Package 'PAMscapes'

September 22, 2024

Title Tools for Summarising and Analysing Soundscape Data

Version 0.7.0

Description A variety of tools relevant to the analysis of marine soundscape data. There are tools for downloading AIS (automatic identification system) data from Marine Cadastre <https://hub.marinecadastre.gov>, connecting AIS data to GPS coordinates, plotting summaries of various soundscape measurements, and downloading relevant environmental variables (wind, swell height) from the National Center for Atmospheric Research data server <https://rda.ucar.edu/datasets/ds084.1/>. Most tools were developed to work well with output from 'Triton' software, but can be adapted to work with any similar measurements.

License GNU General Public License

Encoding UTF-8

RoxygenNote 7.3.1

Depends R (>= 3.5.0)

Imports dplyr, rlang, ggplot2, lubridate, scales, tidyr, httr, data.table, geosphere, sf, PAMmisc, ncdf4, tdigest, purrr, shiny

Suggests testthat

NeedsCompilation no

Author Taiki Sakai [aut, cre], Anne Simonis [ctb], Shannon Rankin [ctb], Megan McKenna [ctb], Kaitlin Palmer [ctb]

Maintainer Taiki Sakai <taiki.sakai@noaa.gov>

Repository CRAN

Date/Publication 2024-09-22 17:20:02 UTC

Contents

`addAIS` *Add AIS Data to Dataframe*

Description

Adds matching AIS data downloaded from Marine Cadastre to a dataframe containing location information

Usage

```
addAIS(
  x,
  ais,
  interpType = c("all", "close", "none"),
  interpTime = 0,
  interpCols = NULL
)
```

Arguments

`x`	a dataframe with `UTC`, `Latitude`, and `Longitude` columns
`ais`	AIS data created using the readLocalAIS function
`interpType`	one of `c('all', 'close', 'none')`, the type of time interpolation to apply to `x`. Often the time scale of points in `x` is much longer than the points in `ais`, which can result in awkward looking AIS paths. `'all'` will interpolate all points in `x` to a smaller timescale. `'close'` will interpolate only time ranges in `ais` marked as `inDist` by readLocalAIS. `'none'` will apply no interpolation

interpTime	time (seconds) between new UTC points. If 0 (default), no interpolation will be done
interpCols	names of any extra columns to interpolate (other than Latitude and Longitude)

Value

a dataframe with AIS data added, will contain more rows than x if ais has more than one vessel. If any interpolation is applied, any non-constant columns not specified to interpCols will be removed

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
gps <- data.frame(Latitude=c(33.2, 33.5,33.6),
                  Longitude=c(-118.1, -118.4, -119),
                  UTC=as.POSIXct(
                    c('2022-04-28 05:00:00',
                      '2022-04-28 10:00:00',
                      '2022-04-28 20:00:00'),
                    tz='UTC'))
ais <- readLocalAIS(gps, aisDir=system.file('extdata/ais', package='PAMscapes'), distance=20e3)
gpsNoInterp <- addAIS(gps, ais, interpType='none')
str(gpsNoInterp)
gpsClose <- addAIS(gps, ais, interpType='close')
str(gpsClose)
gpsAllInterp <- addAIS(gps, ais, interpType='all')
str(gpsAllInterp)
```

---

addAISSummary *Add AIS Data Summary to Dataframe*

Description

Adds a summary of matching AIS data for nearby vessels to a data. Information added includes number of vessels, distance to nearby vessels, and average speed of nearby vessels

Usage

```
addAISSummary(x, ais, distance = 10000)
```

Arguments

x	a dataframe with UTC, Latitude, and Longitude columns
ais	AIS data created using the readLocalAIS function. Can also be a character listing the directory of AIS
distance	distance (meters) within locations in x to mark as "nearby"

Value

a dataframe with AIS summary data added. Will contain new columns

nShips the number of ships within "distance" at this time

meanDist average distance of nearby ships, NA if none

meanSOG average speed over ground of nearby ships, NA if none

closeDist distance of the closest ship, NA if none

closeSOG speed over ground of closest ship, NA if none

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
gps <- data.frame(Latitude=c(33.2, 33.5,33.6),
                  Longitude=c(-118.1, -118.4, -119),
                  UTC=as.POSIXct(
                    c('2022-04-28 05:00:00',
                      '2022-04-28 10:00:00',
                      '2022-04-28 20:00:00'),
                    tz='UTC'))
ais <- readLocalAIS(gps, system.file('extdata/ais', package='PAMscapes'))
aisSummary <- addAISSummary(gps, ais)
str(aisSummary)
```

