Package 'warbleR'

May 17, 2018

Type Package

Title Streamline Bioacoustic Analysis

Version 1.1.13

Date 2018-04-19

Author Marcelo Araya-Salas & Grace Smith Vidaurre

Maintainer Marcelo Araya-Salas <araya-salas@cornell.edu>

Description A tool to streamline the analysis of animal acoustic signal structure. The package offers functions for downloading avian vocalizations from the open-access online repository 'Xeno-Canto' <http://xeno-canto.org/>, displaying the geographic extent of the recordings, manipulating sound files, detecting acoustic signals, assessing performance of methods that measure acoustic similarity, conducting cross-correlations, dynamic time warping, measuring acoustic parameters and analysing interactive vocal signals, among others. Most functions working iteratively allow parallelization to improve computational efficiency.

License GPL (>= 2)

Imports bitops, dtw, fftw, graphics, grDevices, iterators, jpeg, monitoR, parallel, pbapply, proxy, RCurl, rjson, stats, signal, utils, methods, pracma, Sim.DiffProc

Depends R (>= 3.2.1), maps, tuneR, seewave (>= 2.0.1), NatureSounds

LazyData TRUE

URL https://github.com/maRce10/warbleR

BugReports https://github.com/maRce10/warbleR/issues

NeedsCompilation no

Suggests knitr, ggplot2, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.0.1

Repository CRAN

Date/Publication 2016-04-19 08:12:11

R topics documented:

`autodetec` *Automatically detect vocalizations in sound files*

Description

`autodetec` automatically detects the start and end of vocalizations in sound files based on amplitude, duration, and frequency range attributes.

Usage

```
autodetec(X = NULL, threshold = 15, envt = "abs", ssmooth = NULL, msmooth = NULL,
  power = 1, bp = NULL, osci = FALSE, wl = 512, xl = 1, picsize = 1, res = 100,
  flim = c(0,22), ls = FALSE, sxrow = 10, rows = 10, mindur = NULL, maxdur =
 NULL, redo = FALSE, img = TRUE, it = "jpeg", set = FALSE, flist = NULL, smadj = NULL,
  parallel = 1, path = NULL, pb = TRUE, pal = reverse.gray.colors.2,
  fast.spec = FALSE, ...)
```

Arguments

`X`	'selection_table' object or data frame with results from `manualoc` function or any data frame with columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end).
`threshold`	A numeric vector of length 1 specifying the amplitude threshold for detecting signals (in %).
`envt`	Character vector of length 1 specifying the type of envelope to be used: "abs" for absolute amplitude envelope or "hil" for Hilbert amplitude envelope. Default is "abs".
`ssmooth`	A numeric vector of length 1 to smooth the amplitude envelope with a sum smooth function. Default is NULL.
`msmooth`	A numeric vector of length 2 to smooth the amplitude envelope with a mean sliding window. The first component is the window length and the second is the overlap between successive windows (in %). Faster than ssmooth but time detection is much less accurate. Will be deprecated in future versions. Default is NULL.

power — A numeric vector of length 1 indicating a power factor applied to the amplitude envelope. Increasing power will reduce low amplitude modulations and increase high amplide modulations, in order to reduce background noise. Default is 1 (no change).

bp — Numeric vector of length 2 giving the lower and upper limits of a frequency bandpass filter (in kHz). Default is c(0, 22).

osci — Logical argument to add an oscillogram underneath spectrogram, as in spectro. Default is FALSE. Not applied if ls is TRUE.

wl — A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.

xl — Numeric vector of length 1, a constant by which to scale spectrogram width. Default is 1.

picsize — Numeric argument of length 1. Controls the relative size of the spectrogram. Default is 1.

res — Numeric argument of length 1 controling resolution of images. Default is 100 (faster) although 300 - 400 is recommended for publication/ presentation quality.

flim — A numeric vector of length 2 for the frequency limit in kHz of the spectrogram, as in spectro. Default is c(0, 22).

ls — Logical argument. If TRUE, long spectrograms as in lspec are produced.

sxrow — A numeric vector of length 1. Specifies seconds of spectrogram per row when creating long spectrograms. Default is 10. Applied when ls = TRUE and/or when X is not provided.

rows — A numeric vector of length 1. Specifies number of rows per image file when creating long spectrograms. Default is 10. Applied when ls = TRUE and/or when X is not provided.

mindur — Numeric vector of length 1 giving the shortest duration (in seconds) of the signals to be detected. It removes signals below that threshold.

maxdur — Numeric vector of length 1 giving the longest duration (in seconds) of the signals to be detected. It removes signals above that threshold.

redo — Logical argument. If TRUE all selections will be analyzed again when code is rerun. If FALSE only the selections that do not have an 'autodetec' generated image file in the working directory will be analyzed. Default is FALSE.

img — Logical argument. If FALSE, image files are not produced. Default is TRUE.

it — A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".

set — A logical argument indicating wheter the settings of the autodetection process should be included in the image file name. If TRUE, threshold (th), envelope (envt), bandpass (bp), power (pw), smooth (smo, either mmsooth[1] or ssmooth), maxdur (mxdu), and mindur (midu) are included.

flist — character vector or factor indicating the subset of files that will be analyzed. Ignored if X is provided.

smadj	adjustment for amplitude smoothing. Character vector of length one indicating whether start end values should be adjusted. "start", "end" or "both" are the inputs admitted by this argument. Amplitude smoothing through ssmooth generates a predictable deviation from the actual start and end positions of the signals, determined by the threshold and ssmooth values. This deviation is more obvious (and problematic) when the increase and decrease in amplitude at the start and end of the signal (respectively) is not gradual. Ignored if ssmooth is NULL.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.
pal	Color palette function for spectrogram. Default is reverse.gray.colors.2. See spectro for more palettes. Palettes as gray.2 may work better when fast.spec = TRUE.
fast.spec	Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast.spec' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.
...	Additional arguments to be passed to a modified version of spectro for customizing graphical output.

Details

This function determines the start and end of signals in the segments of the sound files listed in the input data frame. Alternatively, if no data frame is provided, the function detects signals across each entire sound file and creates long spectrograms highlighting the start and of the detected signals for all sound files in the working directory. The input data frame should have the following columns: c("sound.files","selec","start","end"). The ouptut of manualoc can be used as the input data frame. This function uses a modified version of the timer function from seewave package to detect signals.

Value

Image files with spectrograms showing the start and end of the detected signals. It also returns a data frame containing the start and end of each signal by sound file and selection number.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>). Implements a modified version of the timer function from seewave.

Examples

```
## Not run:
# Set temporary working directory
```

```
# setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

ad <- autodetec(threshold = 5, env = "hil", ssmooth = 300, power=1,
bp=c(2,9), xl = 2, picsize = 2, res = 200, flim= c(1,11), osci = TRUE,
wl = 300, ls = FALSE, sxrow = 2, rows = 4, mindur = 0.1, maxdur = 1, set = TRUE)

#run it with different settings
ad <- autodetec(threshold = 90, env = "abs", ssmooth = 300, power = 1, redo = TRUE,
bp=c(2,9), xl = 2, picsize = 2, res = 200, flim= c(1,11), osci = TRUE,
wl = 300, ls = FALSE,  sxrow = 2, rows = 4, mindur=0.1, maxdur=1, set = TRUE)

#check this folder!!
getwd()

## End(Not run)
```

`catalog` *Create catalog of vocal signals*

Description

`catalog` produces spectrograms of selections (signals) split into multiple rows and columns.

Usage

```
catalog(X, flim = c(0, 22), nrow = 4, ncol = 3, same.time.scale = TRUE,
collevels = seq(-40, 0, 1), ovlp = 50, parallel = 1, mar = 0.05, prop.mar = NULL,
lab.mar = 1, wl = 512, wn = "hanning", gr = FALSE, pal = reverse.gray.colors.2,
it = "jpeg", path = NULL, pb = TRUE, fast.spec = FALSE, res = 100,
orientation = "v", labels = c("sound.files", "selec"), height = NULL,
width = NULL, tags = NULL, tag.pal = list(temp.colors, heat.colors, topo.colors),
legend = 3, cex = 1, leg.wd = 1, img.suffix = NULL, img.prefix = NULL,
tag.widths = c(1, 1), hatching = 0, breaks = c(5, 5), group.tag = NULL,
spec.mar = 0, spec.bg = "white", max.group.cols = NULL, sub.legend = FALSE,
rm.axes = FALSE, title = NULL, by.row = TRUE, box = TRUE)
```

Arguments

`X`	'selection_table', 'extended_selection_table' or data frame with columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). Default is `NULL`.

flim — A numeric vector of length 2 indicating the highest and lowest frequency limits (kHz) of the spectrogram, as in spectro. Default is c(0,22).

nrow — A numeric vector of length 1. Specifies number of rows. Default is 4.

ncol — A numeric vector of length 1. Specifies number of columns. Default is 3.

same.time.scale — Logical. Controls if all spectrograms are in the same time scale (i.e. have the same duration).

collevels — A numeric vector of length 3. Specifies levels to partition the amplitude range of the spectrogram (in dB). The more levels the higher the resolution of the spectrogram. Default is seq(-40, 0, 1). seq(-115, 0, 1) will produces spectrograms similar to other acoustic analysis software packages.

ovlp — Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 50. High values of ovlp slow down the function but produce more accurate selection limits (when X is provided).

parallel — Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).

mar — Numeric vector of length 1. Specifies the margins (in seconds) adjacent to the start and end points of selections, dealineating spectrogram limits. Default is 0.05.

prop.mar — Numeric vector of length 1. Specifies the margins adjacent to the start and end points of selections as a proportion of the duration of the signal. If provided 'mar' argument is ignored. Default is NULL. Useful when having high variation in signal duration. Ignored if same.time.scale = FALSE.

lab.mar — Numeric vector of length 1. Specifies the space allocated to labels and tags (the upper margin). Default is 1.

wl — A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.

wn — Character vector of length 1 specifying the window function name. See ftwindow for name options. Default is "hanning".

gr — Logical argument to add grid to spectrogram. Default is FALSE.

pal — Color palette function for spectrogram. Default is reverse.gray.colors.2. See spectro for more palettes. Palettes as gray.2 may work better when fast.spec = TRUE.

it — A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".

path — Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.

pb — Logical argument to control progress bar. Default is TRUE.

fast.spec — Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast.spec' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.

`res` Numeric argument of length 1. Controls image resolution. Default is 100 (faster) although 300 is recommended for publication/presentation quality. Note that high resolution produce significantly bigger image files. This could be problematic when creating pdf files using `catalog`.

`orientation` String. Indicates whether a letter page size image is produced in vertical ('v' option) or horizontal orientation ('h' option). Note that width and height can also be specified.

`labels` String vector. Provides the column names that will be used as labels above the corresponding spectrograms.

`height` Numeric. Single value (in inches) indicating the height of the output image files. Default is 11 for vertical orientation.

`width` Numeric. Single value (in inches) indicating the width of the output image files. Default is 8.5 for vertical orientation.

`tags` String vector. Provides the column names that will be used for the color tagging legend above. Tags can also be numeric. Continuous variables would be break down in 10 color classes. spectrograms.

`tag.pal` List of color palette function for tags. Should be of length 1, 2. Default is `list(temp.colors, heat.colors, topo.colors)`.

`legend` A numeric vector of length 1 controling a legend for color tags is added. Ignored if no tags are provided. Four values are allowed:

- `0`: No label
- `1`: Label for the first color tag
- `2`: Label for the second color tag
- `3`: Labels both color tags

Default is 3. Currently no legend can be set for group tags. Use labels instead.

`cex` A numeric vector of length 1 giving the amount by which text (including labels and axis) should be magnified. Default is 1.

