Package 'EchoviewR'

January 15, 2015

Type Package

Title (semi-) automated processing of active acoustic data using Echoview and COM scripting.

Version 1.0

Date 2014-12-09

Author Lisa-Marie Harrison and Martin J. Cox

Maintainer Lisa-Marie Harrison <lisamarie.k.harrison@gmail.com>

Description This package is an interface between Echoview and R that enables automated processing of acoustic data.

License GPL-2

Depends sp,geosphere,maptools,RDCOMClient,rgeos,lubridate

R topics documented:

`centreLawnOnPosition` *Centre a regular rectangular survey on a given position*

Description

Centres a regular rectangular survey on a desired latitude and longitude

Usage

```
centreLawnOnPosition(centreLon, centreLat,
  proj4string = CRS("+proj=longlat +datum=WGS84"), tolerance = 20, ...)
```

Arguments

`centreLon`	Desired centre location of survey
`centreLat`	Desired centre location of survey
`proj4string`	projection string of class CRS-class
`tolerance`	maximum distance (in metres) between desired survey centre and realised survey centre
`...`	other arguments to be passed into lawnSurvey

Value

Line transect coordinates (lon, lat) as specified in lawnSurvey

Examples

```
## Not run:
coords=centreLawnOnPosition(centreLon=-170,centreLat=-60,lineLengthkm=2,lineSpacingkm=0.5,
startBearingdeg=30,numOfLines=5)
plot(0,0,xlim=range(coords[,1]),ylim=range(coords[,2]),type='n',xlab='Longitude, deg',ylab='Latitude, deg')
arrows(x0=coords[1:(nrow(coords)-1),1], y0=coords[1:(nrow(coords)-1),2],
      x1 = coords[2:nrow(coords),1], y1 = coords[2:nrow(coords),2])
text(coords,row.names(coords),cex=0.6)
points(coords[1,1],coords[1,2],col='blue',pch=17,cex=2)
points(coords[nrow(coords),1],coords[nrow(coords),2],col='blue',pch=15,cex=2)
points(-170,-60,col='purple',pch=19,cex=2)
points(geomean(coords),col='red',pch=19,cex=1)
legend('bottomright',c('Beginning','End','Desired centre','Actual centre'),
 col=c('blue','blue','purple','red'),pch=c(17,15,19,19),pt.cex=c(1,1,2,1))

## End(Not run)
```

---

centreZigZagOnPosition

Centre an zig-zag line transect survey on a given position

Description

Centres a zig-zag survey on a desired latitude and longitude

Usage

```
centreZigZagOnPosition(centreLon, centreLat,
  proj4string = CRS("+proj=longlat +datum=WGS84"), tolerance = 20, ...)
```

Arguments

centreLon	Desired centre location of survey
centreLat	Desired centre location of survey
proj4string	projection string of class CRS-class
tolerance	maximum distance (in metres) between desired survey centre and realised survey centre
...	other arguments to be passed into \linkzigzagSurvey

Details

The call of zigzagSurvey has unrotated=FALSE

Value

Line transect coordinates as specified in zigzagSurvey

Examples

```
## Not run:
coords=centreZigZagOnPosition(centreLon=-33,centreLat=-57,lineLengthkm=60,startBearingdeg=30,
rotationdeg=10,numOfLines=21)
plot(0,0,xlim=range(coords[,1]),ylim=range(coords[,2]),type='n',xlab='Longitude, deg',ylab='Latitude, deg')
arrows(x0=coords[1:(nrow(coords)-1),1], y0=coords[1:(nrow(coords)-1),2],
      x1 = coords[2:nrow(coords),1], y1 = coords[2:nrow(coords),2])
text(coords,row.names(coords),cex=0.6)
points(coords[1,1],coords[1,2],col='blue',pch=17,cex=2)
points(coords[nrow(coords),1],coords[nrow(coords),2],col='blue',pch=15,cex=2)
points(-100,-60,col='purple',pch=19,cex=2)
points(geomean(coords),col='red',pch=19,cex=1)
legend('topright',c('Beginning','End','Desired centre','Actual centre'),
 col=c('blue','blue','purple','red'),pch=c(17,15,19,19),pt.cex=c(1,1,2,1))

## End(Not run)
```

EVAcoVarNameFinder *Find an acoustic variable by name*

Description

This function finds an acoustic variable in an Echoview file by name and returns the variable pointer.

Usage

```
EVAcoVarNameFinder(EVFile, acoVarName)
```

Arguments

EVFile	An Echoview file COM object
acoVarName	The name of an acoustic variable in the Echoview file

Value

a list object with two elements. $EVVar: An Echoview acoustic variable object, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
varObj <- EVAcoVarNameFinder(EVFile, "120 7x7 convolution")$EVVar

## End(Not run)
```

EVAddCalibrationFile *Add a calibration file (.ecs) to a fileset*

Description

This function adds a calibration file (.ecs) to a fileset using COM scripting.

Usage

```
EVAddCalibrationFile(EVFile, filesetName, calibrationFile)
```

Arguments

EVFile	An Echoview file COM object
filesetName	An Echoview fileset name
calibrationFile	An Echoview calibration (.ecs) file path and name

Value

a list object with one element. $msg: message for processing log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVAddCalibrationFile(EVFile = EVFile, filesetName = '038-120-200', calibrationFile = '~~/KAOS/20120326_KAOS_

## End(Not run)
```

EVAddRawData *Add raw data files to an open Echoview file (.EV)*

Description

This function adds raw data files to an open Echoview file (.EV) via COM scripting. The function assumes the Echoview fileset name already exists.