---

binSoundscapeData *Summarise Soundscape Data by Time Bin*

Description

Bins soundscape measurements by a unit of time and summarises them using a function (usually the median)

Usage

```
binSoundscapeData(x, bin = "1hour", FUN = median)
```

Arguments

x	a data.frame of soundscape metric data read in with checkSoundscapeInput
bin	amount of time to bin data by, format can be "#Unit" e.g. '2hour' or '1day'
FUN	summary function to apply to data in each time bin

Value

a summarised version of the input data.frame `x`

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

`checkSoundscapeInput` *Check Proper Formatting for Soundscape Inputs*

Description

Reads and checks data to ensure formatting will work for other `PAMscapes` functions. Will read and check the formatting of CSV files, or check the formatting of dataframes. Can also read in MANTA NetCDF files and format the data appropriately.

Usage

```
checkSoundscapeInput(
  x,
  needCols = c("UTC"),
  skipCheck = FALSE,
  timeBin = NULL,
  binFunction = median,
  tz = "UTC"
)
```

Arguments

`x`	a dataframe, path to a CSV file, or path to a MANTA NetCDF file. If `x` is a vector of file paths then all will be read in and combined
`needCols`	names of columns that must be present in `x`, if any are missing will trigger an error
`skipCheck`	logical flag to skip some data checking, recommended to keep as `FALSE`
`timeBin`	amount of time to bin data by, format can be "#Unit" e.g. `'2hour'` or `'1day'`
`binFunction`	summary function to apply to data in each time bin
`tz`	timezone of the data being loaded, will be converted to UTC after load

Details

Files created by MANTA and Triton software will be reformatted to have consisitent formatting. The first column will be renamed to "UTC", and columns containing soundscape metrics will be named using the convention "TYPE_FREQUENCY", e.g. "HMD_1", "HMD_2" for Manta hybrid millidecade mesaurements.

Inputs from sources other than MANTA or Triton can be accepted in either "wide" or "long" format. Wide format must follow the conventions above - first column "UTC", other columns named by "TYPE_FREQUENCY" where TYPE is consistent across all columns and FREQUENCY is in Hertz. Long format data must have the following columns:

"UTC" - time of the measurement, in UTC timezone

"type" - the type of soundscape measurement e.g. PSD or OL, must be the same for all

"frequency" - the frequency of the measurement, in Hertz

"value" - the soundscape measurement value, usually dB

Value

a dataframe

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
manta <- checkSoundscapeInput(system.file('extdata/MANTAExampleSmall1.csv', package='PAMscapes'))
str(manta)
ol <- checkSoundscapeInput(system.file('extdata/OLSmall.csv', package='PAMscapes'))
str(ol)
psd <- checkSoundscapeInput(system.file('extdata/PSDSmall.csv', package='PAMscapes'))
str(psd)
```

---

createOctaveLevel *Create Octave Level Measurements*

Description

Creates octave or third octave level measurements from finer resolution soundscape metrics, like Power Spectral Density (PSD) or Hybrid Millidecade (HMD) measures

Usage

```
createOctaveLevel(
  x,
  type = c("ol", "tol"),
  freqRange = NULL,
  method = c("sum", "mean", "median")
)
```

Arguments

x	dataframe of soundscape metrics
type	either 'ol' to create octave level or 'tol' to create third octave level measures
freqRange	a vector of the minimum and maximum center frequencies (Hz) desired for the output. If NULL, full available range of frequencies will be used.
method	the summary method to apply to soundscape metrics within the octave band, one of 'sum' or 'mean'. The default 'sum' should be used in almost all cases.

Value

a dataframe with summarised octave level band measurements

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
psd <- checkSoundscapeInput(system.file('extdata/PSDSmall.csv', package='PAMscapes'))
str(psd)
tol <- createOctaveLevel(psd, type='tol')
str(tol)
ol <- createOctaveLevel(tol, type='ol')
str(ol)
```