`leg.wd` Numeric. Controls the width of the legend column. Default is 1.

`img.suffix` A character vector of length 1 with a suffix (label) to add at the end of the names of image files. Default is `NULL` (no suffix). Useful to label catalogs from different individuals, species or sites.

`img.prefix` A character vector of length 1 with a prefix (label) to add at the beginning of the names of image files. Default is `NULL` (no prefix). Useful to label catalogs from different individuals, species or sites and ensure they will be grouped together when sorted by file name.

`tag.widths` A numeric vector of length 2 to control de relative width of the color tags (when 2 tags are provided).

`hatching` A numeric vector of length 1 controling cross-hatching is used for color tags. Several cross-hatching patterns are used to make tags with similar colors more distinguishable. Four values are allowed:

- `0`: No cross-hatching
- `1`: Cross-hatching the first color tag
- `2`: Cross-hatching the second color tag

- 3: Cross-hatching both color tags

`breaks` Numeric vector of length 1 or 2 controling the number of intervals in which a numeric tag will be divided. The numbers control the first and second tags respectively. Ignored if tags are not numeric. Default is `c(5, 5)`.

`group.tag` Character vector of length 1 indicating the column name to be used to color the empty plot areas around the spectrograms. If provided selections that belong to the same tag level are clumped together in the catalog (the 'X' data frame is sorted by that column). This tags cannot be included in the legend so it would be better to use the label field to identify the different levels.

`spec.mar` Numeric vector of length 1 to add space at the top, left and right sides of the spectrogram. Useful to better display the grouping of selections when 'group.tag' is provided. Internally applied for setting 'mar' using `par`.

`spec.bg` Character vector of length 1 to control the background color of the spectrogram. Default is 'white'. Ignored if `group.tag = NULL`.

`max.group.cols` Numeric vector of length 1 indicating the number of different colors that will be used for group tags (see 'group.tag' argument). If provided (and the number is smaller than the number of levels in the 'group.tag' column) the colors will be recycled, athough ensuring that adjacent groups do not share the same color. Useful when the 'group.tag' has many levels and the colors assigned become very similar. Default is `NULL`.

`sub.legend` Logical. If `TRUE` then only the levels present on each page are shown in the legend. Default is `FALSE`.

`rm.axes` Logical. If `TRUE` frequency and time axes are excluded. Default is `FALSE`.

`title` Character vector of length 1 to set the tile of catalogs.

`by.row` Logical. If `TRUE` (default) catalogs are filled by rows.

`box` Logical. If `TRUE` (default) a box is drawn around spectrograms and corresponding labels and tags. are

Details

This functions aims to simplify the visual exploration of multiple vocalizations. The function plots a matrix of spectrograms from a selection table. Spectrograms can be labeled or color tagged to facilitate exploring variation related to a parameter of interest (e.g. location, song type). A legend will be added to help match colors with tag levels (if legend is > 0). Different color palettes can be used for each tag. Numeric tags are split in intervals (the number of intervals can be controlled with break argument). The width and height can also be adjusted to fit more column and/or rows. This files can be put together in a single pdf file with `catalog2pdf`. We recommend using low resolution (~60-100) and smaller dimensions (width & height < 10) if aiming to generate pdfs (otherwise pdfs could be pretty big).

Value

Image files with spectrograms of whole sound files in the working directory. Multiple pages can be returned, depending on the length of each sound file.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

https://marce10.github.io/2017/03/17/Creating_song_catalogs.html https://marce10.github.io/2017/07/31/Updates_on_catalog_function.html catalog2pdf

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())
# save sound file examples
data(list = c("Phae.long1", "Phae.long2","selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
 writeWave(Phae.long3,"Phae.long3.wav")
 writeWave(Phae.long4,"Phae.long4.wav")


catalog(X = selec.table, flim = c(1, 10), nrow = 4, ncol = 2, same.time.scale = T,
 ovlp = 90, parallel = 1, mar = 0.01, wl = 200, gr = FALSE,
 orientation = "v", labels = c("sound.files", "selec"), legend = 0)

 #different time scales and tag palette
catalog(X = selec.table, flim = c(1, 10), nrow = 4, ncol = 2, same.time.scale = F,
 ovlp = 90, parallel = 1, mar = 0.01, wl = 200,
 orientation = "v",  labels = c("sound.files", "selec"), legend = 0,
 tag.pal = list(terrain.colors))

 #adding tags and changing spectro palette
catalog(X = selec.table, flim = c(1, 10), nrow = 4, ncol = 2, same.time.scale = F,
 ovlp = 90, parallel = 1, mar = 0.01, wl = 200, pal = reverse.heat.colors,
 orientation = "v",  labels = c("sound.files", "selec"), legend = 1,
 tag.pal = list(terrain.colors), tags = "sound.files")

 #create a bigger selection table
 X <- rbind(selec.table, selec.table, selec.table, selec.table)
 X <- rbind(X, X)

 #create some simulated labels
 X$songtype <- sample(letters[13:15], nrow(X), replace = T)
 X$indiv <- sample(letters[1:12], nrow(X), replace = T)

# 12 columns in 5 rows, 2 tags
catalog(X = X, flim = c(1, 10), nrow = 5, ncol = 12, same.time.scale = F,
 ovlp = 90, parallel = 1, mar = 0.01, wl = 200,
 orientation = "v",  labels = c("sound.files", "selec"), legend = 3,
 collevels = seq(-65, 0, 5), tag.pal = list(terrain.colors), tags = c("songtype", "indiv"))
```

```
# with legend
catalog(X = X, flim = c(1, 10), nrow = 5, ncol = 12, same.time.scale = F,
 ovlp = 90, parallel = 1, mar = 0.01, wl = 200, gr = FALSE,
 orientation = "v",  labels = c("sound.files", "selec"), legend = 3,
 width = 20, collevels = seq(-65, 0, 5), tag.pal = list(terrain.colors),
  tags = c("songtype", "indiv"))

  # horizontal orientation
catalog(X = X, flim = c(1, 10), nrow = 5, ncol = 12, same.time.scale = F,
 ovlp = 90, parallel = 1, mar = 0.01, wl = 200, gr = FALSE,
 orientation = "h",  labels = c("sound.files", "selec"), legend = 3,
 width = 20, collevels = seq(-65, 0, 5), tag.pal = list(terrain.colors),
  tags = c("songtype", "indiv"))
check this floder
getwd()

## End(Not run)
```

---

catalog2pdf catalog2pdf *combines* catalog *images into pdfs*

Description

catalog2pdf combines catalog images into pdfs

Usage

```
catalog2pdf(keep.img = TRUE, overwrite = FALSE, parallel = 1, path = NULL,
pb = TRUE, by.img.suffix = FALSE, ...)
```

Arguments

keep.img	Logical argument. Indicates whether jpeg files should be kept (default) or remove. (including sound file and page number) should be magnified. Default is 1.
overwrite	Logical argument. If TRUE all jpeg pdf will be produced again when code is rerun. If FALSE only the ones missing will be produced. Default is FALSE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the catalog image files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.
by.img.suffix	Logical. If TRUE catalogs with the same image suffix will be put together in a single pdf (so one pdf per image suffix in the catalog images). Default is FALSE (i.e. no suffix).
...	Additional arguments to be passed to the internal pdf creating function pdf for customizing output.

Details

The function combines catalog images in .jpeg format from the catalog function into pdfs. Note that using lower resolution and smaller dimension (width and height) when creating catalogs will substantially decrease the size of pdf files (which could be pretty big).

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

catalog2pdf, https://marce10.github.io/2017/03/17/Creating_song_catalogs.html

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())

# save sound file examples
data(list = c("Phae.long1", "Phae.long2"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")

catalog(X = selec.table, nrow = 2, ncol = 4)

# now create single pdf removing jpeg
catalog2pdf(keep.img = FALSE)

# check this floder
getwd()

## End(Not run)
```

---

checksels *Check selection data frames*

Description

checksels checks whether selections can be read by subsequent functions.

Usage

```
checksels(X, parallel = 1, path = NULL, check.header = FALSE, pb = TRUE,
wav.size = FALSE)
```

Arguments

X	'selection_table' object or data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "sel": number of the selections, 3) "start": start time of selections, 4) "end": end time of selections. Alternatively, a 'selection_table' class object can be input to double check selections. The ouptut of manualoc or autodetec can be used as the input data frame.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
check.header	Logical. Controls whether sound file headers correspond to the actual file properties (i.e. if is corrupted). This could significantly affect the performance of the function (much slower) particularly with long sound files.
pb	Logical argument to control progress bar. Default is TRUE.
wav.size	Logical argument to control if the size of the wave object when the selection is imported into R (as when using readWave is calculated and added as a column. Size is return in MB. Default is FALSE.

Details

This function checks 1) if the selections listed in the data frame correspond to .wav files in the working directory, 2) if the sound files can be read and if so, 3) if the start and end time of the selections are found within the duration of the sound files. Note that the sound files should be in the working directory (or the directory provided in 'path'). This is useful for avoiding errors in dowstream functions (e.g. specan, xcorr, catalog, dfDTW). Note that corrupt files can be fixed using fixwavs) ('sox' must be installed to be able to run this function).

Value

A data frame including the columns in the input data frame (X) and 2 additional columns: "check.res" (check selections), and "min.n.samples" (the smallest number of samples). Note the number of samples available in a selection limits the minimum window length (wl argument in other functions) that can be used in batch analyses.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

checkwavs

Examples

```
{
# First set temporary folder
# setwd(tempdir())
```

```
# save wav file examples
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")

checksels(X = selec.table)
}
```

---

checkwavs *Check .wav files*

Description

checkwavs checks whether .wav files can be read by subsequent functions.

Usage

```
checkwavs(X = NULL, path = NULL)
```

Arguments

X	Optional. 'selection_table' object or data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "sel": number of the selections, 3) "start": start time of selections, 4) "end": end time of selections. The ouptut of manualoc or autodetec can be used as the input data frame. If provided the function also returns the smallest number of samples from the listed selections, which limits the minimum window length (wl argument in other functions) that can be used in batch analyses. This could be useful for avoiding errors in dowstream functions (e.g. specan).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.

Details

This function checks if .wav files in the working directory can be read. Users must set the working directory where they wish to check .wav files beforehand. If X is provided it also returns the smallest number of samples from the selections listed in X (if all files can be read). Note that corrupt files can be fixed using fixwavs) ('sox' must be installed to be able to run this function). The function is intended for a "quick and dirty" check of the .wav files in a selections data frame. For a more thourough analysis see checksels.