Usage

```
EVAddRawData(EVFile, filesetName, dataFiles)
```

Arguments

EVFile	An Echoview file COM object
filesetName	Echoview fileset name
dataFiles	vector of full path and name for each data file.

Value

a list object with two elements. $nbrFilesInFileset: number of raw data files in the fileset, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVNewFile EVCreateNew

Examples

```
## Not run:
filenamesV <- c('~~/KAOS/raw/L0055-D20030115-T171028-EK60.raw', '~~/KAOS/raw/L0055-D20030115-T182914-EK60.r
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVNewFile(EVAppObj,templateFn="~~/KAOS/KAOStemplate.EV")$EVFile
EVAddRawData(EVFile = EVFile, filesetName = '038-120-200', dataFiles = filenamesV)

## End(Not run)
```

EVAdjustDataRngBitmap *Change the data range bitmap of an acoustic object*

Description

This function changes the data range in an Echoview data range bitmap virtual variable

Usage

```
EVAdjustDataRngBitmap(varObj, minRng, maxRng)
```

Arguments

varObj	An Echoview acoustic variable COM object, perhaps resulting from a call of EVAcoVarNameFinder()
minRng	the minimum data value to set
maxRng	the maximum data value to set

Value

a vector of pre- and post-function call data range settings

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVAcoVarNameFinder

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
varObj <- EVAcoVarNameFinder(EVFile, acoVarName = "38 data range bitmap")$EVVar
EVAdjustDataRngBitmap(varObj, minRng = -90, maxRng = 0)

## End(Not run)
```

EVChangeVariableGrid *Change the grid of an acoustic variable*

Description

This function sets the grid separation and depth reference line for an acoustic variable using COM scripting.

Usage

```
EVChangeVariableGrid(EVFile, acousticVar, verticalType, horizontalType,
  verticalDistance = 0, horizontalDistance = 0, EVLine = NULL)
```

Arguments

EVFile	An Echoview file COM object
acousticVar	an EV acoustic variable object
verticalType	0 = no grid, 1 = time (minutes), 2 = GPS distance (NMi), 3 = Vessel Log Distance (NMi), 4 = Pings, 5 = GPS distance (m), 6 = Vessel Log Distance (m).
horizontalType	0 = no grid, 1 = depth grid, 2 = use reference line.
verticalDistance	vertical grid line spacing. Not needed if verticalType = 0.
horizontalDistance	horizontal grid line spacing. Not needed if horizontalType = 0.
EVLine	an EV line object. Not needed if horizontalType = 0.

Value

a list object with two elements. $dataRangeSettings: a vector of pre- and post-function call data range settings, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVAcoVarNameFinder EVFindLineByName

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
varObj <- EVAcoVarNameFinder(EVFile, acoVarName = "38 seabed and surface excluded")$EVVar
EVLine <- EVFindLineByName(EVFile = EVFile, lineName = "Fixed depth 250 m")

#Change grid to 100m vertical distance and 10m depth grid relative to 100m line
EVChangeVariableGrid(EVFile = EVFile, acousticVar = varObj, verticalType = 5, horizontalType = 2, verticalDis
#remove horizontal and vertical grid
EVChangeVariableGrid(EVFile = EVFile, acousticVar = varObj, verticalType = 0, horizontalType = 0)

## End(Not run)
```

EVClearRawData *Clear all files from a fileset*

Description

This function clears all .raw files from a fileset using COM scripting

Usage

```
EVClearRawData(EVFile, filesetName)
```

Arguments

EVFile	An Echoview file COM object
filesetName	An Echoview fileset name

Value

A list object with one element. $msg message for processing log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVClearRawData(EVFile = EVFile, filesetName = '038-120-200')

## End(Not run)
```

EVCloseFile *Close an open Echoview file (.EV)*

Description

This function closes an open Echoview file (.EV) via COM scripting

Usage

```
EVCloseFile(EVFile)
```

Arguments

EVFile	An Echoview file COM object

Value

a list object with two elements. $chk: Boolean check indicating if the file was successfully closed; $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVCloseFile(EVFile)

## End(Not run)
```

EVCreateFileset *Create a new Echoview fileset*

Description

This function creates a new echoview fileset via COM scripting

Usage

```
EVCreateFileset(EVFile, filesetName)
```

Arguments

EVFile	An Echoview file COM object
filesetName	Echoview fileset name to create

Value

a list object with two elements. $fileset: created fileset COM object, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVNewFile EVCreateNew EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVCreateFileset(EVFile = EVFile, filesetName = 'example')

## End(Not run)
```

EVCreateNew *Create a new Echoview (.EV) file and adds raw data files to it*

Description

This function creates a new Echoview (.EV) file and adds raw data files to it via COM scripting. Works well when populating an existing Echoview template file with raw data files. The newly created Echoview file will remain open in Echoview and can be accessed via the $EVFile objected returned by a successful call of this function.

Usage

