expanded command name is returned as an attribute, and execGRASS returns what system or system2 return, particularly depending on the intern argument when the character strings output by GRASS modules are returned. If intern is FALSE, system returns the module exit code, while system2 returns the module exit code with “resOut” and “resErr” attributes.

Note

If any package command fails with a UTF-8 error from the XML package, try using setXMLencoding to work around the problem that GRASS modules declare -interface-description output as UTF-8 without ensuring that it is (French is of 6.4.0 RC5 latin1).

Author(s)

Roger S. Bivand, e-mail: <Roger.Bivand@nhh.no>

See Also

system

Examples

if (nchar(Sys.getenv("GISRC")) > 0 &
  read.dcf(Sys.getenv("GISRC"))[1,"LOCATION_NAME"] == "nc_basic_spm_grass7") {
  echoCmd <- get.echoCmdOption()
  set.echoCmdOption(TRUE)
  print(parseGRASS("r.slope.aspect"))
  doGRASS("r.slope.aspect", flags=c("overwrite"),
    elevation="elevation.dem", slope="slope", aspect="aspect")
  pars <- list(elevation="elevation", slope="slope", aspect="aspect")
  doGRASS("r.slope.aspect", flags=c("overwrite"), parameters=pars)
  print(parseGRASS("r.buffer"))
  doGRASS("r.buffer", flags=c("overwrite"), input="schools", output="bmap",
    distances=seq(1000,15000,1000))
  pars <- list(input="schools", output="bmap", distances=seq(1000,15000,1000))
  doGRASS("r.buffer", flags=c("overwrite"), parameters=pars)
  set.echoCmdOption(legacyExec)
  try(res <- execGRASS("r.stats", input = "fire_blocksgg", # no such file
    flags = c("C", "n"), silent=FALSE)
  res <- execGRASS("r.stats", input = "fire_blocksgg", flags = c("C", "n"),
    legacyExec=TRUE)
  print(res)
if (res != 0) {
  resERR <- execGRASS("r.stats", input = "fire_blocksgg",
    flags = c("C", "n"), redirect=TRUE, legacyExec=TRUE)
  print(resERR)
}
}

---

**gmeta**

Reads GRASS metadata from the current LOCATION

---

**Description**

GRASS LOCATION metadata are read into a list in R; helper function `getLocationProj` returns an `spproj`-compliant PROJ.4 string of projection information. The helper function `gmeta2grd` creates a `GridTopology` object from the current GRASS mapset region definitions.

**Usage**

```r
getLocationProj(ignore.stderr = FALSE)
gmeta2grd(ignore.stderr = FALSE)
## S3 method for class 'gmeta'
print(x, ...)
get.ignore.stderrOption()
get.stop_on_no_flags_parasOption()
get.useGDALOption()
get.pluginOption()
get.echoCmdOption()
get.useInternOption()
get.legacyExecOption()
get.defaultFlagsOption()
get.suppressEchoCmdInFuncOption()
set.ignore.stderrOption(value)
set.stop_on_no_flags_parasOption(value)
set.useGDALOption(value)
set.pluginOption(value)
set.echoCmdOption(value)
set.useInternOption(value)
set.legacyExecOption(value)
set.defaultFlagsOption(value)
set.suppressEchoCmdInFuncOption(value)
```

**Arguments**

- `ignore.stderr` default FALSE, can be set to TRUE to silence `system()` output to standard error; does not apply on Windows platforms
- `x` S3 object returned by `gmeta`
arguments passed through print method
logical value for setting options on ignore.stderr set by default on package load to FALSE, useGDAL set by default on package load to TRUE, plugin set by default on package load to TRUE, echoCmd set by default on package load to FALSE, useIntern sets the intern argument globally; legacyExec sets the legacyExec option globally, but is initialized to FALSE on unix systems (all but Windows) and TRUE on Windows; defaultFlags is initialized to NULL, but may be a character vector with values from c("quiet", "verbose") suppressEchoCmdInFunc default TRUE suppresses the effect of echoCmd within package functions, may be set FALSE for debugging.