Value

If all .wav files are ok, returns message "All files can be read". Otherwise returns the names of the corrupted .wav files.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

checksels seltailor

Examples

```
{
# First set temporary folder
# setwd(tempdir())

# save wav file examples
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

# without selection data frame
checkwavs()

# without selection data frame
checkwavs(X = selec.table)
}
```

---

color.spectro *Highlight spectrogram regions*

Description

color.spectro highlights spectrogram regions specified by users

Usage

```
color.spectro(wave, wl = 512, wn = "hanning", ovlp = 70,
dB = "max0", collevels = NULL, selec.col = "red2", col.clm = NULL,
base.col = "black", bg.col = "white", cexlab = 1, cexaxis = 1, tlab = "Time (s)",
flab = "Frequency (kHz)", title = NULL, axisX = TRUE, axisY = TRUE,
flim = NULL, rm.zero = FALSE, X = NULL, fast.spec = FALSE, t.mar = NULL, f.mar = NULL,
interactive = NULL, add = FALSE)
```

Arguments

wave	A 'wave' object produced by readWave or similar functions.
wl	A numeric vector of length 1 specifying the window length of the spectrogram. Default is 512.

wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
ovlp	Numeric vector of length 1 specifying the percent overlap between two consecutive windows, as in spectro. Default is 70.
dB	Character vector of length 1 controling the amplitude weights as in spectro. Default is 'max0'.
collevels	Numeric. Levels used to partition amplitude range as in spectro. Default is NULL.
selec.col	Character vector of length 1 specifying the color to be used to highlight selection. See 'col.clm' for specifying unique colors for each selection. Default is 'red2'. Ignored if 'col.cm' and 'X' are provided.
col.clm	Character vector of length 1 indicating the name of the column in 'X' that contains the color names for each selection. Ignored if X == NULL or interactive != NULL. Default is NULL.
base.col	Character vector of length 1 specifying the color of the bacground spectrogram. Default is 'black'.
bg.col	Character vector of length 1 specifying the background color for both base and hlighlighted spectrograms. Default is 'white'.
cexlab	Numeric vector of length 1 specifying the relative size of axis labels. See spectro. Default is 1.
cexaxis	Numeric vector of length 1 specifying the relative size of axis. See spectro. Default is 1.
tlab	Character vector of length 1 specifying the label of the time axis.
flab	Character vector of length 1 specifying the label of the frequency axis.
title	Logical argument to add a title to individual spectrograms. Default is TRUE.
axisX	Logical to control whether time axis is plotted. Default is TRUE.
axisY	Logical to control whether frequency axis is plotted. Default is TRUE.
flim	A numeric vector of length 2 for the frequency limit (in kHz) of the spectrogram, as in spectro. Default is NULL.
rm.zero	Logical indicated if the 0 at the start of the time axis should be removed. Default is FALSE.
X	Optional. Data frame containing columns for start and end time of signals ('start' and 'end') and low and high frequency ('bottom.freq' and 'top.freq').
fast.spec	Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.
t.mar	Numeric vector of length 1. Specifies the margins adjacent to the start and end points to be added when highlighting selection. Default is NULL.

f.mar	Numeric vector of length 1. Specifies the margins adjacent to the low and high frequencies to be added when highlighting selection. Default is NULL.
interactive	Numeric. Allow user to interactively select the signals to be highlighted by cliking on the graphic device. Users must select the opposite corners of a square delimiting the spectrogram region to be highlighted. Controls the number of signals that users would be able to select (2 clicks per signal).
add	Logical. If TRUE new highlighting can be applied to the current plot (which means that the function with add = FALSE should be run first). Default is FALSE.

Details

This function highlights regions of the spectrogram with diferent colors. The regions to be highlighted can be provided in a selection table (as the example data 'selec.table') or interactively ('interactive' argument).

Value

A plot is produced in the graphic device.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) and Grace Smith Vidaurre

See Also

trackfreqs for creating spectrograms to visualize frequency measurements by specan, snrspecs for creating spectrograms to optimize noise margins used in sig2noise

Other spectrogram creators: dfDTW, dfts, ffDTW, ffts, snrspecs, sp.en.ts, specreator, trackfreqs

Examples

```
## Not run:
# First set empty folder
# setwd(tempdir())
data(list = c("Phae.long1", "selec.table"))
writeWave(Phae.long1, "Phae.long1.wav") #save sound files

 # subset selection table
 st <- selec.table[selec.table$sound.files == "Phae.long1.wav",]

 # read wave file as an R object
 sgnl <- tuneR::readWave(as.character(st$sound.files[1]))

 # create color column
 st$colors <- c("red2", "blue", "green")

 # highlight selections
color.spectro(wave = sgnl, wl = 300, ovlp = 90, flim = c(1, 8.6), collevels = seq(-90, 0, 5),
dB = "B", X = st, col.clm = "colors", base.col = "skyblue",  t.mar = 0.07, f.mar = 0.1,
```

```
 interactive = NULL)

 # interactive (selected manually: you have to select them by clicking on the spectrogram)
 color.spectro(wave = sgnl, wl = 300, ovlp = 90, flim = c(1, 8.6), collevels = seq(-90, 0, 5),
  dB = "B", col.clm = "colors", t.mar = 0.07, f.mar = 1, interactive = 2)

## End(Not run)
```

---

compare.methods *Assessing the performance of acoustic distance measurements*

Description

compare.methods creates graphs to visually assess performance of acoustic distance measurements

Usage

```
compare.methods(X = NULL, flim = c(0, 22), bp = c(0, 22), mar = 0.1, wl = 512, ovlp = 90,
res = 150, n = 10, length.out = 30, methods = c("XCORR", "dfDTW", "ffDTW", "SP"),
it = "jpeg", parallel = 1, path = NULL, sp = NULL, pb = TRUE, grid = TRUE,
clip.edges = TRUE, threshold = 15, na.rm = FALSE, scale = FALSE,
 pal = reverse.gray.colors.2, img = TRUE, ...)
```

Arguments

X	'selection_table' object or data frame with results from manualoc function, autodetec function, or any data frame with columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). Default NULL.
flim	A numeric vector of length 2 for the frequency limit in kHz of the spectrogram, as in spectro. Default is c(0, 22).
bp	numeric vector of length 2 giving the lower and upper limits of the frequency bandpass filter (in kHz) used in the acoustic distance methods. Default is c(0, 22). Note that for XCORR this argument sets the frange argument from the xcorr function.
mar	Numeric vector of length 1. Specifies plot margins around selection in seconds. Default is 0.1.
wl	A numeric vector of length 1 specifying the window length of the spectrogram and cross-correlation, default is 512.
ovlp	Numeric vector of length 1 specifying the percent overlap between two consecutive windows, as in spectro. Default is 90.
res	Numeric argument of length 1. Controls image resolution. Default is 150.
n	Numeric argument of length 1. Defines the number of plots to be produce. Default is 10.

`length.out` A character vector of length 1 giving the number of measurements of fundamental or dominant frequency desired (the length of the time series). Default is 30.

`methods` A character vector of length 2 giving the names of the acoustic distance methods that would be compared. The methods available are: cross-correlation (XCORR, from `xcorr`), dynamic time warping on dominant frequency time series (dfDTW, from `dtw` applied on `dfts` output), dynamic time warping on dominant frequency time series (ffDTW, from `dtw` applied on `ffts` output), spectral parameters (SP, from `specan`).

`it` A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".

`parallel` Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).

`path` Character string containing the directory path where the sound files are located. If `NULL` (default) then the current working directory is used.

`sp` Data frame with acoustic parameters as the one generated by `specan`. Must contain 'sound.files' and "selec' columns and the same selections as in 'X'.

`pb` Logical argument to control progress bar. Default is `TRUE`.

`grid` Logical argument to control the presence of a grid on the spectrograms (default is `TRUE`).

`clip.edges` Logical argument to control whether edges (start or end of signal) in which amplitude values above the threshold were not detected will be removed when using dfDTW and ffDTW methods. If `TRUE` this edges will be excluded and signal contour will be calculated on the remainging values. Default is `TRUE`.

`threshold` amplitude threshold (%) for dominant and/or fundamental frequency detection when using dfDTW, ffDTW and SP methods. Default is 15.

`na.rm` Logical. If `TRUE` all NAs produced when pairwise cross-correlations failed are removed from the results. This means that all selections with at least 1 cross-correlation that failed are excluded in both methods under comparison. Only apply if XCORR is one of the methods being compared.

`scale` Logical. If `TRUE` dominant and/or fundamental frequency values are z-transformed using the `scale` function, which "ignores" differences in absolute frequencies between the signals in order to focus the comparison in the frequency contour, regardless of the pitch of signals. Default is `TRUE`.

`pal` A color palette function to be used to assign colors in the spectrograms, as in `spectro`. Default is reverse.gray.colors.2.

`img` A logical argument specifying whether an image files would be produced. Default is `TRUE`.

`...` Additional arguments to be passed to a modified version of `spectro` for customizing graphical output. This includes fast.spec, an argument that speeds up the plotting of spectrograms (see description in `specreator`).

Details

This function produces graphs with spectrograms from 4 signals in the provided data frame that allow visual inspection of the performance of acoustic distance methods at comparing those signals. The signals are randomly picked up from the provided data frame (X argument).The spectrograms are all plotted with the same frequency and time scales. The function compares 2 methods at a time. The methods available are: cross-correlation (XCORR, from `xcorr`), dynamic time warping on dominant frequency time series (dfDTW, from `dtw` applied on `dfts` output), dynamic time warping on dominant frequency time series (ffDTW, from `dtw` applied on `ffts` output), spectral parameters (SP, from `specan`). The graph also contains 2 scatterplots (1 for each method) of the acoustic space of all signals in the input data frame 'X'. The compared selections are randomly picked up from the pool of selections in the input data frame. The argument 'n' defines the number of comparisons (i.e. graphs) to be produced. The acoustic pairwise distance between signals is shown next to the arrows linking them. The font color of a distance value correspond to the font color of the method that generated it, as shown in the scatterplots. Distances are standardized, being 0 the distance of a signal to itself and 1 the farthest pairwise distance in the pool of signals. Principal Component Analysis (`princomp`) is applied to calculate distances when using spectral parameters (SP). In that case the first 2 PC's are used. Classical Multidimensional Scalling (also known as Principal Coordinates Analysis, (`cmdscale`)) is used for all other methods. Note that SP can only be used with at least 22 selections (number of rows in input data frame) as PCA only works with more units than variables. The graphs are return as image files in the working directory. The file name contains the methods being compared and the rownumber of the selections. This function uses internally a modified version of the `spectro` function from seewave package to create spectrograms.

Value

Image files with 4 spectrograms of the selection being compared and scatterplots of the acoustic space of all signals in the input data frame 'X'.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>). It uses internally a modified version of the `spectro` function from seewave package to create spectrograms.

See Also

https://marce10.github.io/2017/02/17/Choosing_the_right_method_for_measuring_acoustic_signal_structure.html

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")
```

```
compare.methods(X = selec.table, flim = c(0, 10), bp = c(0, 10), mar = 0.1, wl = 300,
ovlp = 90, res = 200, n = 10, length.out = 30,
methods = c("XCORR", "dfDTW"), parallel = 1, it = "jpeg")

#remove progress bar
compare.methods(X = selec.table, flim = c(0, 10), bp = c(0, 10), mar = 0.1, wl = 300,
ovlp = 90, res = 200, n = 10, length.out = 30,
methods = c("XCORR", "dfDTW"), parallel = 1, it = "jpeg", pb = FALSE)

#check this folder!
getwd()


#compare SP and XCORR
#first we need to create a larger data set as the PCA that summarizes the spectral parameters
#needs more units (rows) that variables (columns)
#so I just create a new selection table repeating 3 times selec.table
st2 <- rbind(selec.table, selec.table, selec.table)

#note that the selection labels should be also changed
st2$selec <- 1:nrow(st2)
#now we can compare SP method against XCORR
compare.methods(X = st2, flim = c(0, 10), bp = c(0, 10), mar = 0.1, wl = 300,
ovlp = 90, res = 200, n = 10, length.out = 30,
methods = c("XCORR", "SP"), parallel = 1, it = "jpeg")

#compare SP method against dfDTW
compare.methods(X = st2, flim = c(0, 10), bp = c(0, 10), mar = 0.1, wl = 300,
ovlp = 90, res = 200, n = 10, length.out = 30,
methods = c("dfDTW", "SP"), parallel = 1, it = "jpeg")

#alternatively we can provide our own SP matrix
sp <- specan(selec.table, bp = c(0, 10))

#and selec just a few variables to avoid the problem of # observations vs # parameters in PCA
sp <- sp[, 1:7]

compare.methods(X = selec.table, flim = c(0, 10), sp = sp, bp = c(0, 10), mar = 0.1, wl = 300,
ovlp = 90, res = 200, n = 10, length.out = 30,
methods = c("XCORR", "SP"), parallel = 1, it = "jpeg")