```
EVCreateNew(EVAppObj, templateFn = NULL, EVFileName, filesetName, dataFiles,
  CloseOnSave = TRUE)
```

Arguments

EVAppObj	An EV application COM object arising from the call COMCreate('EchoviewCom.EvApplication')
EVFileName	Full path and filename of Echoview (.EV) file to be created.
templateFn	= NULL Full path and filename of template file if used.
filesetName	Echoview fileset name
dataFiles	vector of full path and name for each data file.
CloseOnSave	= TRUE close the EV file in EVFileName once saved.

Details

For the example code to run, the example data must be downloaded from. NB the example code assumes the data and directory structure of the example data has been maintained.

Value

a list object with two elements. $EVFile: EVFile COM object for the newly created Echoview file, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVNewFile EVAddRawData EVCloseFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
pathAndFn=list.files("~~/KAOS/raw/", full.names=TRUE)
#remove any evi type files
eviLoc=grep('.evi',pathAndFn)
if(length(eviLoc)>0) (pathAndFn=pathAndFn[-eviLoc])
EVCreateNew(EVAppObj=EVAppObj,
                templateFn="~~/KAOS/KAOStemplate.EV",
                EVFileName='~~/KAOS/kaos.ev',
                filesetName="038-120-200",
                dataFiles=pathAndFn,
                CloseOnSave = TRUE)

## End(Not run)
```

EVDeleteLine *Deletes an Echoview line object*

Description

This function deletes an Echoview line object using COM scripting

Usage

```
EVDeleteLine(EVFile, evLine)
```

Arguments

EVFile	An Echoview file COM object
evLine	an Echoview line object

Value

a list object with one element- function message for log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVNewFixedDepthLine EVFindLineByName

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

testline <- EVNewFixedDepthLine(EVFile = EVFile, depth = 50, lineName = "test_line")
EVDeleteLine(EVFile = EVFile, evLine = testline)

## End(Not run)
```

EVDeleteRegionClass *Delete an Echoview region class*

Description

This function deletes a region class within an Echoview object using COM scripting.

Usage

```
EVDeleteRegionClass(EVFile, regionClassCOMObj)
```

Arguments

EVFile	An Echoview file COM object
regionClassCOMObj	An Echoview region object

Value

a list object with two elements. $EVVar: An Echoview acoustic variable object, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVRegionClassFinder

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
exampleClass <- EVRegionClassFinder(EVFile, "003")$regionClass
EVDeleteRegionClass(EVFile, exampleClass)

## End(Not run)
```

EVExportIntegrationByCells *Export integration by cells for an acoustic variable*

Description

This function exports the integration by cells for an acoustic variable using COM scripting. Note: This function will only work if the acoustic variable has a grid.

Usage

```
EVExportIntegrationByCells(EVFile, variableName, filePath)
```

Arguments

EVFile	An Echoview file COM object
variableName	a string containing the name of an EV acoustic variable
filePath	a string containing the file path and name to save the exported data to

Value

a list object with 1 element: message for progessing log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVExportIntegrationByCells(EVFile = EVFile, variableName = '38 seabed and surface excluded', filePath = '~~/K

## End(Not run)
```

EVExportRegionDef *Exports an Echoview region definition*

Description

This function exports a single region's definition as a .csv file using COM scripting

Usage

```
EVExportRegionDef(EVFile, regionName, filePath)
```

Arguments

EVFile	An Echoview file COM object
regionName	a string containing the region name to export definitions for
filePath	a string containing the name and file path of the file to export to

Value

a list object with one element- fucntion message for log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

EVExportRegionDef(EVFile, regionName = "Region1", filePath = "~~/KAOS/EVExportRegionDef.csv")

## End(Not run)
```

EVExportRegionDefByClass *Exports definitions for all Echoview regions in a region class*

Description

This function exports definitions for all Echoview regions within a region class using COM scripting

Usage

```
EVExportRegionDefByClass(evRegionClass, filePath)
```

Arguments

evRegionClass	an Echoview region class object
filePath	a string containing the name and file path of the file to export to

Value

a list object with one element- fucntion message for log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVRegionClassFinder

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

regionClass <- EVRegionClassFinder(EVFile, "aggregations")$regionClass
EVExportRegionDefByClass(evRegionClass = regionClass, filePath = "~~/KAOS/EVExportRegionDefByClass_example.c

## End(Not run)
```

EVExportRegionSv *Export Sv data for an Echoview acoustic variable by region*

Description

This function exports the Sv values as a .csv file for an acoustic variable by region using COM scripting

Usage

```
EVExportRegionSv(EVFile, variableName, regionName, filePath)
```

Arguments

EVFile	An Echoview file COM object
variableName	Echoview variable name for which to extract the data
regionName	Echoview region name for which to extract the data
filePath	File path and name (.csv) to save the data

Value

A list object with one element. $msg message for processing log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVImportRegionDef

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVImportRegionDef(EVFile = EVFile, evrFile = '~~/KAOS/off transect regions/20030114_1200000000.evr', regionN
EVExportRegionSv(EVFile = EVFile, variableName = '120 seabed and surface excluded', regionName = 'region_1',

## End(Not run)
```