---

downloadMarCadAIS *Download AIS Data from Marine Cadastre*

Description

Downloads daily AIS files from https://hub.marinecadastre.gov/pages/vesseltraffic covering the date range present in input data

Usage

```
downloadMarCadAIS(x, outDir, overwrite = FALSE, unzip = TRUE, verbose = TRUE)
```

Arguments

x	a dataframe with column UTC in POSIXct format
outDir	directory to save the downloaded files
overwrite	logical flag to overwrite existing data. Recommended to be FALSE to avoid re-downloading large files unnecessarily
unzip	logical flag to unzip downloaded files. Original downloads from Marine Cadastre come as large .zip
verbose	logical flag to print messages about download progress

Value

a vector of the paths to the downloaded .zip files, any days that were unable to download will be NA

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
gps <- data.frame(Latitude=c(33.2, 33.5,33.6),
                  Longitude=c(-118.1, -118.4, -119),
                  UTC=as.POSIXct(
                    c('2022-04-28 05:00:00',
                      '2022-04-28 10:00:00',
                      '2022-04-28 20:00:00'),
                    tz='UTC'))
tempDir <- tempdir()
# Commented out because running this will download
# a ~500mb file
# marcadFiles <- downloadMarCadAIS(gps, outDir=tempDir)
```

---

loadMantaNc *Load MANTA NetCDF File*

Description

Reads in hybrid millidecade data from a MANTA NetCDF output file and formats it into the dataframe format required for use in other PAMscapes functions

Usage

```
loadMantaNc(x, keepQuals = c(1))
```

Arguments

x	path to .nc file
keepQuals	quality flag values to keep. Accepts vector of integers from (1, 2, 3, 4) corresponding to flag labels "Good", "Not evaluated/Unknown", "Compromised/Questionable", and "Unusable/Bad". HMD levels for points with data quality flags outside of keepQuals will be marked as NA.

Value

a dataframe with first column UTC and other columns named HMD_Frequency

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
# no sample NetCDF provided (too large)

manta <- loadMantaNc('MANTA.nc')
```

---

markNA *Mark NA Values by Time and Frequency*

Description

Marks values within a soundscape dataframe as NA according to provided time and (optionally) frequency values

Usage

```
markNA(x, na, by = NULL)
```

Arguments

x	dataframe of soundscape data to mark NAs in
na	dataframe listing areas to mark NA. Must have columns start and end in UTC listing time ranges. Can also have columns freqMin and freqMax to also have accompanying frequency ranges, otherwise all frequency values within the time range will be set to NA
by	optional column name in both x and na if only certain rows of na should apply to certain rows of x (e.g. if these contain multiple deployments overlapping in time, a "DeploymentName" column can be used to only mark appropriate times)

Value

same dataframe as x but with some values replaced with NA

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
manta <- checkSoundscapeInput(system.file('extdata/MANTAExampleSmall1.csv', package='PAMscapes'))
naDf <- data.frame(start=min(manta$UTC),
                   end=max(manta$UTC),
                   freqMin=100,
                   freqMax=500)
plotHourlyLevel(manta)
plotHourlyLevel(markNA(manta, na=naDf))
```

matchGFS *Match GFS Environmental Data*

Description

Downloads and matches wind and precipitation data from the Global Forecast System (GFS) weather model. Data is downloaded from the National Center for Atmospheric Research data server https://rda.ucar.edu/datasets/ds084.1/. The particular GFS dataset downloaded is the closest "forecast" dataset to the particular time (e.g. .f000 or .f003)

Usage

```
matchGFS(x)
```

Arguments

x	a dataframe with columns UTC, Latitude and Longitude to add environmental data to

Value

a dataframe with wind (m/s) and precipitation rate (kg/m^2/s) columns added

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
# API response may be slow for this example

gps <- data.frame(Latitude=c(33.2, 33.5,33.6),
                  Longitude=c(-118.1, -118.4, -119),
                  UTC=as.POSIXct(
                    c('2022-04-28 05:00:00',
                      '2022-04-28 10:00:00',
                      '2022-04-28 20:00:00'), tz='UTC'))
gps <- matchGFS(gps)
```

matchSeascape *Match Seascape Class to Data*

Description

Downloads and matches relevant Seascape class data from the ERDDAP (Environmental Research Division's Data Access Program) server at https://cwcgom.aoml.noaa.gov/erddap/index.html. More information on theclasses can be found on the help page for the seascapeR package https://marinebon.github.io/seascapeR/index.html.