#note that "SP" should also be included as a method in 'methods'
#again, all images are saved in the working directory

## End(Not run)
```

---

consolidate *Consolidate (sound) files into a single folder*

Description

consolidate copies (sound) files scattered in several directories into a single folder.

Usage

```
consolidate(files = NULL, path = NULL, dest.path = NULL, pb = TRUE, file.ext = ".wav$",
parallel = 1, save.csv = TRUE, ...)
```

Arguments

files	character vector or factor indicating the subset of files that will be analyzed. The files names should include the full file path. Optional.
path	Character string containing the directory path where the sound files are located. 'wav.path' set by warbleR_options is ignored. If NULL (default) then the current working directory is used.
dest.path	Character string containing the directory path where the cut sound files will be saved. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.
file.ext	Character string defining the file extension for the files to be consolidated. Default is '.wav$' ignoring case.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
save.csv	Logical. Controls whether a data frame containing sound file information is saved in the new folder.
...	Additional arguments to be passed to the internal file.copy function for customizing file copyin.

Details

This function allow users to put files scattered in several folders in a single folder. By default it works on sound files in '.wav' format but can work with other type of files (for instance '.txt' selection files).

Value

All (sound) files are consolidated (copied) to a single folder ("consolidated_files"). If csv = TRUE (default) a '.csv' file with the information about file location, old and new names (if any renaming happen) is also saved in the same folder. This data frame is also return as an object in the R environment.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

fixwavs for making sound files readable in R

Other selection manipulation, sound file manipulation: cut_sels

Examples

```
{
# First set empty folder
# setwd(tempdir())

# save wav file examples
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))

# create first folder
dir.create("folder1")
writeWave(Phae.long1, file.path("folder1","Phae.long1.wav"))
writeWave(Phae.long2, file.path("folder1","Phae.long2.wav"))

# create second folder
dir.create("folder2")
writeWave(Phae.long3, file.path("folder2","Phae.long3.wav"))
writeWave(Phae.long4, file.path("folder2","Phae.long4.wav"))

# consolidate in a single folder
consolidate()

# or if tempdir wa used
# consolidate(path = tempdir())
}
```

---

coor.graph *Coordinated singing graphs*

Description

coor.graph creates graphs of coordinated singing and highlights the signals that overlap in time. The signals are represented by polygons of different colors.

Usage

```
coor.graph(X, only.coor = FALSE, ovlp = TRUE, xl = 1, res= 80, it = "jpeg", img = TRUE,
tlim = NULL, pb = TRUE)
```

Arguments

X	Data frame containing columns for singing event (sing.event), individual (indiv), and start and end time of signal (start and end).
only.coor	Logical. If TRUE only the segment in which both individuals are singing is included (solo singing is removed). Default is FALSE.
ovlp	Logical. If TRUE the vocalizations that overlap in time are highlighted. Default is TRUE.

xl	Numeric vector of length 1, a constant by which to scale spectrogram width. Default is 1.
res	Numeric argument of length 1. Controls image resolution. Default is 80.
it	A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".
img	Logical argument. If FALSE, image files are not produced and the graphs are shown in the current graphic device. Default is TRUE.
tlim	Numeric vector of length 2 indicating the start and end time of the coordinated singing events to be displayed in the graphs.
pb	Logical argument to control progress bar and messages. Default is TRUE.

Details

This function provides visualization for coordination of acoustic signals. Signals are shown as polygon across a time axis. It also shows which signals overlap, the amount of overlap, and highlights the individual responsible for the overlap using a color code. The width of the polygons depicting the time of overlap.

Value

The function returns a list of graphs, one for each singing event in the input data frame. The graphs can be plotted by simply calling the list. If 'img' is TRUE then the graphs are also saved in the working directory as files.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

Examples

```
{

# First set temporary folder
# setwd(tempdir())

# load simulate singing events  (see data documentation)
data(sim.coor.sing)

# make coor.graphs in tiff format
coor.graph(X = sim.coor.sing, ovlp = TRUE, only.coor = FALSE, xl =2, res =80,
it = "tiff", img = TRUE)


#' # make coor.graphs in graphic device format
cgs <- coor.graph(X = sim.coor.sing, ovlp = TRUE, only.coor = FALSE, img = FALSE)

cgs
}
```

coor.test *Randomization test for singing coordination*

Description

Monte Carlo randomization test to assess the statistical significance of singing coordination

Usage

```
coor.test(X, iterations = 1000, less.than.chance = TRUE, parallel = 1, pb = TRUE,
rm.imcomp = FALSE, cutoff = 2, rm.solo = FALSE)
```

Arguments

X	Data frame containing columns for singing event (sing.event), individual (indiv), and start and end time of signal (start and end).
iterations	number of iterations for shuffling and calculation of the expected number of overlaps. Default is 1000.
less.than.chance	Logical. If TRUE the test evaluates whether overlaps occur less often than expected by chance. If FALSE the opposite pattern is evaluted (whether overlaps occur more often than expected by chance). Default is TRUE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
pb	Logical argument to control progress bar. Default is TRUE.
rm.imcomp	Logical. If TRUE removes the events that don't have 2 interacting individuals. Default is FALSE.
cutoff	Numeric. Determines the minimum number of signals per individual in a singing event. Events not meeting this criterium are removed if rm.imcomp is TRUE. If rm.icomp is FALSE cutoff is ignored. Default is 2. Note that randomization tests are not reliable with very small sample sizes. Ideally 10 or more signals per individual should be available in each singing event.
rm.solo	Logical. Controls if signals that are not intercalated at the start or end of the sequence are removed (if TRUE). For instances the sequence of signals A-A-A-B-A-B-A-B-B-B (in which A and B represent different individuals, as in the 'indiv' column) would be subset to A-B-A-B-A-B. Default is FALSE.

Details

This function calculates the probability of finding and equal or lower number (or higher if les.than.chance is TRUE) of song overlaps in a coordinated singing event. The function shuffles the sequences of signals and silence-between-signals for both individuals to produce a null distribution of expected number of overlaps by chance. The observed number of overlaps is compared to this expected values. The p-values are calculated as the proportion of random expected values that were lower (or higher) than the observed value. The function runs one test for each singing event in the input data frame. The function is equivalent to the "KeepGaps" methods described in Masco et al. 2015.

Value

A data frame with the following columns: #'

- `sing.event`: singing event ID
- `obs.overlaps`: observed number of overlaps
- `mean.random.ovlps`: mean number of overlaps expected by chance
- `p.value`: p value
- `coor.score`: coordination score (**sensu** Araya-Salas et al. 2017), calculated as '(obs.overlaps - mean.random.ovlps) / mean.random.ovlps'. Positive values indicate a tendency to overlap while negative values indicate a tendency to alternate.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

References

Araya-Salas M., Wojczulanis-Jakubas K., Phillips E.M., Mennill D.J., Wright T.F.\ (2017) To overlap or not to overlap: context-dependent coordinated singing in lekking long-billed hermits. Anim Behav. Masco, C., Allesina, S., Mennill, D. J., and Pruett-Jones, S. (2015). The Song Overlap Null model Generator (SONG): a new tool for distinguishing between random and non-random song overlap. Bioacoustics.

Examples

```
{
#load  simulated singing data (see data documentation)
data(sim_coor_sing)

# testing if coordination happens less than expected by chance
coor.test(sim_coor_sing, iterations = 100, less.than.chance = TRUE)

# testing if coordination happens more than expected by chance
coor.test(sim_coor_sing, iterations = 100, less.than.chance = FALSE)
}
```

---

cut_sels *Cut selections into individual sound files*

Description

`cut_sels` cuts selections from a selection table into individual sound files.

Usage

```
cut_sels(X, mar = 0.05, parallel = 1, path = NULL, dest.path = NULL, pb = TRUE,
labels = c("sound.files", "selec"), overwrite = FALSE, ...)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame containing columns for sound file name (sound.files), selection number (selec), and start and end time of signals (start and end). The ouptut of manualoc or autodetec can be used as the input data frame.
mar	Numeric vector of length 1. Specifies the margins adjacent to the start and end points of selections, dealineating spectrogram limits. Default is 0.05.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
dest.path	Character string containing the directory path where the cut sound files will be saved. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.
labels	String vector. Provides the column names that will be used as labels to create sound file names. Note that they should provide unique names (otherwise sound files will be overwritten). Default is c("sound.files", "selec").
overwrite	Logical. If TRUE sound files with the same name will be overwritten. Default is FALSE.
...	Additional arguments to be passed to the internal writeWave function for customizing sound file output (e.g. normalization).

Details

This function allow users to produce individual sound files from the selections listed in a selection table as in selec.table.

Value

Sound files of the signals listed in the input data frame.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) and Grace Smith Vidaurre

See Also

seltailor for tailoring selections https://marce10.github.io/2017/06/06/Individual_sound_files_for_each_selection.html

Other selection manipulation, sound file manipulation: consolidate

Examples

```
{
# First set empty folder
# setwd(tempdir())
```

```
# save wav file examples
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

# make spectrograms

cut_sels(selec.table)

cut_sels(selec.table, overwrite = TRUE, labels = c("sound.files", "selec", "sel.comment"))

 #check this folder!!
getwd()
}
```

---

dfDTW *Acoustic dissimilarity using dynamic time warping on dominant frequency contours*

Description

dfDTW calculates acoustic dissimilarity of dominant frequency contours using dynamic time warping. Internally it applies the dtwDist function from the dtw package.

Usage

```
dfDTW(X = NULL, wl = 512, wl.freq = 512, length.out = 20, wn = "hanning", ovlp = 70,
bp = c(0, 22), threshold = 15, threshold.time = NULL, threshold.freq = NULL, img = TRUE,
parallel = 1, path = NULL, ts.df = NULL, img.suffix = "dfDTW", pb = TRUE,
clip.edges = TRUE, window.type = "none", open.end = FALSE, scale = FALSE,
frange.detec = FALSE,  fsmooth = 0.1, ...)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame containing columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). The ouptut of manualoc or autodetec can be used as the input data frame.
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
wl.freq	A numeric vector of length 1 specifying the window length of the spectrogram for measurements on the frecuency spectrum. Default is 512. Higher values would provide more accurate measurements.
length.out	A numeric vector of length 1 giving the number of measurements of dominant frequency desired (the length of the time series).

wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 70.
bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz). Default is c(0, 22).
threshold	amplitude threshold (%) for dominant frequency detection. Default is 15.
threshold.time	amplitude threshold (%) for the time domain. Use for fundamental and dominant frequency detection. If NULL (default) then the 'threshold' value is used.
threshold.freq	amplitude threshold (%) for the frequency domain. Use for frequency range detection from the spectrum (see 'frange.detec'). If NULL (default) then the 'threshold' value is used.
img	Logical argument. If FALSE, image files are not produced. Default is TRUE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
ts.df	Optional. Data frame with frequency contour time series of signals to be compared. If provided "X" is ignored.
img.suffix	A character vector of length 1 with a suffix (label) to add at the end of the names of image files. Default is NULL.
pb	Logical argument to control progress bar. Default is TRUE.
clip.edges	Logical argument to control whether edges (start or end of signal) in which amplitude values above the threshold were not detected will be removed. If TRUE (default) this edges will be excluded and signal contour will be calculated on the remainging values. Note that DTW cannot be applied if missing values (e.i. when amplitude is not detected).
window.type	dtw windowing control parameter. Character: "none", "itakura", or a function (see dtw).
open.end	dtw control parameter. Performs open-ended alignments (see dtw).
scale	Logical. If TRUE dominant frequency values are z-transformed using the scale function, which "ignores" differences in absolute frequencies between the signals in order to focus the comparison in the frequency contour, regardless of the pitch of signals. Default is TRUE.
frange.detec	Logical. Controls whether frequency range of signal is automatically detected using the frange.detec function. If so, the range is used as the bandpass filter (overwriting 'bp' argument). Default is FALSE.
fsmooth	A numeric vector of length 1 to smooth the frequency spectrum with a mean sliding window (in kHz) used for frequency range detection (when frange.detec = TRUE). This help to average amplitude "hills" to minimize the effect of amplitude modulation. Default is 0.1.
...	Additional arguments to be passed to trackfreqs for customizing graphical output.

Details

This function extracts the dominant frequency values as a time series and then calculates the pairwise acoustic dissimilarity using dynamic time warping. The function uses the approx function to interpolate values between dominant frequency measures. If 'img' is TRUE the function also produces image files with the spectrograms of the signals listed in the input data frame showing the location of the dominant frequencies.

Value

A matrix with the pairwise dissimilarity values. If img is FALSE it also produces image files with the spectrograms of the signals listed in the input data frame showing the location of the dominant frequencies.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

specreator for creating spectrograms from selections, snrspecs for creating spectrograms to optimize noise margins used in sig2noise and dfts, ffts, ffDTW for frequency contour overlaid spectrograms. https://marce10.github.io/2016/09/12/Similarity_of_acoustic_signals_with_dynamic_time_warping_(DTW).html

Other spectrogram creators: color.spectro, dfts, ffDTW, ffts, snrspecs, sp.en.ts, specreator, trackfreqs

Examples