EVFilesInFileset *Find names of all raw files in a fileset*

Description

This function returns the names of all .raw files in a fileset using COM scripting

Usage

```
EVFilesInFileset(EVFile, filesetName)
```

Arguments

EVFile	An Echoview file COM object
filesetName	An Echoview fileset name

Value

A character vector containing the names of all .raw files in the fileset

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
file.names <- EVFilesInFileset(EVFile = EVFile, filesetName = '038-120-200')

## End(Not run)
```

EVFindFilesetByName *Find an Echoview fileset in an Echoview file*

Description

This function finds an Echoview fileset in an Echoview file via COM scripting

Usage

```
EVFindFilesetByName(EVFile, filesetName)
```

Arguments

EVFile	An Echoview file COM object
filesetName	Echoview fileset name to find

Value

a list object with two elements. $fileset: found fileset COM object, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVNewFile EVCreateNew EVCreateFileset EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVFileset <-EVFindFilesetByName(EVFile = EVFile, filesetName = '038-120-200')$filesetObj

## End(Not run)
```

EVFindFilesetTime *Find the time and date of the start and end of an Echoview fileset*

Description

This function finds the date and time of the first and last measurement in a fileset using COM scripting

Usage

```
EVFindFilesetTime(EVFile, filesetName)
```

Arguments

EVFile	An Echoview file COM object
filesetName	An Echoview fileset name

Value

A list object with two elements $start.time: The date and time of the first measurement in the fileset, and $end.time: The date and time of the last measurement in the fileset

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
survey.time = EVFindFilesetTime(EVFile = EVFile, filesetName = '038-120-200')
start.time <- survey.time$start.time
end.time <- survey.time$end.time

## End(Not run)
```

EVFindLineByName *Find an EV Line object by name*

Description

This function finds an EV Line in an EV file object by name using COM scripting.

Usage

```
EVFindLineByName(EVFile, lineName)
```

Arguments

EVFile	An Echoview file COM object
lineName	a string containing the name of the line to find

Value

an Echoview line object

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVLine <- EVFindLineByName(EVFile = EVFile, lineName = "Fixed depth 250 m")

## End(Not run)
```

EVFindRegionByName *Finds an Echoview region by name*

Description

This function finds an Echoview region by name using COM scripting

Usage

```
EVFindRegionByName(EVFile, regionName)
```

Arguments

EVFile	An Echoview file COM object
regionName	a string containing the name of the region to find

Value

the Echoview region object

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

testRegion <- EVFindRegionByName(EVFile, "Region1")

## End(Not run)
```

EVFindRegionClass *Find the class of an Echoview region object*

Description

This function finds the class of an Echoview region object using COM scripting.

Usage

```
EVFindRegionClass(evRegion)
```

Arguments

evRegion	and Echoview Region object

Value

a string containing the class of the region

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVFindRegionByName

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

ev.region <- EVFindRegionByName(EVFile, "Region1")
EVFindRegionClass(ev.region)

## End(Not run)
```

EVGetCalibrationFileName *Gets the calibration file name of a fileset*

Description

This function gets the calibration file name of a filesset using COM scripting

Usage

```
EVGetCalibrationFileName(EVFile, filesetName)
```

Arguments

EVFile	An Echoview file COM object
filesetName	a string containing the name of the Echoview fileset

Value

an object: returns the calibration file name

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVGetCalibrationFileName(EVFile = EVFile, filesetName = "038-120-200")

## End(Not run)
```

EVImportLine *Imports an Echoview Line file This function imports an Echoview line file (.evl) using COM scripting*

Description

Imports an Echoview Line file This function imports an Echoview line file (.evl) using COM scripting

Usage

```
EVImportLine(EVFile, pathAndFn = NULL, lineName = NULL)
```

Arguments

EVFile	An Echoview file COM object
pathAndFn	string path and filename to .evl file.
lineName	= NULL optional line name for imported line.

Value

a list object with two elements- [[1]] imported EV line COM object, and [[2]] function message for log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

#Example 1 - import a .evl line file
EVImportLine(EVFile, pathAndFn = '~~/KAOS/lineKAOS.evl')

#Example 2 - rename the imported line
EVImportLine(EVFile, pathAndFn='~~/KAOS/lineKAOS.evl', lineName='Line6' )

## End(Not run)
```

EVImportRegionDef *Import an Echoview region definitions file (.evr)*

Description

This function imports a region definitions file (.evr) using COM scripting

Usage

```
EVImportRegionDef(EVFile, evrFile, regionName)
```

Arguments

EVFile	An Echoview file COM object
evrFile	An Echoview region definitions file (.evr) path and name
regionName	The name of the Echoview region

Value

A list object with one element. $msg message for processing log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run: EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVImportRegionDef(EVFile = EVFile, evrFile = '~~/KAOS/off transect regions/20030114_1200000000.evr', regionN

## End(Not run)
```

EVIntegrationByRegionsByCellsExport *Export integration by regions by cells from an Echoview acoustic variable*

Description

This function performs integration by regions by cells for a specified region class and exports the results using COM scripting.

Usage

```
EVIntegrationByRegionsByCellsExport(EVFile, acoVarName, regionClassName,
  exportFn, dataThreshold = NULL)
```

Arguments

EVFile	An Echoview file object
acoVarName	A string containing the name of an Echoview acoustic variable
regionClassName	A string containing the name of an Echoview region class
exportFn	export filename and path
dataThreshold	An optional data threshold for export

Value

a list object with one element, $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVIntegrationByRegionsByCellsExport(EVFile, "120 aggregations", "aggregations", exportFn = "~~/KAOS/EVIntegr

## End(Not run)
```

`EVIntegrationByRegionsExport` *Export integration by regions from an Echoview acoustic variable*

Description

This function performs integration by regions and exports the results using COM scripting.