Value

Returns list of g.gisenv, g.region -g3, and g.proj values

Author(s)

Roger S. Bivand, e-mail: <Roger.Bivand@nhh.no.>

Examples

```r
if (nchar(Sys.getenv("GISRC")) > 0 &
    read.dcf(Sys.getenv("GISRC"))[1,"LOCATION_NAME"] == "nc_basic_spm_grass7") {
  G <- gmeta()
  print(G)
  CRS(getLocationProj())
  grd <- gmeta2grd()
  print(grd)
  ncells <- prod(slot(grd, "cells.dim"))
  df <- data.frame(k=rep(1, ncells))
  mask_SG <- SpatialGridDataFrame(grd, data=df)
  print(summary(mask_SG))
}
```

Description

Run GRASS interface in an R session not started within GRASS. In general, most users will use initGRASS in throwaway locations, to use GRASS modules on R objects without the need to define and populate a location. The function initializes environment variables used by GRASS, the .gisrc used by GRASS for further environment variables, and a temporary location.

The locking functions are used internally, but are exposed for experienced R/GRASS scripters needing to use the GRASS module “g.mapset” through initGRASS in an existing GRASS location. In particular, “g.mapset” may leave a .gislock file in the current MAPSET, so it may be important to call unlink_gislock to clean up before quitting the R session. remove_GISRC may be used
to try to remove the file given in the “GISRC” environment variable if created by \texttt{initGRASS} with argument \texttt{remove\_GISRC=TRUE}.

\textbf{Usage}

\texttt{initGRASS(gisBase, home, SG, gisDbase, addon\_base, location, mapset, override = FALSE, use\_g.dirseps.exe = TRUE, pid, remove\_GISRC=FALSE)}

\texttt{get\_GIS\_LOCK()}
\texttt{set\_GIS\_LOCK(pid)}
\texttt{unset\_GIS\_LOCK()}
\texttt{unlink\_gislock()}
\texttt{remove\_GISRC()}

\textbf{Arguments}

\begin{tabular}{ll}
\texttt{gisBase} & The directory path to GRASS binaries and libraries \\
\texttt{home} & The directory in which to create the .gisrc file; defaults to $\texttt{HOME}$ on Unix systems and to \texttt{USERPROFILE} on Windows systems; can usually be set to \texttt{tempdir()} \\
\texttt{SG} & An optional \texttt{SpatialGrid} object to define the DEFAULT\_WIND of the temporary location \\
\texttt{gisDbase} & if missing, \texttt{tempdir()} will be used; GRASS GISDBASE directory for the working session \\
\texttt{addon\_base} & if missing, assumed to be “$\texttt{HOME}/.grass7/addons” on Unix-like platforms, on MS Windows “$\texttt{APPDATA}/GRASS7/addons”, and checked for existence \\
\texttt{location} & if missing, basename(tempfile()) will be used; GRASS location directory for the working session \\
\texttt{mapset} & if missing, basename(tempfile()) will be used; GRASS mapset directory for the working session \\
\texttt{override} & default FALSE, set to TRUE if accidental trashing of GRASS .gisc files and locations is not a problem \\
\texttt{use\_g.dirseps.exe} & default TRUE; when TRUE appears to work for WinGRASS Native binaries, when FALSE for QGIS GRASS binaries; ignored on other platforms. \\
\texttt{pid} & default \texttt{as.integer(round(runif(1, 1, 1000)))}, integer used to identify GIS\_LOCK; the value here is arbitrary, but probably should be set correctly \\
\texttt{remove\_GISRC} & default FALSE; if TRUE, attempt to unlink the temporary file named in the “GISRC” environment variable when the R session terminates or when this package is unloaded
\end{tabular}

\textbf{Details}

The function establishes an out-of-GRASS working environment providing GRASS commands with the environment variable support required, and may also provide a temporary location for use until the end of the running R session if the \texttt{home} argument is set to \texttt{tempdir()}, and the \texttt{gisDbase} argument is not given. Running \texttt{gmeta6} shows where the location is, should it be desired to archive it before leaving R.
Value

The function runs gmeta6 before returning the current values of the running GRASS session that it provides.

Note

If any package command fails with a UTF-8 error from the XML package, try using setXMLencoding to work around the problem that GRASS modules declare –interface-description output as UTF-8 without ensuring that it is (French is of 6.4.0 RC5 latin1).