```
{
# set the temp directory
# setwd(tempdir())

#load data
data(list = c("Phae.long1", "Phae.long2","selec.table"))
writeWave(Phae.long2, "Phae.long2.wav") #save sound files
writeWave(Phae.long1, "Phae.long1.wav")

# run function
dfDTW(selec.table, length.out = 30, flim = c(1, 12), bp = c(2, 9), wl = 300)

}
```

---

dfts *Extract the dominant frequency values as a time series*

Description

dfts extracts the dominant frequency values as a time series. of signals selected by manualoc or autodetec.

Usage

```
dfts(X, wl = 512, wl.freq = 512, length.out = 20, wn = "hanning", ovlp = 70,
bp = c(0, 22), threshold = 15, threshold.time = NULL, threshold.freq = NULL,
img = TRUE, parallel = 1, path = NULL, img.suffix = "dfts", pb = TRUE,
clip.edges = FALSE, leglab = "dfts", frange.detec = FALSE, fsmooth = 0.1,
 raw.contour = FALSE, track.harm = FALSE, adjust.wl = FALSE, ...)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame containing columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). The ouptut of manualoc or autodetec can be used as the input data frame.
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
wl.freq	A numeric vector of length 1 specifying the window length of the spectrogram for measurements on the frecuency spectrum. Default is 512. Higher values would provide more accurate measurements.
length.out	A numeric vector of length 1 giving the number of measurements of dominant frequency desired (the length of the time series).
wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 70.
bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz). Default is c(0, 22).
threshold	amplitude threshold (%) for dominant frequency detection. Default is 15. Note that amplitude threshold for time and frequency domains can be defined independently. See "threshold.time" and "threshold.freq" arguments.
threshold.time	amplitude threshold (%) for the time domain. Use for dominant frequency detection. If NULL (default) then the 'threshold' value is used.
threshold.freq	amplitude threshold (%) for the frequency domain. Use for frequency range detection from the spectrum (see 'frange.detec'). If NULL (default) then the 'threshold' value is used.
img	Logical argument. If FALSE, image files are not produced. Default is TRUE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located.
img.suffix	A character vector of length 1 with a sufix (label) to add at the end of the names of image files.
pb	Logical argument to control progress bar. Default is TRUE.
clip.edges	Logical argument to control whether edges (start or end of signal) in which amplitude values above the threshold were not detected will be removed. If TRUE this edges will be excluded and signal contour will be calculated on the remainging values. Default is FALSE.

leglab	A character vector of length 1 or 2 containing the label(s) of the frequency contour legend in the output image.
frange.detec	Logical. Controls whether frequency range of signal is automatically detected using the frange.detec function. If so, the range is used as the bandpass filter (overwriting 'bp' argument). Default is FALSE.
fsmooth	A numeric vector of length 1 to smooth the frequency spectrum with a mean sliding window (in kHz) used for frequency range detection (when frange.detec = TRUE). This help to average amplitude "hills" to minimize the effect of amplitude modulation. Default is 0.1.
raw.contour	Logical. If TRUE then a list with the original contours (i.e. without interpolating values to make all contours of equal length) is returned.
track.harm	Logical. If true warbleR's track_harm function is used to track frequency contours. Otherwise seewave's dfreq is used by default.
adjust.wl	Logical. If TRUE the 'wl' is reset to be equal at the number of samples in a selections if the samples are less than 'wl'. Default is FALSE.
...	Additional arguments to be passed to trackfreqs.

Details

This function extracts the dominant frequency values as a time series. The function uses the approx function to interpolate values between dominant frequency measures. If there are no frequencies above the amplitude theshold at the begining or end of the signals then NAs will be generated. On the other hand, if there are no frequencies above the amplitude theshold in between signal segments in which amplitude was detected then the values of this adjacent segments will be interpolated to fill out the missing values (e.g. no NAs in between detected amplitude segments).

Value

If raw.contour = TRUE (default) a data frame with the dominant frequency values measured across the signals. Otherwise, a list with the raw frequency detections (i.e. without interpolating values to make all contours of equal length) is returned. If img is TRUE it also produces image files with the spectrograms of the signals listed in the input data frame showing the location of the dominant frequencies (see trackfreqs description for more details).

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

sig2noise, trackfreqs, sp.en.ts, ffts, ffDTW, dfDTW

Other spectrogram creators: color.spectro, dfDTW, ffDTW, ffts, snrspecs, sp.en.ts, specreator, trackfreqs

Examples

```
{
# set the temp directory
# setwd(tempdir())

#load data
data(list = c("Phae.long1", "Phae.long2","selec.table"))
writeWave(Phae.long2, "Phae.long2.wav") #save sound files
writeWave(Phae.long1, "Phae.long1.wav")

# run function
dfts(X = selec.table, length.out = 30, flim = c(1, 12), bp = c(2, 9), wl = 300)

}
```

---

ffDTW *Acoustic dissimilarity using dynamic time warping on fundamental frequency contours*

Description

ffDTW calculates acoustic dissimilarity of fundamental frequency contours using dynamic time warping. Internally it applies the dtwDist function from the dtw package.

Usage

```
ffDTW(X, wl = 512, length.out = 20, wn = "hanning", ovlp = 70,
bp = c(0, 22), threshold = 5, img = TRUE, parallel = 1, path = NULL,
img.suffix = "ffDTW", pb = TRUE, clip.edges = TRUE, window.type = "none",
open.end = FALSE, scale = FALSE, ...)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame containing columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). The ouptut of manualoc or autodetec can be used as the input data frame.
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
length.out	A numeric vector of length 1 giving the number of measurements of fundamental frequency desired (the length of the time series).
wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 70.
bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz). Default is c(0, 22).

threshold	amplitude threshold (%) for fundamental frequency detection. Default is 5.
img	Logical argument. If FALSE, image files are not produced. Default is TRUE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
img.suffix	A character vector of length 1 with a sufix (label) to add at the end of the names of image files. Default is NULL.
pb	Logical argument to control progress bar. Default is TRUE.
clip.edges	Logical argument to control whether edges (start or end of signal) in which amplitude values above the threshold were not detected will be removed. If TRUE (default) this edges will be excluded and signal contour will be calculated on the remainging values. Note that DTW cannot be applied if missing values (e.i. when amplitude is not detected).
window.type	dtw windowing control parameter. Character: "none", "itakura", or a function (see dtw).
open.end	dtw control parameter. Performs open-ended alignments (see dtw).
scale	Logical. If TRUE dominant frequency values are z-transformed using the scale function, which "ignores" differences in absolute frequencies between the signals in order to focus the comparison in the frequency contour, regardless of the pitch of signals. Default is TRUE.
...	Additional arguments to be passed to trackfreqs for customizing graphical output.

Details

This function extracts the fundamental frequency values as a time series and then calculates the pairwise acoustic dissimilarity of the selections using dynamic time warping. The function uses the approx function to interpolate values between fundamental frequency measures. If 'img' is TRUE the function also produces image files with the spectrograms of the signals listed in the input data frame showing the location of the fundamental frequencies. Note that if no amplitude is detected at the begining or end of the signals then NAs will be generated. On the other hand, if amplitude is not detected in between signal segments in which amplitude was detected then the values of this adjacent segments will be interpolated to fill out the missing values (e.g. no NAs in between detected amplitude segments).

Value

A matrix with the pairwise dissimilarity values. If img is FALSE it also produces image files with the spectrograms of the signals listed in the input data frame showing the location of the fundamental frequencies.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

specreator for creating spectrograms from selections, snrspecs for creating spectrograms to optimize noise margins used in sig2noise

dfDTW dfts, ffts, dfDTW

Other spectrogram creators: color.spectro, dfDTW, dfts, ffts, snrspecs, sp.en.ts, specreator, trackfreqs

Examples

```
{
# set the temp directory
# setwd(tempdir())

#load data
data(list = c("Phae.long1", "Phae.long2","selec.table"))
writeWave(Phae.long2, "Phae.long2.wav") #save sound files
writeWave(Phae.long1, "Phae.long1.wav")

# run function
ffDTW(selec.table[1:4,], length.out = 30, flim = c(1, 12), img = TRUE, bp = c(1, 9), wl = 300)
}
```

---

ffts *Extract the fundamental frequency values as a time series*

Description

ffts extracts the fundamental frequency values as a time series of signals selected by manualoc or autodetec.

Usage

```
ffts(X, wl = 512, length.out = 20, wn = "hanning", ovlp = 70, bp = c(0, 22),
  threshold = 15, img = TRUE, parallel = 1, path = NULL, img.suffix = "ffts", pb = TRUE,
  clip.edges = FALSE, leglab = "ffts", ff.method = "seewave", ...)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame containing columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). The ouptut of manualoc or autodetec can be used as the input data frame.
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
length.out	A numeric vector of length 1 giving the number of measurements of fundamental frequency desired (the length of the time series).

wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 70.
bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz). Default is c(0, 22).
threshold	amplitude threshold (%) for fundamental frequency detection. Default is 15.
img	Logical argument. If FALSE, image files are not produced. Default is TRUE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
img.suffix	A character vector of length 1 with a suffix (label) to add at the end of the names of image files.
pb	Logical argument to control progress bar. Default is TRUE.
clip.edges	Logical argument to control whether edges (start or end of signal) in which amplitude values above the threshold were not detected will be removed. If TRUE this edges will be excluded and signal contour will be calculated on the remainging values. Default is FALSE. #' @param leglab A character vector of length 1 or 2 containing the label(s) of the frequency contour legend in the output image.
leglab	A character vector of length 1 or 2 containing the label(s) of the frequency contour legend in the output image.
ff.method	Character. Selects the method used to calculate the fundamental frequency. Either 'tuneR' (using FF) or 'seewave' (using fund). Default is 'seewave'. 'tuneR' performs faster (and seems to be more accurate) than 'seewave'.
...	Additional arguments to be passed to trackfreqs. for customizing graphical output.

Details

This function extracts the fundamental frequency values as a time series. The function uses the approx function to interpolate values between fundamental frequency #' measures. If there are no frequencies above the amplitude theshold at the begining or end of the signals then NAs will be generated. On the other hand, if there are no frequencies above the amplitude theshold in between signal segments in which amplitude was detected then the values of this adjacent segments will be interpolated to fill out the missing values (e.g. no NAs in between detected amplitude segments).