Usage

```
matchSeascape(x, type = c("monthly", "8day"), progress = TRUE)
```

Arguments

x	a dataframe with columns UTC, Latitude and Longitude to add environmental data to
type	the type of seascape data to download, one of "monthly" or "8day"
progress	logical flag whether or not to show download progress

Details

This function is just a wrapper around matchEnvData pointing to the specific base URL and dataset ID relevant for seascape data

Value

the same dataframe as x, but with new columns seascapeClass and seascapeProb representing the "CLASS" and "P" variables from the dataset

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

plotAcousticScene *Plot Acoustic Scene*

Description

Plots a representation of the acoustic scene using detections in data. Frequency ranges for detections are taken from user input and displayed as different colored bars

Usage

```
plotAcousticScene(
  x,
  freqMap,
  typeCol = "species",
  title = NULL,
  bin = "1day",
  scale = c("log", "linear"),
  freqMin = NULL,
  freqMax = NULL,
  fill = TRUE,
  alpha = 1,
  returnData = FALSE,
  add = FALSE
)
```

Arguments

x	dataframe of detections, must have column UTC and a column to connect detection types to the frequency type map
freqMap	a dataframe listing frequency ranges to use for various detection types in x. Must have columns type, freqMin (Hz), freqMax (Hz), and optionally color (color to use for this type of detection on plot)
typeCol	column name in x that matches names in type column in freqMap
title	optional title to use for the plot
bin	time bin to use for plotting time axis. Each detection will be displayed as covering this amount of time
scale	one of log or linear, the frequency scale for the plot
freqMin	optional minimum frequency for plot, useful for log scale
freqMax	optional maximum frequency for plot
fill	logical flag if TRUE then filled boxes will be plotted, if FALSE then only outlines will be plotted
alpha	transparency percentage for plotting, values less than 1 will allow multiple overlapping colors to be seen
returnData	if TRUE then no plot will be generated, instead the dataframe that would normally be used to make the plot will be returned
add	logical flag if FALSE plots normally if TRUE then the output can be (maybe) added to an existing ggplot object

Value

a ggplot object

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
detDf <- data.frame(
   UTC=as.POSIXct(c('2023-01-01 00:00:00',
                    '2023-01-03 00:00:00',
                    '2023-01-02 12:00:00',
                    '2023-01-04 00:00:00'),
                  tz='UTC'),
  species = c('Dolphin', 'Whale', 'Whale', 'Dolphin'))
freqMap <- data.frame(type=c('Dolphin', 'Whale'),
                      freqMin=c(10e3, 100),
                      freqMax=c(30e3, 400),
                      color=c('darkgreen', 'blue'))
plotAcousticScene(detDf, freqMap=freqMap, typeCol='species', bin='1day')
```

`plotHourlyLevel` *Plot Hourly Sound Level*

Description

Plots a heatmap of summarised sound levels. Y-axis is hour of the day, X-axis is frequency bin. Plotted values are the median of the `value` column for each hour/frequency pairing across the dataset. This function is designed to work with sound level outputs with consistent frequency bins measured across time

Usage

```
plotHourlyLevel(
  x,
  title = NULL,
  units = NULL,
  scale = c("log", "linear"),
  freqMin = NULL,
  dbRange = NULL,
  toTz = "UTC",
  cmap = viridis_pal()(25),
  returnData = FALSE
)
```

Arguments

`x`	a dataframe with columns `UTC`, `frequency`, and `value`
`title`	title for the plot. If `NULL` (default) it will use the first value in the `type` column of `x` (if present)
`units`	name of units for plot labeling, default is taken from common soundscape units
`scale`	one of `'log'` or `'linear'` for the scale of the frequency axis

freqMin	minimum frequency for the plot range, if desired to be different than the minimum frequency of the data
dbRange	range of dB values to plot
toTz	timezone to use for the time axis (input data must be UTC). Specification must be from OlsonNames
cmap	color palette map to use for plot, default is viridis_pal
returnData	if TRUE then no plot will be generated, instead the dataframe that would normally be used to make the plot will be returned

Value

a ggplot object

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
plotHourlyLevel(system.file('extdata/OLSmall.csv', package='PAMscapes'))
```