Author(s)

Roger S. Bivand, e-mail: <roger.Bivand@nhh.no>

See Also

gmeta

Examples

```r
## Not run:
initGRASS("/usr/bin/grass-7.0.0", home=tempdir())
initGRASS("C:/GRASS", home=tempdir())

## End(Not run)
```

readRAST

Read and write GRASS 7 raster files

Description

Read GRASS 7 raster files from GRASS 7 into R SpatialGridDataFrame objects, and write single columns of R SpatialGridDataFrame objects to GRASS 7. readRAST and writeRAST use temporary binary files and r.out.bin and r.in.bin for speed reasons.

Usage

```r
readRAST(vname, cat=NULL, ignore.stderr = get.ignore.stderrOption(),
          NODATA=NULL, plugin=get.pluginOption(), mapset=NULL,
          useGDAL=get.useGDALOption(), close_OK=TRUE, drivername="GTiff",
          driverFileExt=NULL, return_SGDF=TRUE)
writeRAST(x, vname, zcol = 1, NODATA=NULL,
          ignore.stderr = get.ignore.stderrOption(), useGDAL=get.useGDALOption(),
          overwrite=FALSE, flags=NULL, drivername="GTiff")
```
Arguments

vname
A vector of GRASS 7 raster file names

cat
default NULL; if not NULL, must be a logical vector matching vname, stating which (CELL) rasters to return as factor

ignore.stderr
default taking the value set by set.ignore.stderrOption; can be set to TRUE to silence system() output to standard error; does not apply on Windows platforms

plugin
default taking the value set by set.pluginOption; NULL does auto-detection, changes to FALSE if vname is longer than 1, and a sanity check will be run on raster and current region, and the function will revert to FALSE if mismatch is found; if TRUE, the plugin is available and the raster should be read in its original region and resolution; if the plugin is used, no further arguments other than mapset are respected

mapset
default NULL, if plugin is TRUE, the mapset of the file to be imported will be autodetected; if not NULL and if plugin is TRUE, a character string overriding the autodetected mapset, otherwise ignored

useGDAL
(effectively defunct, only applies to use of plugin) default taking the value set by set.useGDALOption; use plugin and readGDAL if autodetected or plugin=TRUE; or for writing writeGDAL, GTiff, and r.in.gdal, if FALSE using r.out.bin or r.in.bin

close_OK
default TRUE - clean up possible open connections used for reading metadata; may be set to FALSE to avoid the side-effect of other user-opened connections being broken

drivername
default "GTiff"; a valid GDAL writable driver name to define the file format for intermediate files

driverFileExt
default NULL; otherwise string value of required driver file name extension

return_SGDF
default TRUE returning a SpatialGridDataFrame object, if FALSE, return a list with a GridTopology object, a list of bands, and a proj4string; see example below

x
A SpatialGridDataFrame object for export to GRASS as a raster layer

zcol
Attribute column number or name

NODATA
by default NULL, in which case it is set to one less than floor() of the data values, otherwise an integer NODATA value (required to be integer by GRASS r.out.bin)

overwrite
default FALSE, if TRUE inserts "overwrite" into the value of the flags argument if not already there to allow existing GRASS rasters to be overwritten

flags
default NULL, character vector, for example "overwrite"

Value
readRAST returns a SpatialGridDataFrame objects with an data.frame in the data slots, and with the projection argument set. Note that the projection argument set is the the GRASS rendering of proj4, and will differ from the WKT/ESRI rendering returned by readVECT in form but not meaning. They are exchangeable but not textually identical, usually with the +ellps= term replaced by ellipsoid parameters verbatim. If return_SGDF is FALSE, a list with a GridTopology object, a list of bands, and a proj4string is returned, with an S3 class attribute of "gridList".
readRAST