Value

A data frame with the fundamental frequency values measured across the signals. If img is TRUE it also produces image files with the spectrograms of the signals listed in the input data frame showing the location of the fundamental frequencies (see trackfreqs description for more details).

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

sig2noise, trackfreqs, dfts, ffDTW, dfDTW

Other spectrogram creators: color.spectro, dfDTW, dfts, ffDTW, snrspecs, sp.en.ts, specreator, trackfreqs

Examples

```
{
# set the temp directory
# setwd(tempdir())

#load data
data(list = c("Phae.long1", "Phae.long2","selec.table"))
writeWave(Phae.long1, "Phae.long1.wav") #save sound files
writeWave(Phae.long2, "Phae.long2.wav") #save sound files

# run function
ffts(selec.table, length.out = 50, flim = c(1, 12), bp = c(2, 9), wl = 300)

# Fundamental frequency is not accurate for noisy signals, works better with pure tones

}
```

---

filtersels *Subset selection data frames based on manually filtered image files*

Description

filtersels subsets selection data frames based on image files that have been manually filtered.

Usage

```
filtersels(X, path = NULL, lspec = FALSE, img.suffix = NULL, it = "jpeg",
 incl.wav = TRUE, missing = FALSE, index = FALSE)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "sel": number of the selections. The ouptut of manualoc or autodetec can be used as the input data frame.
path	Character string containing the directory path where the image files are located. If NULL (default) then the current working directory is used. warbleR_options 'wav.path' argument does not apply.
lspec	A logical argument indicating if the image files to be use for filtering were produced by the function lspec. All the image files that correspond to a sound file must be deleted in order to be filtered out.

img.suffix	A character vector of length 1 with the suffix (label) at the end of the names of the image files. Default is NULL (i.e. no suffix as in the images produced by specreator). Ignored if lspec = TRUE.
it	A character vector of length 1 giving the image type ("tiff", "jpeg" or "pdf") Default is "jpeg". Note that pdf files can only be generated by lspec2pdf.
incl.wav	Logical. To indicate if sound files extensions (".wav") are included (TRUE, default) or not in the image file names.
missing	Logical. Controls whether the output data frame (or row index if is index = TRUE) contains the selections with images in the working directory (Default, missing = FALSE) or the ones with no image.
index	Logical. If TRUE and missing = FALSE the row index for the selections with images in the working directory is returned. If missing = TRUE) then the row index of the ones with no image is returned instead. Default is FALSE.

Details

This function subsets selections (or sound files if lspec is TRUE) listed in a data frame based on the image files from spectrogram-creating functions (e.g. specreator) in the working directory. Only the selections/sound files with and image in the working directory will remain. This is useful for excluding selections from undesired signals. Note that the image files should be in the working directory (or the directory provided in 'path').

Value

If all .wav files are ok, returns message "All files are ok!". Otherwise returns "These file(s) cannot be read" message with names of the corrupted .wav files.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

Examples

```
## Not run:
# First set temporary folder
# setwd(tempdir())

# save wav file examples
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")

specreator(selec.table, flim = c(0, 11), inner.mar = c(4,4.5,2,1), outer.mar = c(4,2,2,1),
picsize = 2, res = 300, cexlab = 2, mar = 0.05, wl = 300)

#go to the working directory and delete some images

#filter selection data frame
fmloc <- filtersels(X = selec.table)
```

```
#this data frame does not have the selections corresponding to the images that were deleted
fmloc

#now using lspec images
lspec(sxrow = 2, rows = 8, pal = reverse.heat.colors, wl = 300, ovlp = 10)

#go to the working directory and delete lspec images (the ones with several rows of spectrograms)

#filter selection data frame


## End(Not run)
```

`fixwavs` *Fix .wav files to allow importing them into R*

Description

`fixwavs` fixes sound files in .wav format so they can be imported into R.

Usage

```
fixwavs(checksels = NULL, files = NULL, samp.rate = NULL, bit.rate = NULL,
 path = NULL, ...)
```

Arguments

`checksels`	Data frame with results from `checksels`.
`files`	Character vector with the names of the wav files to fix. Default is `NULL`.
`samp.rate`	Numeric vector of length 1 with the sampling rate (in kHz) for output files. Default is `NULL`.
`bit.rate`	Numeric vector of length 1 with the dynamic interval (i.e. bit rate) for output files. Default is `NULL`. Currently not available.
`path`	Character string containing the directory path where the sound files are located. If `NULL` (default) then the current working directory is used.
`...`	Additional arguments to be passed to `sox`.

Details

This function aims to simplify the process of converting sound files that cannot be imported into R to a format that can actually be imported. Problematic files can be determined using `checksels`. The `checksels` output can be directly input using the argument 'checksels'. Alternatively a vector of file names to be "fixed" can be provided (argument 'files'). Internally the function calls 'sox' through the `sox` function. 'sox' must be installed to be able to run this function.

Value

A folder inside the working directory (or path provided) all 'converted_sound_files', containing sound files in a format that can be imported in R.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) #last modification on march-15-2017 (MAS)

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

fixwavs(files = selec.table$sound.files)

#check this folder
getwd()

## End(Not run)
```

---

fix_extended_selection_table

Fix extended selection tables

Description

`fix_extended_selection_table` fixes extended selection tables that have lost their attributes

Usage

```
fix_extended_selection_table(X, Y)
```

Arguments

X	an object of class 'selection_table' or data frame that contains columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end).
Y	an object of class 'extended_selection_table'

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

Examples

```
{
# First set temporary folder
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))

# setwd(tempdir())

writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

# create extended selection table
ext_st <- selection_table(selec.table, extended = TRUE, confirm.extended = FALSE)

# remove attributes
st <- as.data.frame(ext_st)

# check class
class(st)

# fix selection table
st <- fix_extended_selection_table(X = st, Y = ext_st)

# check class
class(st)
}
```

---

frange *Detect frequency range iteratively*

Description

frange detect frequency range iteratively from signals in a selection table.

Usage

```
frange(X, wl = 512, it = "jpeg", line = TRUE, fsmooth = 0.1, threshold = 10,
wn = "hanning", flim = c(0, 22), bp = NULL, propwidth = FALSE, xl = 1, picsize = 1,
res = 100, fast.spec = FALSE, ovlp = 50, pal = reverse.gray.colors.2, parallel = 1,
 widths = c(2, 1), main = NULL, img = TRUE, mar = 0.05, path = NULL, pb = TRUE)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "sel": number of the selections, 3) "start": start time of selections, 4) "end": end time of selections. The ouptut of manualoc or autodetec can be used as the input data frame.
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512. This is used for calculating the frequency spectrum (using meanspec) and producing the spectrogram (using spectro, if img = TRUE).
it	A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".
line	Logical argument to add red lines (or box if bottom.freq and top.freq columns are provided) at start and end times of selection. Default is TRUE.
fsmooth	A numeric vector of length 1 to smooth the frequency spectrum with a mean sliding window in kHz. This help to average amplitude "hills" to minimize the effect of amplitude modulation. Default is 0.1.
threshold	Amplitude threshold (%) for fundamental frequency and dominant frequency detection. Default is 10.
wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options. This is used for calculating the frequency spectrum (using meanspec) and producing the spectrogram (using spectro, if img = TRUE).
flim	A numeric vector of length 2 for the frequency limit of the spectrogram (in kHz), as in spectro. Default is c(0, 22).
bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz) or "frange" to indicate that values in 'bottom.freq' and 'top.freq' columns will be used as bandpass limits. Default is c(0, 22).
propwidth	Logical argument to scale the width of spectrogram proportionally to duration of the selected call. Default is FALSE.
xl	Numeric vector of length 1. A constant by which to scale spectrogram width. Default is 1.
picsize	Numeric argument of length 1. Controls relative size of spectrogram. Default is 1.
res	Numeric argument of length 1. Controls image resolution. Default is 100 (faster) although 300 - 400 is recommended for publication/ presentation quality.
fast.spec	Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast.spec' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 50. This is used for calculating the

	frequency spectrum (using meanspec) and producing the spectrogram (using spectro, if img = TRUE).
pal	Color palette function for spectrogram. Default is reverse.gray.colors.2. See spectro for more palettes. Palettes as gray.2 may work better when fast.spec = TRUE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
widths	Numeric vector of length 2 to control the relative widths of the spectro (first element) and spectrum (second element).
main	Character vector of length 1 specifying the img title. Default is NULL.
img	Logical. Controls whether a plot is produced. Default is TRUE.
mar	Numeric vector of length 1. Specifies the margins adjacent to the selections to set spectrogram limits. Default is 0.05.
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar and messages. Default is TRUE.

Details

This functions aims to automatize the detection of frequency ranges. The frequency range is calculated as follows:

- bottom.freq = the start frequency of the first amplitude "hill"
- top.freq = the end frequency of the last amplitude "hill"

If img = TRUE a graph including a spectrogram and a frequency spectrum is produced for each selection (saved as an image file in the working directory). The graph would include gray areas in the frequency ranges exluded by the bandpass ('bp' argument), dotted lines highlighting the detected range.

Value

The original data frame with an additional 2 columns for low and high frequency values. A plot is produced in the working directory if img = TRUE (see details).

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

frange.detec, autodetec

Examples

```
{
# First set temporary folder
# setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

frange(X = selec.table, wl = 112, fsmooth = 1, threshold = 13, widths = c(4, 1),
img = TRUE, pb = TRUE, it = "tiff", line = TRUE, mar = 0.1, bp = c(1,10.5),
flim = c(0, 11))
}
```

`frange.detec` *Detect frequency range on wave objects*

Description

`frange.detec` detects the frequency range of acoustic signals on wave objects.

Usage

```
frange.detec(wave, wl = 512, fsmooth = 0.1, threshold = 10, wn = "hanning",
 flim = c(0, 22), bp = NULL, fast.spec = FALSE, ovlp = 50, pal = reverse.gray.colors.2,
 widths = c(2, 1), main = NULL, plot = TRUE, all.detec = FALSE)
```

Arguments

`wave`	A 'wave' object produced by `readWave` or similar functions.
`wl`	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512. This is used for calculating the frequency spectrum (using `meanspec`) and producing the spectrogram (using `spectro`, if `plot = TRUE`).
`fsmooth`	A numeric vector of length 1 to smooth the frequency spectrum with a mean sliding window in kHz. This help to average amplitude "hills" to minimize the effect of amplitude modulation. Default is 0.1.
`threshold`	Amplitude threshold (%) for fundamental frequency and dominant frequency detection. Default is 10.
`wn`	Character vector of length 1 specifying window name. Default is "hanning". See function `ftwindow` for more options. This is used for calculating the frequency spectrum (using `meanspec`) and producing the spectrogram (using `spectro`, if `plot = TRUE`).
`flim`	A numeric vector of length 2 for the frequency limit of the spectrogram (in kHz), as in `spectro`. Default is c(0, 22).

bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz) or "frange" to indicate that values in 'bottom.freq' and 'top.freq' columns will be used as bandpass limits. Default is c(0, 22).
fast.spec	Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast.spec' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 50. This is used for calculating the frequency spectrum (using meanspec) and producing the spectrogram (using spectro, if plot = TRUE).
pal	Color palette function for spectrogram. Default is reverse.gray.colors.2. See spectro for more palettes. Palettes as gray.2 may work better when fast.spec = TRUE.
widths	Numeric vector of length 2 to control the relative widths of the spectro (first element) and spectrum (second element).
main	Character vector of length 1 specifying the plot title. Default is NULL.
plot	Logical. Controls whether an image file is produced for each selection (in the working directory). Default is TRUE.
all.detec	Logical. If TRUE returns the start and end of all detected amplitude "hills". Otherwise only the range is returned. Default is FALSE.

Details

This functions aims to automatize the detection of frequency ranges. The frequency range is calculated as follows:

- bottom.freq = the start frequency of the first amplitude "hill"
- top.freq = the end frequency of the last amplitude "hill"

If plot = TRUE a graph including a spectrogram and a frequency spectrum is produced in the graphic device. The graph would include gray areas in the frequency ranges exluded by the bandpass ('bp' argument), dotted lines highlighting the detected range.

Value

A data frame with 2 columns for low and high frequency values. A plot is produced (in the graphic device) if plot = TRUE (see details).

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

frange, autodetec

Examples