Usage

```
EVIntegrationByRegionsExport(EVFile, acoVarName, regionClassName, exportFn,
  dataThreshold = NULL)
```

Arguments

`EVFile`	An Echoview file object
`acoVarName`	A string containing the name of an Echoview acoustic variable
`regionClassName`	A string containing the name of an Echoview region class
`exportFn`	export filename and path
`dataThreshold`	An optional data threshold for export

Value

a list object with one element, $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

`EVOpenFile`

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVIntegrationByRegionsExport(EVFile, "120 aggregations", "aggregations", exportFn = "~~/KAOS/EVIntegrationBy

## End(Not run)
```

EVminThresholdSet *Sets minimum data threshold for a variable object*

Description

This function sets the minimum data threshold

Usage

```
EVminThresholdSet(varObj, thres)
```

Arguments

varObj	An Echoview variable object
thres	The new threshold to be set

Value

a list object with one element. $thresholdSettings: The new threshold settings

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVAcoVarNameFinder

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
varObj <- EVAcoVarNameFinder(EVFile, "38 seabed and surface excluded")$EVVar
EVminThresholdSet(varObj, -80)

## End(Not run)
```

EVNewAcousticVar *Add a new acoustic variable*

Description

This function adds a new acoustic variable using COM scripting

Usage

```
EVNewAcousticVar(EVFile, oldVarName, enum)
```

Arguments

EVFile	An Echoview file COM object
oldVarName	a string containing the name of the acoustic variable to base the new variable on
enum	Enum code for operator. See Echoview help file on EOperator for enum codes.

Value

an object: returns the new variable

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
#create a 7x7 convolution of a variable
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVNewAcousticVar(EVFile = EVFile, oldVarName = "38 seabed and surface excluded", enum = 43)

## End(Not run)
```

EVNewFile *Create a new Echoview file (.EV)*

Description

This function creates a new Echoview file (.EV) via COM scripting which may be created from a template file if available

Usage

```
EVNewFile(EVAppObj, templateFn = NULL)
```

Arguments

EVAppObj	An EV application COM object arising from the call COMCreate('EchoviewCom.EvApplication')
templateFn	full path and filename for an Echoview template

Value

a list object with two elements. $EVFile: EVFile COM object for the newly created Echovie file, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

Examples

```
## Not run:
EVAppObj=COMCreate('EchoviewCom.EvApplication')
EVFile=EVNewFile(EVAppObj)$EVFile

## End(Not run)
```

EVNewFixedDepthLine *Creates a new fixed depth Echoview line*

Description

This function creates a new Echoview line at a fixed depth using COM scripting

Usage

```
EVNewFixedDepthLine(EVFile, depth, lineName)
```

Arguments

EVFile	An Echoview file COM object
depth	an integer specifying the fixed depth of the new line
lineName	a string containing the name for the new line

Value

returns the EV Line object

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

#create a new line at 50m depth named "testline"
newLine <- EVNewFixedDepthLine(EVFile = EVFile, depth = 50, lineName = "testline")

## End(Not run)
```

EVNewLineRelativeRegion

Creates a new line relative region in the current variable

Description

This function creates a new line relative region in the current variable using COM scripting. The upper and lower depths are specified using Echoview line objects (these must already exist). Left and right bounds are optionally specified using ping number.

Usage

```
EVNewLineRelativeRegion(EVFile, varName, regionName, line1, line2,
  firstPing = NA, lastPing = NA)
```

Arguments

EVFile	An Echoview file COM object
varName	a string containing the name of the acoustic variable to create the region in
regionName	a string containing the name to assign to the new region
line1	a string containing the name of the upper line limit
line2	a string containing the name of the lower line limit
firstPing	an optional integer for the ping to begin the region at
lastPing	an optional integer for the ping to end the region at

Value

returns the EV Region object

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

#create a region between pings 1 - 100 and depths 20-250m
newRegion <- EVNewLineRelativeRegion(EVFile, "38 seabed and surface excluded", "test", "Fixed depth 6 m", "Fi

#create an unbounded region between depths 250-750m
newRegion <- EVNewLineRelativeRegion(EVFile, "38 seabed and surface excluded", "test", "Fixed depth 6 m", "Fi

## End(Not run)
```

EVNewRegionClass *Create a new Echoview region class*

Description

This function creates a new region class using COM scripting

Usage

```
EVNewRegionClass(EVFile, className)
```

Arguments

EVFile	An Echoview file COM object
className	The name of the new Echoview region class

Value

A list object with one element. $msg message for processing log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVNewRegionClass(EVFile = EVFile, className = 'test_class')

## End(Not run)
```

EVOpenFile *Open an existing Echoview file (.EV)*

Description

This function opens an existing Echoview (.EV) file using COM scripting.