Author(s)
Roger S. Bivand, e-mail: <Roger.Bivand@nhh.no>

Examples

```r
if (nchar(Sys.getenv("GISRC")) > 0 &&
    read.dcf(Sys.getenv("GISRC"))[1,"LOCATION_NAME"] == "nc_basic_spm_grass7") {
    GV <- Sys.getenv("GRASS_VERBOSE")
    Sys.setenv("GRASS_VERBOSE"="0")
    require(rgdal)
    ois <- get-ignore.stderrOption()
    set.ignore.stderrOption(TRUE)
    get.useGDALOption()
    nc_basic <- readRAST(c("geology", "elevation"), cat=c(TRUE, FALSE),
                         useGDAL=FALSE)
    nc_basic <- readRAST(c("geology", "elevation"), cat=c(TRUE, FALSE),
                         useGDAL=TRUE)
    print(table(nc_basic$geology))
    execGRASS("r.stats", flags=c("c", "l", "quiet"), input="geology")
    boxplot(nc_basic$geology - nc_basic$geology)
    nc_basic$sqdem <- sqrt(nc_basic$geology)
    if (!("GRASS\%gdaldrivers\$name") {
        execGRASS("g.region", raster="elevation")
        dem1 <- readRAST("elevation", plugin=TRUE, mapset="PERMANENT")
        print(summary(dem1))
        execGRASS("g.region", raster="elevation")
    }
    writeRAST(nc_basic, "sqdemSP", zcol="sqdem", flags="quiet")
    execGRASS("r.info", map="sqdemSP")
    execGRASS("g.remove", flags="f", name="sqdemSP", type="raster")
    writeRAST(nc_basic, "sqdemSP", zcol="sqdem", useGDAL=TRUE, flags="quiet")
    execGRASS("r.info", map="sqdemSP")
    print(system.time(sqdemSP <- readRAST(c("sqdemSP", "elevation"),
                                          useGDAL=TRUE, return_SGDF=FALSE)))
    print(system.time(sqdemSP <- readRAST(c("sqdemSP", "elevation"),
                                          useGDAL=TRUE, return_SGDF=TRUE)))
    print(system.time(sqdemSP <- readRAST(c("sqdemSP", "elevation"),
                                          useGDAL=FALSE, return_SGDF=TRUE)))
    print(system.time(sqdemSP <- readRAST(c("sqdemSP", "elevation"),
                                          useGDAL=FALSE, return_SGDF=FALSE)))
    str(sqdemSP)
    mat <- do.call("cbind", sqdemSP$dataList)
    str(mat)
    print(system.time(SGDF <- SpatialGridDataFrame(grid=sqdemSP$grid,
                                                  proj4string=sqdemSP$proj4string, data=as.data.frame(sqdemSP$dataList))))
    summary(SGDF)
    execGRASS("g.remove", flags="f", name="sqdemSP", type="raster")
    execGRASS("r.mapcalc", expression="basins0 = basins - 1")
    execGRASS("r.stats", flags="c", input="basins0")
    basins0 <- readRAST("basins0")
    print(table(basins0@$basins0))
    basins0 <- readRAST("basins0", plugin=FALSE)
```
print(table(basins@basins))
execGRASS("g.remove", flags="f", name="basins", type="raster")
Sys.setenv("GRASS_VERBOSITY"="GV")
set.ignore.stderrOption(ois)

---

**Description**

readVECT moves one GRASS 7 vector object file with attribute data through a temporary shapefile to a Spatial*DataFrame object of type determined by the GRASS 7 vector object; writeVECT moves a Spatial*DataFrame object through a temporary shapefile to a GRASS vector object file. vect2neigh returns neighbour pairs with shared boundary length as described by Markus Neteler, in https://stat.ethz.ch/pipermail/r-sig-geo/2005-October/000616.html. cygwin_clean_temp can be called to try to clean the GRASS mapset-specific temporary directory under cygwin.