```
{
data(tico)
frange.detec(wave = tico, wl = 512, fsmooth = 0.01, threshold = 1, bp = c(2, 8),
 widths = c(4, 2))

data(sheep)
frange.detec(wave = sheep, wl = 512, fsmooth = 0.2, threshold = 50, bp = c(0.3, 1),
flim = c(0, 1.5), pal = reverse.heat.colors, main = "sheep")
}
```

---

inflections *Count number of inflections in a frequency contour*

Description

`inflections` counts the number of inflections in a frequency contour (or any time series)

Usage

```
inflections(X = NULL, parallel = 1, pb = TRUE)
```

Arguments

`X`	data frame with the columns for "sound.files" (sound file name), "selec" (unique identifier for each selection) and columns for each of the frequency values of the contours. No other columns should be included
`parallel`	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
`pb`	Logical argument to control progress bar and messages. Default is `TRUE`.

Details

The function counts the number of inflections in a frequency contour.

Value

A data frame with 3 columns: "sound.files", "selec" and "infls" (number of inflections).

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

`dfts`, `trackfreqs`,

Examples

```
{
# Set temporary working directory

# get warbleR sound file examples
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

# measure frequency contours
dom.freq.ts <- dfts(X = selec.table)

# get number of inflections
inflections(X = dom.freq.ts)
}
```

---

is_extended_selection_table

Class 'extended_selection_table': selection table containing wave objects

---

Description

Class for selections of signals in sound files and corresponding wave objects

Usage

```
is_extended_selection_table(x)
```

Arguments

x	R object

Details

An object of class `extended_selection_table` created by `selection_table` is a list with the following elements:

- `selections`: data frame containing the frequency/time coordinates of the selections, sound file names, and any additional information
- `check.resutls`: results of the checks on data consistency using checksels
- `wave.objects`: list of wave objects corresponding to each selection
- `by.song`: a list with 1) a logical argument defining if the 'extended_selection_table' was created 'by song' and 2) the name of the song column (see `selection_table`)

Value

A logical argument indicating whether the object class is 'extended_selection_table'

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

selection_table, selection_table Check if object is of class "extended_selection_table"

is_extended_selection_table Check if the object belongs to the class "extended_selection_table"

selection_table; is_selection_table

Examples

```
{
# First set temporary folder
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))

is_extended_selection_table(selec.table)

# setwd(tempdir())

writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

st <- selection_table(selec.table, extended = TRUE, confirm.extended = FALSE)

is_extended_selection_table(st)

class(st)
}
```

---

is_selection_table *Class 'selection_table': double-checked frequency/time coordinates of selections*

Description

Class for selections of signals in sound files

Usage

```
is_selection_table(x)
```

Arguments

x	R object.

Details

An object of class `selection_table` created by `selection_table` is a list with the following elements:

- `selections`: data frame containing the frequency/time coordinates of the selections, sound file names, and any additional information
- `check.resutls`: results of the checks on data consistency using checksels

Value

A logical argument indicating whether the object class is 'selection_table'

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

`selection_table` Check if object is of class "selection_table"

`is_selection_table` Check if the object belongs to the class "selection_table"

`selection_table`

Examples

```
{
# First set temporary folder
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))

is_selection_table(selec.table)

# setwd(tempdir())

writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

st <- selection_table(selec.table)

is_selection_table(st)

class(st)
}
```

lspec *Create long spectrograms of whole sound files*

Description

lspec produces image files with spectrograms of whole sound files split into multiple rows.

Usage

```
lspec(X = NULL, flim = c(0,22), sxrow = 5, rows = 10, collevels = seq(-40, 0, 1),
ovlp = 50, parallel = 1, wl = 512, gr = FALSE, pal = reverse.gray.colors.2,
cex = 1, it = "jpeg", flist = NULL, redo = TRUE, path = NULL, pb = TRUE,
fast.spec = FALSE)
```

Arguments

X	'selection_table' object or data frame with results from manualoc or any data frame with columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). If given, two red dotted lines are plotted at the start and end of a selection and the selections are labeled with the selection number (and selection comment, if available). Default is NULL.
flim	A numeric vector of length 2 indicating the highest and lowest frequency limits (kHz) of the spectrogram, as in spectro. Default is c(0,22).
sxrow	A numeric vector of length 1. Specifies seconds of spectrogram per row. Default is 5.
rows	A numeric vector of length 1. Specifies number of rows per image file. Default is 10.
collevels	A numeric vector of length 3. Specifies levels to partition the amplitude range of the spectrogram (in dB). The more levels the higher the resolution of the spectrogram. Default is seq(-40, 0, 1).
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 50. High values of ovlp slow down the function but produce more accurate selection limits (when X is provided).
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
gr	Logical argument to add grid to spectrogram. Default is FALSE.
pal	Color palette function for spectrogram. Default is reverse.gray.colors.2. See spectro for more palettes.
cex	A numeric vector of length 1 giving the amount by which text (including sound file and page number) should be magnified. Default is 1.
it	A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".

flist	character vector or factor indicating the subset of files that will be analyzed. Ignored if X is provided.
redo	Logical argument. If TRUE all selections will be analyzed again when code is rerun. If FALSE only the selections that do not have a image file in the working directory will be analyzed. Default is FALSE.
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.
fast.spec	Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.

Details

The function creates spectrograms for complete sound files, printing the name of the sound files and the "page" number (p1-p2...) at the upper right corner of the image files. If results from manualoc are supplied (or an equivalent data frame), the function delimits and labels the selections. This function aims to facilitate visual inspection of multiple files as well as visual classification of vocalization units and the analysis of animal vocal sequences.

Value

image files with spectrograms of whole sound files in the working directory. Multiple pages can be returned, depending on the length of each sound file.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

lspec2pdf, catalog2pdf, https://marce10.github.io/2017-01-07-Create_pdf_files_with_spectrograms_of_full_recordings/

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())

# save sound file examples
data(list = c("Phae.long1", "Phae.long2","selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")

lspec(sxrow = 2, rows = 8, pal = reverse.heat.colors, wl = 300)
```

```
# including selections
lspec(sxrow = 2, rows = 8, X = selec.table, pal = reverse.heat.colors, redo = TRUE, wl = 300)

#check this floder
getwd()

## End(Not run)
```

---

lspec2pdf lspec2pdf *combines* lspec *images in .jpeg format to a single pdf file.*

Description

lspec2pdf combines lspec images in .jpeg format to a single pdf file.

Usage

```
lspec2pdf(keep.img = TRUE, overwrite = FALSE, parallel = 1, path = NULL, pb = TRUE)
```

Arguments

keep.img	Logical argument. Indicates whether jpeg files should be kept (default) or remove. (including sound file and page number) should be magnified. Default is 1.
overwrite	Logical argument. If TRUE all jpeg pdf will be produced again when code is rerun. If FALSE only the ones missing will be produced. Default is FALSE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.

Details

The function combines spectrograms for complete sound files from the lspec function into a single pdf (for each sound file).

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

lspec, catalog2pdf, https://marce10.github.io/2017-01-07-Create_pdf_files_with_spectrograms_of_full_recordings/

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())

# save sound file examples
data(list = c("Phae.long1", "Phae.long2"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")

lspec(sxrow = 2, rows = 8, pal = reverse.heat.colors, wl = 300, it = "jpeg")

#now create single pdf removing jpeg
lspec2pdf(keep.img = FALSE)

# check this floder
getwd()

## End(Not run)
```

---

manualoc *Interactive view of spectrograms*

Description

manualoc produces an interactive spectrographic view in which the start and end times of acoustic signals can be measured.

Usage

```
manualoc(wl = 512, flim = c(0,12), seltime = 1, tdisp = NULL, reccomm =
  FALSE, wn = "hanning", title = TRUE, selcomm = FALSE, osci = FALSE, player =
  NULL, pal = reverse.gray.colors.2, path = NULL, flist = NULL,
  fast.spec = FALSE, ext.window = TRUE, width = 15, height = 5)
```

Arguments

wl	A numeric vector of length 1 specifying the spectrogram window length. Default is 512.
flim	A numeric vector of length 2 specifying the frequency limit (in kHz) of the spectrogram, as in the function spectro. Default is c(0,12).
seltime	A numeric vector of length 1 indicating the time interval in seconds at which the spectrograms are produced with higher resolution (ovlp = 70) and oscilograms (if osci = TRUE). Default is 1 second.
tdisp	A numeric vector of length 1 specifying the length in seconds of the total sound file to be displayed. Default is NULL which displays the full sound file.

reccomm	Logical argument. If TRUE pops up a comment window at the end of each sound file. The comment needs to be quoted. Default is FALSE.
wn	A character vector of length 1 specifying the window function (by default "hanning"). See function ftwindow for more options.
title	Logical argument. If TRUE the name of the sound file will be printed as the main title of the spectrogram window. Default is TRUE
selcomm	Logical argument. If TRUE pops up a comment window after each selection. The comment is printed as a label on the selected unit. The comment must be quoted. Default is FALSE
osci	Logical argument. If TRUE adds a oscillogram whenever the spectrograms are produced with higher resolution (see seltime). Default is FALSE.
player	Path to or name of a program capable of playing a wave file by invocation from the command line. If under Windows and no player is given, windows player will be chosen as the default. "vlc" works in Linux if vlc player is installed. The external program must be closed before resuming analysis. Default is NULL.
pal	A color palette function to be used to assign colors in the plot, as in spectro. Default is reverse.gray.colors.2. See Details.
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
flist	character vector or factor indicating the subset of files that will be analyzed. Ignored if X is provided.
fast.spec	Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.
ext.window	Logical. If TRUE then and external graphic window is used. Default dimensions can be set using the 'width' and 'height' arguments. Default is TRUE.
width	Numeric of length 1 controling the width of the external graphic window. Ignored if ext.window = FALSE. Default is 15.
height	Numeric of length 1 controling the height of the external graphic window. Ignored if ext.window = FALSE. Default is 5.

Details

Users can zoom-in a specific sound file segment by clicking at the start and end (left side and right side) of the segment. To select the start and end of a vocalization unit the users need to click at the end and then at the start (right side and left side) of the unit. In addition, 6 "buttons" are provided at the upper right side of the spectrogram that allow to display a full view of the spectrogram ("Full view"), go back to the previous view ("Previous view"), stop the analysis ("Stop"), go to the next sound file ("Next rec"), play the current view using external software ("Play", see "player" argument), or delete the last manual selection in the current sound file ("Delete"). When a unit has been selected, the function plots a red circle with the selection number in the middle point

of the selection in the spectrogram. It also plots vertical dotted lines at the start and end of the selection. The circle and lines "disappear" when the selection is deleted ("Delete" button). Only the last selection can be deleted.

The function produces a .csv file (manualoc_output.csv) with information about the .wav file name, selection number, start and end time, selection comment (selcomm), and sound file comment (reccomm). The file is saved in the working directory and is updated every time the user moves into the next sound file (Next rec "button") or stop the process (Stop "button"). When resuming the process (after "stop" and re-running the function in the same working directory), the function will keep the previous selections and will only pick up .wav files that are not present in the .csv file (not previously analyzed). When users go to the next sound file (Next rec "button") without making any selection the file is still included in the .csv file, with NA's in the "end", "time" and "selec" field.

Windows length (wl) controls the temporal and frequency precision of the spectrogram. A high "wl" value increases the frequency resolution but reduces the temporal resolution, and vice versa. Any color palette that comes with the seewave package can be used: temp.colors, reverse.gray.colors.1, reverse.gray.colors.2, reverse.heat.colors, reverse.terrain.colors, reverse.topo.colors, reverse.cm.colors, heat.colors, terrain.colors, topo.colors, cm.colors. The function is slow when working on files of length > 5min. In most cases other sound analysis softwares for manually selecting acoustic signals (e.g. Raven, Syrinx) should be prefered.

Value

.csv file saved in the working directory with start and end time of selections.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

seltailor

Examples