Usage

```
EVOpenFile(EVAppObj, fileName)
```

Arguments

EVAppObj	An EV application COM object arising from the call COMCreate('EchoviewCom.EvApplication')
fileName	An Echoview file path and name.

Value

a list object with two elements. $EVFile: EVFile COM object, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')

## End(Not run)
```

EVRegionClassFinder *Find an Echoview region class by name*

Description

This function finds an echoview region class by name.

Usage

```
EVRegionClassFinder(EVFile, regionClassName)
```

Arguments

EVFile	An Echoview file COM object
regionClassName	A string containing the name of an Echoview region

Value

a list object with two elements. $regionClass: The class of the region, and $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
aggregationsClass <- EVRegionClassFinder(EVFile, "aggregations")$regionClass

## End(Not run)
```

EVRenameLine *Renames an Echoview Line object*

Description

This function renames an Echoview line object using COM scripting

Usage

```
EVRenameLine(EVFile, evLine, newName)
```

Arguments

EVFile	An Echoview file COM object
evLine	an Echoview line object
newName	a string containing the new name for the line

Value

a list object with one element- fucntion message for log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVNewFixedDepthLine EVFindLineByName

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

testline <- EVNewFixedDepthLine(EVFile = EVFile, depth = 50, lineName = "testline")
EVRenameLine(EVFile = EVFile, evLine = testline, newName = "line40")

## End(Not run)
```

EVSaveAsFile *Perform save as operation on an open Echoview file (.EV)*

Description

This function performs a save as operation on an existing Echoview (.EV) file using COM scripting.

Usage

```
EVSaveAsFile(EVFile, fileName)
```

Arguments

EVFile	An Echoview file COM object
fileName	An Echoview file path and name.

Value

a list object with two elements. $chk: Boolean check indicating if the file was successfully saved; $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVSaveFile EVCloseFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVSaveAsFile(EVFile = EVFile, fileName = '~~/KAOS/KAOStemplate_test.EV')

## End(Not run)
```

EVSaveFile *Save an open Echoview file (.EV)*

Description

This function saves an existing Echoview (.EV) file using COM scripting.

Usage

```
EVSaveFile(EVFile)
```

Arguments

EVFile	An Echoview file COM object

Value

a list object with two elements. $chk: Boolean check indicating if the file was successfully saved; $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVSaveFile(EVFile)

## End(Not run)
```

EVSchoolsDetect *Schools Detection in Echoview*

Description

This function performs schools detection in Echoview using COM scripting.

Usage

```
EVSchoolsDetect(EVFile, acoVarName, outputRegionClassName,
  deleteExistingRegions, distanceMode, maximumHorizontalLink,
  maximumVerticalLink, minimumCandidateHeight, minimumCandidateLength,
  minimumSchoolHeight, minimumSchoolLength, dataThreshold)
```

Arguments

EVFile	An Echoview file COM object
acoVarName	A string containing the name of the acoustic variable to perform the analysis on
outputRegionClassName	A string containing the name of the output region
deleteExistingRegions	Logical TRUE or FALSE
distanceMode	for schools detection (see Echoview help).
maximumHorizontalLink	The maximum horizontal link in meters

`maximumVerticalLink`
: The maximum vertical link in meters

`minimumCandidateHeight`
: The minimum candidate height in meters

`minimumCandidateLength`
: the minimum candidate length in meters

`minimumSchoolHeight`
: The minimum school height in meters

`minimumSchoolLength`
: The minimum school length in meters

`dataThreshold` minimum integration threshold (units: dB re 1m^-1)

Value

a list object with four elements. $nbrOfDetectedschools, $thresholdData, $schoolsSettingsData, and $msg: message for processing log

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
EVSchoolsDetect(EVFile = EVFile,
                acoVarName='120 7x7 convolution',
                outputRegionClassName = 'aggregations',
                deleteExistingRegions = TRUE,
                distanceMode = "GPS distance",
                maximumHorizontalLink = 15, #m
                maximumVerticalLink = 5,#m
                minimumCandidateHeight = 1, #m
                minimumCandidateLength = 10, #m
                minimumSchoolHeight = 2, #m
                minimumSchoolLength = 15, #m
                dataThreshold = -80)

## End(Not run)
```

EVSchoolsDetSet *Change schools detection settings*

Description

This function changes schools detection settings for an acoustic variable using COM scripting

Usage

```
EVSchoolsDetSet(EVFile, varObj, distanceMode, maximumHorizontalLink,
  maximumVerticalLink, minimumCandidateHeight, minimumCandidateLength,
  minimumSchoolHeight, minimumSchoolLength)
```

Arguments

EVFile	An Echoview file COM object
varObj	the EV acoustic object to change schools detection parameters for
distanceMode	which distance mode to use
maximumHorizontalLink	maximum linking distance for a swarm
maximumVerticalLink	maximum vertical linking distance for a school
minimumCandidateHeight	minimum candidate height
minimumCandidateLength	minimum candidate length
minimumSchoolHeight	minimum school height
minimumSchoolLength	minimum school length