---

plotLTSA *Plot Long-Term Spectral Average (LTSA)*

Description

Creates a long-term spectral average (LTSA) style plot of the data, a plot where the x-axis is time and the y-axis is frequency. Color represents the magnitude of sound. In order to compress the time axis, data are binned into time chunks and the median value within that time bin is displayed

Usage

```
plotLTSA(
  x,
  bin = "1hour",
  scale = c("log", "linear"),
  title = NULL,
  freqRange = NULL,
  dbRange = NULL,
  units = NULL,
  cmap = viridis_pal()(25),
  toTz = "UTC",
  alpha = 1,
  maxBins = 800,
  returnData = FALSE
)
```

Arguments

x	a soundscape metric file that can be read in with checkSoundscapeInput, or a dataframe with UTC, frequency, and value
bin	amount of time to bin for each LTSA slice, format can be "#Unit" e.g. '2hour' or '1day'
scale	scaling for frequency axis, one of log or linear
title	optional title for plot
freqRange	if not NULL, a vector of two numbers specifying the range of frequencies (Hz) to plot. Providing NA for either value will use the max/min frequency present in the dataset
dbRange	if not NULL, a fixed limit to use for the color scaling of dB values in the plot
units	units for plot labeling, will attempt to read them from the input
cmap	color palette map to use for plot, default is viridis_pal
toTz	timezone to use for the time axis (input data must be UTC). Specification must be from OlsonNames
alpha	alpha to use for the plot fill
maxBins	the maximum number of time bins to create for the plot. If bin would divide the range of dates in x into more than maxBins, then a warning will be given and a larger time bin will be used that reduces the number of time bins plotted. Trying to show a large number of bins will cause this function to be much slower
returnData	if TRUE then no plot will be generated, instead the dataframe that would normally be used to make the plot will be returned

Value

ggplot object of the LTSA plot

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
hmd <- checkSoundscapeInput(system.file('extdata/MANTAExampleSmall1.csv', package='PAMscapes'))
# time range is too small for nice plots
plotLTSA(hmd, bin='1min', title='Every Minute')
plotLTSA(hmd, bin='2min', title='2 Minute Bins')
```

plotPSD *Plot Power Spectral Density*

Description

Plots the distribution of summarised sound levels across frequency, either as lines of quantile levels or a heatmap showing the full distribution. Multiple PSD sources can be combined and plotted as long as they have identical frequency levels.

Usage

```
plotPSD(
  x,
  style = c("quantile", "density"),
  scale = c("log", "linear"),
  q = 0.5,
  color = "black",
  freqRange = NULL,
  dbRange = NULL,
  dbInt = 1,
  densityRange = NULL,
  units = "dB re: 1uPa^2/Hz",
  cmap = viridis_pal()(25),
  by = NULL,
  title = NULL,
  returnData = FALSE,
  progress = TRUE
)