**Usage**

```r
readVECT(vname, layer, type=NULL, plugin=NULL, 
         remove.duplicates = TRUE, ignore.stderr=NULL, 
         with_prj=TRUE, with_c=FALSE, mapset=NULL, 
         pointDropZ=FALSE, driver=NULL)
writeVECT(SDF, vname, v.in.ogr_flags=NULL, 
         ignore.stderr = NULL, driver=NULL, 
         min_area=0.001, snap=1)
vInfo(vname, layer, ignore.stderr = NULL)
vcColumns(vname, layer, ignore.stderr = NULL)
vDataCount(vname, layer, ignore.stderr = NULL)
vect2neigh(vname, ID=NULL, ignore.stderr = NULL, remove=TRUE, vname2=NULL, 
         units="k")
```

**Arguments**

- `vname` A GRASS 7 vector file name
- `layer` a layer name (string); if missing set to default of “1”
- `type` override type detection when multiple types are non-zero, passed to v.out.ogr
- `plugin` default NULL if which case it will be set to the value set by set.pluginOption; NULL for auto-detection, may be set to FALSE to avoid or TRUE if the plugin is known to be available; if the plugin is used, no further arguments other than mapset are respected
- `remove.duplicates` In line and area vector objects, multiple geometrical features may be associated with a single cat number, leading to duplication of data rows; this argument attempts to combine the geometrical features so that they match a single data row
ignore.stderr default the value set by set.ignore.stderrOption; NULL, taking the value set by set.ignore.stderrOption, can be set to TRUE to silence system() output to standard error; does not apply on Windows platforms

with_prj default TRUE, write ESRI-style PRJ file for transferred data

with_c default FALSE in GRASS 7; if FALSE, export features with category (labeled) only; if not default, all features are exported, including GRASS “islands” which are probably spurious exterior rings filling holes.

mapset if plugin is TRUE, the mapset of the file to be imported may be changed from the current mapset by passing a character string

pointDropZ default FALSE, if TRUE, discard third coordinates for point geometries; third coordinates are alway discarded for line and polygon geometries

driver default NULL, which will lead to the choice of the first driver found in a ordered preferred vector, currently c("SQLite", "ESRI Shapefile"); a valid OGR writable driver name to define the file format for intermediate files, one of c("GML", "SQLite"), c("ESRI Shapefile", "MapInfo_File") is preferred as these construct the names of the intermediate files adequately

min_area default 0.0001); Minimum size of area to be imported (square meters) Smaller areas and islands are ignored. Should be greater than snap^2

snap default -1); Snapping threshold for boundaries (map units). '-1' for no snap

SDF A Spatial*DataFrame to be moved to GRASS 7 as a vector object, for SpatialPointsDataFrame, SpatialLinesDataFrame, and SpatialPolygonsDataFrame objects

v.in.ogr_flags Character vector containing additional optional flags and/or options for v.in.ogr, particularly "o" and "overwrite"

ID A valid DB column name for unique identifiers (optional)

remove default TRUE, remove copied vectors created in vect2neigh

vname2 If on a previous run, remove was FALSE, the name of the temporary vector may be given to circumvent its generation

units default “k”; see GRASS v.to.db manual page for alternatives

Value

readVECT imports a GRASS 7 vector object into a Spatial*DataFrame object with the type determined by the type of the GRASS 7 vector object. readVECT and writeVECT attempt to preserve longer column/field names despite using the “ESRI Shapefile” format for transfer.

vect2neigh returns a data frame object with left and right neighbours and boundary lengths, also given class GRASSneigh and spatial.neighbour (as used in spdep). The incantation to retrieve the neighbours list is sn2listw(vect2neigh())$neighbours, and to retrieve the boundary lengths: sn2listw(vect2neigh())$weights. The GRASSneigh object has two other useful attributes: external is a vector giving the length of shared boundary between each polygon and the external area, and total giving each polygon’s total boundary length.
Note

Please note that the OGR drivers used may not handle missing data gracefully. From rgdal release 0.5-27, missing values are taken as unset OGR field values. If the OGR driver encodes them in this way, NAs will be moved across the interface correctly from R to GRASS, and from GRASS to R using the OGR GRASS vector plugin. Work is continuing to correct v.out.ogr so that it emits unset fields, which affects users with no OGR GRASS plugin for the present. Thanks to Dylan Beaudette for helping with missing data handling.