Value

a list object with two elements.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile EVAcoVarNameFinder

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile
varObj <- EVAcoVarNameFinder(EVFile, "120 7x7 convolution")$EVVar
changeSettings <- EVSchoolsDetSet(EVFile, varObj, distanceMode = "GPS distance",
                                  maximumHorizontalLink = 10,
                                  maximumVerticalLink = 5,
                                  minimumCandidateHeight = 2,
                                  minimumCandidateLength = 3,
                                  minimumSchoolHeight = 4,
                                  minimumSchoolLength = 2)

## End(Not run)
```

EVShiftRegionDepth *Change the depth of an Echoview Region*

Description

This function shifts the depth of an echoview region using COM scripting. Vertical size of the region and depth offset can be changed.

Usage

```
EVShiftRegionDepth(EVFile, regionName, depthMultiply, depthAdd)
```

Arguments

EVFile	An Echoview file COM object
regionName	a string containing the name of the Echoview region
depthMultiply	a numeric value to multiply the vertical size of the region by
depthAdd	a numeric value to offset the region depth by. Positive = decrease depth; Negative = increase depth

Value

a list object with one element, $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

#Double region vertical size and decrease depth by 100m
EVShiftRegionDepth(EVFile, "testregion", 2, 100)

#Triple region vertical size without changing depth offset
EVShiftRegionDepth(EVFile, "testregion", 3, 0)

#Change region depth offset by -50m without changing vertical size
EVShiftRegionDepth(EVFile, "testregion", 1, -50)

## End(Not run)
```

EVShiftRegionTime *Change the time of an Echoview Region*

Description

This function shifts the time of an echoview region using COM scripting

Usage

```
EVShiftRegionTime(EVFile, regionName, days = 0, hours = 0, minutes = 0,
  seconds = 0, milliseconds = 0)
```

Arguments

EVFile	An Echoview file COM object
regionName	a string containing the name of the Echoview region
days	an integer value specifying days to add (positive) or subtract (negative). Default = 0
hours	an integer value specifying hours to add (positive) or subtract (negative). Default = 0
minutes	an integer value specifying minutes to add (positive) or subtract (negative). Default = 0
seconds	an integer value specifying seconds to add (positive) or subtract (negative). Default = 0
milliseconds	an integer value specifying milliseconds to add (positive) or subtract (negative). Default = 0

Value

a list object with one element, $msg: message for processing log.

References

http://support.echoview.com/WebHelp/Echoview.htm/

See Also

EVOpenFile

Examples

```
## Not run:
EVAppObj <- COMCreate('EchoviewCom.EvApplication')
EVFile <- EVOpenFile(EVAppObj,'~~/KAOS/KAOStemplate.EV')$EVFile

#Shift the region time by 10 seconds
EVShiftRegionTime(EVFile, "testregion", seconds = 10)

#Subtract 1 hour from the region time
EVShiftRegionTime(EVFile, "testregion", hours = -1)

## End(Not run)
```

`exportMIF` *Write a map info file for import into Echoview.*

Description

This function writes a polygon MIF file for import into Echoview and will typically be used to export survey line transects from, for example, centreLawnOnPosition.

Usage

```
exportMIF(coords, pathAndFileName, pointNameExport = FALSE,
  pointNameScaleFactor = c(0.05, 0.03))
```

Arguments

`coords`	a set of coordinates as a two column (longitude and latitude) matrix.
`pathAndFileName`	character string MIF file export path and filename
`pointNameExport`	Boolean (default=FALSE) export the name of each point.
`pointNameScaleFactor`	=c(0.005,0.03) see details

Details

Exporting the name of each point uses the row name in the `coords` argument.

The `pointNameScaleFactor` argument is a two element numeric vector specifying the bounding box for each point name. The bounding box is calculated for point name i by specifying the lower left hand corner of the point name position, with the upper right hand corner specified as c(lon_i,lat_i)+diff(range(lon_1..n,lat_1..n))*pointNameScaleFactor where n is the total number of points i.e nrow(coords)

Value

Nothing

Examples

```
## Not run:
coords=centreLawnOnPosition(centreLon=-33,centreLat=-55.5,lineLengthkm=100,lineSpacingkm=10,
startBearingdeg=30,numOfLines=50)
exportMIF(coords=coords,pathAndFileName='c:\\Users\\martin_cox\\Documents\\test4.mif')

## End(Not run)
```

lawnSurvey *Generate a coordinate list for a regular rectangular survey*

Description

The coordinate list is generated in degrees decimal degree format (dd.ddd), with Southern hemisphere denoted by negative numbers. Transect length and inter-transect spacing are specified in km and bearings in degress where North 0 deg, East 90 deg, South 180 deg and West 270 deg.

Usage

```
lawnSurvey(startLon, startLat, lineLengthkm, lineSpacingkm, startBearingdeg,
  numOfLines)
```

Arguments

startLon	start longitude of survey.
startLat	start latitude of survey.
lineLengthkm	transect line length in km.
lineSpacingkm	inter-transect spacing in km.
startBearingdeg	Orientation of survey grid in degrees.
numOfLines	Number of transects.