prepPSDData(
  x,
  freqRange = NULL,
  style = c("density", "quantile"),
  by = NULL,
  dbInt = 1,
  compression = 10000,
  progress = TRUE
)
```

Arguments

x	a dataframe or list of dataframes, or file path or vector of file paths, or the output from prepPSDData
style	character specifying plot style to create, either "quantile", "density", or a vector with both
scale	scale to use for frequency axis, one of "log" or "linear"

q	quantile to plot
color	color for quantile
freqRange	range of frequencies to plot
dbRange	range of dB values to plot
dbInt	bin interval size for density plot
densityRange	optional range of values for density color scale
units	units for dB axis of plot
cmap	color map to use for density plot
by	optional column to plot different quantile lines by, only affects style='quantile'. If x is a data.frame, by can also be one of 'hour', 'month', or 'year' and that column will be created automatically if not present.
title	optional title for plot
returnData	if TRUE then no plot will be generated, instead the dataframe that would normally be used to make the plot will be returned
progress	logical flag to show progress bar
compression	compression factor for tdigest, lower values are less accurate but will compute faster. Only relevant for style='quantile' when loading and combining multiple datasets

Details

prepPSDData is called by the plotting code, and does not necessarily need to be called separately from plotPSD. Loading PSD data can be time consuming, so it may be useful to load the data first, then it is easier to spend time adjusting plot settings.

The output of prepPSDData is a list with 5 elements:

frequency - the frequency values of the input data

freqRange - the value of the "freqRange" parameter if it was supplied

dbVals - the dB values of breakpoints used for "density" plotting

quantileData - the data used for quantile plots. These are stored as "tidgest" objects serialized using as.list.tdigest, from which quantiles can be computed

densityData - the data used fro quantile plots. These are stored as a matrix of bin counts - each column corresponds to the "frequency" output, each row corresponds to bins defined using "dbVals" as boundaries

Value

a ggplot object for plotPSD, see details for prepPSDData

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
psd <- checkSoundscapeInput(system.file('extdata/PSDSmall.csv', package='PAMscapes'))
# Plotting only first 1000 columns for brevity
plotPSD(psd[1:1000], style='density')
plotPSD(psd[1:1000], style='quantile', q=.05)
```

`plotScaledTimeseries` *Plot Rescaled Timeseries*

Description

Plot timeseries of different values, rescaled so that multiple types of data are visible on the same plot

Usage

```
plotScaledTimeseries(
  x,
  columns,
  title = NULL,
  units = NULL,
  cpal = hue_pal(),
  lwd = 0.5,
  minVals = NA,
  relMax = 1,
  toTz = "UTC"
)
```

Arguments

`x`	a dataframe with column `UTC`
`columns`	the names of the columns to plot. Values of columns will be rescaled to appear similar to range of the first column
`title`	title for the plot
`units`	name of units for plot labeling, default is taken from common soundscape units
`cpal`	colors to use for different lines, can either be a color palette function or a vector of color names
`lwd`	line width, either a single value or a vector of widths matching the length of `columns`
`minVals`	minimum value for each of `columns` to use for rescaling, either a single value to use for all or a vector matching the length of `columns`. A value of `NA` will use the minimum value present in the data. See Details for more info
`relMax`	the percentage of the maximum value for all rescaled columns relative to the first column. See Details for more info
`toTz`	timezone to use for the time axis (input data must be UTC). Specification must be from OlsonNames

Details

The data in the different `columns` of `x` may have very different ranges, so they must be rescaled in order to create a useful comparison plot. The default behavior is to rescale all columns to have the same min/max range as the first column in `columns`. This means that the Y-axis values will only be accurate for the first column, and all lines will have their minimum value at the bottom edge of the plot and their maximum value at the top edge of the plot.

There are some cases where this full-range rescaling is not desirable. One case is when one of the variables should have a minimum value of zero, but the lowest value present in your data is larger than zero. For example, wind speed might in your data might range from values of 0.5 to 3, so by default this 0.5 value would appear at the bottom of the plot. However, it would make much more sense if the values were plotted relative to a minimum of zero. The `minVals` argument lets you control this. The default `NA` value uses the minimum of your data range, but you can provide a value of zero (or anything else) to control the displayed minimum.

It can also be distracting or busy to display all lines at the same relative height, especially as the number of columns displayed grows. There are two ways to help this. First, the `lwd` parameter can be used to display certain lines more prominently, making it easier to keep track of more important information. Second, the `relMax` can be used to control the maximum relative height of each line plot. The default value of 1 makes each line the same maximum height as the first column, reducing this to a value of 0.75 would make it so that all lines other than the first will not go higher than 75% of the Y-axis

Value

a ggplot object

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
manta <- checkSoundscapeInput(system.file('extdata/MANTAExampleSmall1.csv', package='PAMscapes'))
plotScaledTimeseries(manta, columns=c('HMD_50', 'HMD_100', 'HMD_200'))
```

---

plotTimeseries *Plot Timeseries*

Description

Plot simple timeseries of values

Usage