Author(s)

Roger S. Bivand, e-mail: <Roger.Bivand@nhh.no>

Examples

```r
if (nchar(Sys.getenv("GISRC")) > 0 &
    read.dcf(Sys.getenv("GISRC"))[1,"LOCATION_NAME"] == "nc_basic_spm_grass7") {
  GV <- Sys.getenv("GRASS_VERBOSE")
  Sys.setenv("GRASS_VERBOSE"="0")
  require(rgdal)
  ois <- get.ignore.stderrOption()
  set.ignore.stderrOption(TRUE)
  execGRASS("v.info", map="schools", layer="1")
  print(vInfo("schools"))
  schs <- readVECT("schools", plugin=NULL)
  print(summary(schs))
  schs1 <- readVECT("schools", plugin=FALSE)
  print(summary(schs1))
  writeVECT(schs, "newsch", v.in.ogr_flags="", "overwrite")
  execGRASS("v.info", map="newsch", layer="1")
  nschs <- readVECT("newsch")
  print(summary(nschs))
  print(all.equal(names(nschs), character(vColumns("newsch")}[2])))
  names(nschs) <- paste("ABCDEFGHIJKLMNOPQRSTUVWXYZ", names(nschs), sep="")
  writeVECT(nschs, "newsch1", v.in.ogr_flags="", "overwrite")
  print(all.equal(names(nschs), character(vColumns("newsch1")}[2])))
  nschs1 <- readVECT("newsch1")
  print(all.equal(names(nschs), names(nschs1)[-1]))
  print(summary(nschs1))
  schs <- readVECT("schools", driver="ESRI Shapefile")
  names(schs) <- paste("ABCDEFGHIJKLMNOPQRSTUVWXYZ", names(schs), sep="")
  writeVECT(schs, "newsch", v.in.ogr_flags="", "overwrite",
            driver="ESRI Shapefile")
  print(all.equal(names(schs), character(vColumns("newsch")[-1,2])))
  nschs <- readVECT("newsch", driver="ESRI Shapefile")
  all.equal(names(schs), names(nschs)[-1])
  print(vInfo("roadsmajor"))
  roads <- readVECT("roadsmajor")
  print(summary(roads))
  cen_neig <- vect2neigh("census")
  str(cen_neig)
  Sys.setenv("GRASS_VERBOSE"=GV)
}
```
readVECT

    set.ignore.stderrOption(ois)
}
Index

*Topic package
  rgrass7-package, 2
*Topic spatial
  execGRASS, 3
  gmeta, 6
  initGRASS, 7
  readRAST, 9
  readVECT, 12
  rgrass7-package, 2
doGRASS (execGRASS), 3
execGRASS, 3
get.defaultFlagsOption (gmeta), 6
get.echoCmdOption (gmeta), 6
get.GIS_LOCK (initGRASS), 7
get.ignore.stderrOption (gmeta), 6
get.legacyExecOption (gmeta), 6
get.pluginOption (gmeta), 6
get.stop_on_no_flags_parasOption (gmeta), 6
get.suppressEchoCmdInFuncOption (gmeta), 6
get.useGDALOption (gmeta), 6
get.useInternOption (gmeta), 6
getLocationProj (gmeta), 6
getXMLencoding (execGRASS), 3
gmeta, 6, 9
gmeta2grd (gmeta), 6
initGRASS, 7
parseGRASS (execGRASS), 3
print.gmeta (gmeta), 6
print.GRASS_interface_desc (execGRASS), 3
readRAST, 9
readVECT, 12
remove_GISRC (initGRASS), 7
rgrass7 (rgrass7-package), 2
rgrass7-package, 2
set.defaultFlagsOption (gmeta), 6
set.echoCmdOption (gmeta), 6
set.GIS_LOCK (initGRASS), 7
set.ignore.stderrOption (gmeta), 6
set.legacyExecOption (gmeta), 6
set.pluginOption (gmeta), 6
set.stop_on_no_flags_parasOption (gmeta), 6
set.suppressEchoCmdInFuncOption (gmeta), 6
set.useGDALOption (gmeta), 6
set.useInternOption (gmeta), 6
setXMLencoding (execGRASS), 3
system, 5
unlink_.gislock (initGRASS), 7
unset.GIS_LOCK (initGRASS), 7
vColumns (readVECT), 12
vDataCount (readVECT), 12
vect2neigh (readVECT), 12
vInfo (readVECT), 12
writeRAST (readRAST), 9
writeVECT (readVECT), 12