Value

Geographical coordinate list of start and end of line positions

Author(s)

Martin Cox <martin.cox@aad.gov.au>

See Also

zigzagSurvey

Examples

```
## Not run:
(coords=lawnSurvey(startLon=-170,startLat=-60,lineLengthkm=2,lineSpacingkm=0.5,
startBearingdeg=30,numOfLines=5))
plot(0,0,xlim=range(coords[,1]),ylim=range(coords[,2]),type='n',xlab='Longitude, deg',ylab='Latitude, deg')
arrows(x0=coords[1:(nrow(coords)-1),1], y0=coords[1:(nrow(coords)-1),2],
      x1 = coords[2:nrow(coords),1], y1 = coords[2:nrow(coords),2])
text(coords,row.names(coords),cex=0.6)
points(coords[1,1],coords[1,2],col='blue',pch=17,cex=2)
points(coords[nrow(coords),1],coords[nrow(coords),2],col='blue',pch=15,cex=2)
legend('topright',c('Beginning','End'),col='blue',pch=c(17,15))

## End(Not run)
```

msDATEConversion *Convert a Microsoft DATE object to a human readable date and time*

Description

Time stamps in Echoview, such as start and end times of, for example, invididual regions, use the Microsoft DATE format. This function converts the Microsoft DATE object to a human readable date and time. NB no time zone is returned.

Usage

```
msDATEConversion(dateObj)
```

Arguments

dateObj	a Microsoft date object

Value

time stamp in the format yyyy-mm-dd hh:mm:ss

See Also

www.echoview.com

zigzagSurvey *Generate a coordinate list for a zig-zag survey*

Description

The coordinate list is generated in degrees decimal degree format (dd.ddd), with Southern hemisphere denoted by negative numbers. Transect length are specified in km and bearings in degress where North 0 deg, East 90 deg, South 180 deg and West 270 deg.

Usage

```
zigzagSurvey(startLon, startLat, lineLengthkm, startBearingdeg, rotationdeg,
  numOfLines, proj4string = CRS("+proj=longlat +datum=WGS84"),
  unrotated = FALSE)
```

Arguments

startLon	start longitude of survey.
startLat	start latitude of survey.
lineLengthkm	transect line length in km.
startBearingdeg	Bearing of each transect in degrees.
rotationdeg	rotation angle for entire survey pattern.
numOfLines	Number of transects.
proj4string	projection string of class CRS-class
unrotated	FALSE return rotated coordinates TRUE list of rotated and unrotated coordinates.

Value

Geographical coordinates of start and end of line positions. unrotated=TRUE list of rotated and unrotated coordinates

Author(s)

Martin Cox <martin.cox@aad.gov.au>

See Also

zigzagSurvey

Examples

```
## Not run:
coords=zigzagSurvey(startLon=-100,startLat=-60,lineLengthkm=2,startBearingdeg=30,
rotationdeg=10,numOfLines=11)
plot(0,0,xlim=range(coords[,1]),ylim=range(coords[,2]),type='n',xlab='Longitude, deg',ylab='Latitude, deg')
arrows(x0=coords[1:(nrow(coords)-1),1], y0=coords[1:(nrow(coords)-1),2],
      x1 = coords[2:nrow(coords),1], y1 = coords[2:nrow(coords),2])
text(coords,row.names(coords),cex=0.6)
points(coords[1,1],coords[1,2],col='blue',pch=17,cex=2)
points(coords[nrow(coords),1],coords[nrow(coords),2],col='blue',pch=15,cex=2)
legend('topright',c('Beginning','End'),col='blue',pch=c(17,15))
#Use unrotated=TRUE and check coordinates in the returned coordinate list object are identical:
coordL= zigzagSurvey(startLon=-100,startLat=-60,lineLengthkm=2,startBearingdeg=30,
rotationdeg=0,numOfLines=11,unrotated=TRUE)
identical(coordL[[1]],coordL[[2]])
#display rotated and unrotated coordinates"
coordL= zigzagSurvey(startLon=-100,startLat=-60,lineLengthkm=2,startBearingdeg=30,
rotationdeg=5,numOfLines=11,unrotated=TRUE)
coords=coordL$unrotatedGeogs; coordsRotate=coordL$rotatedGeogs
plot(0,0,xlim=range(c(coords[,1],coordsRotate[,1])),
ylim=range(c(coords[,2],coordsRotate[,2])),type='n',xlab='Longitude, deg',ylab='Latitude, deg')
arrows(x0=coords[1:(nrow(coords)-1),1], y0=coords[1:(nrow(coords)-1),2],
      x1 = coords[2:nrow(coords),1], y1 = coords[2:nrow(coords),2])
arrows(x0=coordsRotate[1:(nrow(coordsRotate)-1),1], y0=coordsRotate[1:(nrow(coordsRotate)-1),2],
     x1 = coordsRotate[2:nrow(coordsRotate),1], y1 = coordsRotate[2:nrow(coordsRotate),2],col='blue')
legend('bottomleft',c('Unrotated','Rotated'),lty=1,col=c(1,'blue'))

## End(Not run)
```

Index