```
plotTimeseries(
  x,
  bin = "1hour",
  column,
  title = NULL,
  units = NULL,
  style = c("line", "heatmap"),
  q = 0,
  by = NULL,
  cmap = viridis_pal()(25),
  toTz = "UTC"
)
```

Arguments

x	a dataframe with column UTC
bin	time bin for summarising data. The median of values within the same time bin will be plotted
column	the name of the column to plot
title	title for the plot, if left as default NULL it will use the column name
units	name of units for plot labeling, default is taken from common soundscape units
style	one of 'line' or 'heatmap'. 'line' will create a simple line time series plot, 'heatmap' will create a grid plot with hour of day as X-axis and Date as y-axis where the value of column is the color
q	only valid for style='line', quantile level for plotting, between 0 and 1. If left as 0, none will be plotted. If a single value, then levels q and 1-q will be plotted. Users can also specify both values for non-symmettric intervals.
by	only valid for style='line', optional categorical column to plot separate lines for
cmap	only valid for style='heatmap', the color palette to use for plotting values
toTz	timezone to use for the time axis (input data must be UTC). Specification must be from OlsonNames

Value

a ggplot object

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
manta <- checkSoundscapeInput(system.file('extdata/MANTAExampleSmall2.csv', package='PAMscapes'))
plotTimeseries(manta, bin='1minute', column='HMD_150')
```

readLocalAIS *Read AIS Data Near GPS Track*

Description

Reads in AIS data downloaded from Marine Cadastre of ship tracks that come within a certain distance of a given GPS track. Also calculates the distance to the GPS track for each AIS point

Usage

```
readLocalAIS(gps, aisDir, distance = 10000, timeBuff = 0)
```

Arguments

gps	a dataframe with columns UTC, Latitude, and Longitude to get nearby AIS data for
aisDir	directory of AIS CSV files to read from
distance	distance in meters around the GPS track to read AIS data for
timeBuff	extra time (seconds) before and after the GPS points to read AIS data for. This can help create a better picture of ship activity surrounding the GPS

Value

a dataframe of AIS data, with additional columns related to distance to provided buoy GPS track

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
gps <- data.frame(Latitude=c(33.2, 33.5,33.6),
                  Longitude=c(-118.1, -118.4, -119),
                  UTC=as.POSIXct(
                    c('2022-04-28 05:00:00',
                      '2022-04-28 10:00:00',
                      '2022-04-28 20:00:00'),
                    tz='UTC'))
ais <- readLocalAIS(gps, aisDir=system.file('extdata/ais', package='PAMscapes'), distance=20e3)
str(ais)
```

runSoundscapeExplorer *Run Soundscape Explorer App*

Description

Launches a shiny app that allows users to browse the various plotting functions available to visualize soundscape data

Usage

```
runSoundscapeExplorer(data = NULL)
```

Arguments

data	file path to soundscape data or data that has been loaded with checkSoundscapeInput

Value

invisible TRUE

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
if(interactive()) {
 hmd <- checkSoundscapeInput(system.file('extdata/MANTAExampleSmall1.csv', package='PAMscapes'))
  runSoundscapeExplorer(hmd)
}
```

subsetMarCadAIS *Subset Marine Cadastre AIS Data to Region*

Description

Subsets the full download files from Marine Cadastre to a smaller region so that they are easier to work with

Usage

```
subsetMarCadAIS(
  inDir,
  outDir,
  latRange = c(20, 50),
  lonRange = c(-140, -110),
  name = "West_",
  overwrite = FALSE,
  progress = TRUE
)
```

Arguments

inDir	directory containing Marine Cadastre AIS CSV files to subset
outDir	directory to write subsetted files to
latRange	range of desired latitudes (decimal degrees)
lonRange	range of desired longitudes (decimal degrees)
name	prefix to append to new filenames
overwrite	logical flag to overwrite existing files
progress	logical flag to show progress bar

Value

invisibly return new file names

Author(s)

Taiki Sakai <taiki.sakai@noaa.gov>

Examples

```
outDir <- tempdir()
localFiles <- subsetMarCadAIS('AISData', outDir=outDir,
                              latRange=c(20, 50), lonRange=c(-140, -110),
                              name='West_')
```

Index