```
## Not run:
#Set temporary working directory
# setwd(tempdir())

# save wav file examples
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

manualoc(wl = 300)
# need to use the buttoms to manipulate function
# check working directory for .csv file after stopping function
#check here:
getwd()
```

```
## End(Not run)
```

`move.imgs` *Move/copy image files between directories*

Description

`move.imgs` moves/copies image files created by `warbleR` between directories (folders).

Usage

```
move.imgs(from = NULL, to = NULL, it = "all", cut = TRUE,
overwrite = FALSE, create.folder = TRUE, folder.name = "image_files",
parallel = 1, pb = TRUE)
```

Arguments

`from`	Directory path where image files to be copied are found. If `NULL` (default) then the current working directory is used.
`to`	Directory path where image files will be copied to.
`it`	A character vector of length 1 giving the image type to be used. "all", "tiff", "jpeg" and "pdf" are admitted ("all" includes all the rest). Default is "all".
`cut`	Logical. Determines if files are removed from the original location after being copied (cut) or not (just copied). Default is `TRUE`.
`overwrite`	Logical. Determines if files that already exist in the destination directory should be overwritten. Default is `FALSE`.
`create.folder`	Logical. Determines if files are moved to a new folder (which is named with the "folder.name" argument). Ignored if 'to' is provided. Default is `TRUE`.
`folder.name`	Character string with the name of the new folder where the files will be copied to. Ignored if 'to' is provided. Default is `"image_files"`.
`parallel`	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
`pb`	Logical argument to control progress bar. Default is `TRUE`.

Details

This function aims to simplify the manipulation of the image files generated by many of the `warbleR` function. It copies/cuts files between directories.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

filtersels

Other data manipulation: open_wd

Examples

```
{
#Set temporary working directory
# setwd(tempdir())

#load data
data("Cryp.soui")
writeWave(Cryp.soui, "Cryp.soui.wav") #save sound files

#autodetec location of signals
ad <- autodetec(threshold = 6, bp = c(1, 3), mindur = 1.2,
maxdur = 3, img = FALSE, ssmooth = 600, wl = 300, flist = "Cryp.soui.wav")

#track dominant frequency graphs with freq reange detection
trackfreqs(X = ad[!is.na(ad$start),], flim = c(0, 5), ovlp = 90, it = "tiff",
bp = c(1, 3), contour = "df", wl = 300, frange = TRUE)

#copy files
move.imgs(cut = FALSE)

#cut files
move.imgs(cut = TRUE, to = "image_files")

# Check this folder
getwd()
}
```

---

mp32wav *Convert .mp3 files to .wav*

Description

mp32wav converts several .mp3 files in working directory to .wav format

Usage

```
mp32wav(samp.rate = 44.1, parallel = 1, from = NULL, to = NULL,
normalize = NULL, pb = TRUE)
```

Arguments

samp.rate	Sampling rate at which the .wav files should be written. The maximum permitted is 44.1 kHz (default). Units should be kHz.

parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
from	Character string containing the directory path where the .mp3 files are located. If NULL (default) then the current working directory is used.
to	Character string containing the directory path where the .wav files will be saved. If NULL (default) then the current working directory is used.
normalize	Character string containing the units to be used for amplitude normalization. Check (normalize) for details. If NULL (default) no normalization is carried out.
pb	Logical argument to control progress bar. Default is TRUE.

Details

convert all .mp3 files in working directory to .wav format. Function used internally to read .mp3 files (readMP3) sometimes crashes.

Value

.wav files saved in the working directory with same name as original mp3 files.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) and Grace Smith Vidaurre

Examples

```
## Not run:
# First set temporary folder
# setwd(tempdir())

#Then download mp3 files from xeno-canto
querxc(qword = "Phaethornis aethopygus", download = TRUE)

# Convert all files to .wav format
mp32wav()

#check this folder!!
getwd()

## End(Not run)
```

---

open_wd *Open working directory*

Description

open_wd opens the working directory in the default file browser.

Usage

```
open_wd(path = getwd(), verbose = TRUE)
```

Arguments

path	Directory path to be opened. By default it's the working directory. 'wav.path' set by warbleR_options is ignored in this case.
verbose	Logical to control whether the 'path' is printed in the console.

Details

The function opens the working directory using the default file browser and prints the working directory in the R console. This function aims to simplify the manipulation of sound files and other files produced by many of the warbleR function.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

move.imgs

Other data manipulation: move.imgs

Examples

```
{
open_wd()
}
```

---

ovlp_sels *Find overlapping selections*

Description

ovlp_sels finds which selections overlap in time within a given sound file.

Usage

```
ovlp_sels(X, index = FALSE, pb = TRUE, max.ovlp = 0, relabel = FALSE,
drop = FALSE, priority = NULL, priority.col = NULL, unique.labs = TRUE,
indx.row = FALSE, parallel = 1)
```

Arguments

X	'selection_table' object or data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "selec": number of the selections, 3) "start": start time of selections, 4) "end": end time of selections. The ouptut of manualoc or autodetec can be used as the input data frame. Other data frames can be used as input, but must have at least the 4 columns mentioned above.
index	Logical. Indicates if only the index of the overlapping selections would be returned. Default is FALSE.
pb	Logical argument to control progress bar and messages. Default is TRUE.
max.ovlp	Numeric vector of length 1 specifying the maximum overlap allowed (in seconds) . Default is 0.
relabel	Logical. If TRUE then selections names (selec column) are reset. Default is FALSE.
drop	Logical. If TRUE, when 2 or more selections overlap the function will remove all but one of the overlapping selection. Default is FALSE.
priority	Character vector. Controls the priority criteria used for removing overlapped selections. It must list the levels of the column used to determine priority (argument priority.col) in the desired priority order. Default is NULL.
priority.col	Character vector of length 1 with the name of the column use to determine the priority of overlapped selections. Default is NULL.
unique.labs	Logical to control if labels are reused across different sound files (if TRUE, default).
indx.row	Logical. If TRUE then a character column with the indices of all selections that overlapped with each selection is added to the ouput data frame (if index = TRUE). For instance, if the selections in rows 1,2 and 3 all overlapped with each other, the 'indx.row' value would be "1/2/3" for all. However, if selection 3 only overlaps with 2 but not with 1, then it returns, "1/2" for row 1, "1/2/3" for row 2, and "2/3" for row 3. Default is FALSE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).

Details

This function detects selections within a selection table that overlap in time. Selections must be listed in a data frame similar to selec.table.

Value

A data frame with the columns in X plus an additional column ('ovlp_sels') indicating which selections overlap. The ones with the same number overlap with each other. If drop = TRUE only the non-overlapping selections are return. If 2 or more selections overlap only the first one is kept.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

filtersels selec.table

Examples

```
{
#no overlap
ovlp_sels(X =  selec.table)

# modified selec.table to make the first and second selection overlap
Y <- selec.table
Y$end[4] <- 1.5

 ovlp_sels(X =  Y)

# drop overlapping
 ovlp_sels(X =  Y, drop = TRUE)

# get index instead
 ovlp_sels(X =  Y, index = TRUE)
}
```

---

querxc *Access 'Xeno-Canto' recordings and metadata*

Description

querxc downloads recordings and metadata from 'Xeno-Canto' (https://www.xeno-canto.org/).

Usage

```
querxc(qword, download = FALSE, X = NULL, file.name = c("Genus", "Specific_epithet"),
parallel = 1, path = NULL, pb = TRUE)
```

Arguments

qword	Character vector of length one indicating the genus, or genus and species, to query 'Xeno-Canto' database. For example, *Phaethornis* or *Phaethornis longirostris*. (https://www.xeno-canto.org/). More complex queries can be done by using search terms that follow the xeno-canto advance query syntax.This syntax uses tags to search within a particular aspect of the recordings (e.g. country, location, sound type). Tags are of the form tag:searchterm'. For instance, 'type:song' will search for all recordings in which the sound type description contains the word 'song'. Several tags can be included in the same query. The query "phaethornis cnt:belize' will only return results for birds in the genus *Phaethornis* that were recorded in Belize. See https://www.xeno-canto.org/help/search for a full description and see examples below for queries using terms with more than one word.

download	Logical argument. If FALSE only the recording file names and associated metadata are downloaded. If TRUE, recordings are also downloaded to the working directory as .mp3 files. Default is FALSE. Note that if the recording is already in the working directory (as when the downloading process has been interrupted) it will be skipped. Hence, resuming downloading processes will not start from scratch.
X	Data frame with a 'Recording_ID' column and any other column listed in the file.name argument. Only the recordings listed in the data frame will be downloaded (download argument is automatically set to TRUE). This can be used to select the recordings to be downloaded based on their attributes.
file.name	Character vector indicating the tags (or column names) to be included in the sound file names (if download = TRUE). Several tags can be included. If NULL only the 'Xeno-Canto' recording identification number ("Recording_ID") is used. Default is c("Genus", "Specific_epithet"). Note that recording id is always used (whether or not is listed by users) to avoid duplicated names.
parallel	Numeric. Controls whether parallel computing is applied when downloading mp3 files. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing). Applied both when getting metadata and downloading files.
path	Character string containing the directory path where the sound files will be saved. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.

Details

This function queries for avian vocalization recordings in the open-access online repository 'Xeno-Canto' (https://www.xeno-canto.org/). It can return recordings metadata or download the associated sound files. Complex queries can be done by using search terms that follow the xeno-canto advance query syntax (check "qword" argument description). Files are double-checked after downloading and "empty" files are re-downloaded. File downloading process can be interrupted and resume later as long as the working directory is the same. Maps of recording coordinates can be produced using xcmaps.

Value

If X is not provided the function returns a data frame with the following recording information: recording ID, Genus, Specific epithet, Subspecies, English name, Recordist, Country, Locality, Latitude, Longitude, Vocalization type, Audio file, License, URL, Quality, Time, Date. Sound files in .mp3 format are downloaded into the working directory if download = TRUE or if X is provided; a column indicating the names of the downloaded files is included in the output data frame.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

xcmaps, https://marce10.github.io/2016/12/22/Download_a_single_recording_for_each_species_in_a_site_from_Xeno-Canto.html

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())

# search without downloading
df1 <- querxc(qword = 'Phaethornis anthophilus', download = FALSE)
View(df1)

# downloading files
querxc(qword = 'Phaethornis anthophilus', download = TRUE)

# check this folder
getwd()

## search using xeno-canto advance query ###
orth.pap <- querxc(qword = 'gen:orthonyx cnt:papua loc:tari', download = FALSE)

# download file using the output data frame as input
querxc(X = orth.pap)

# use quotes for queries with more than 1 word (e.g. Costa Rica),note that the
# single quotes are used for the whole 'qword' and double quotes for the 2-word term inside
#Phaeochroa genus in Costa Rica
phae.cr <- querxc(qword = 'gen:phaeochroa cnt:"costa rica"', download = FALSE)

# several terms can be searched for in the same field
# search for all female songs in sound type
femsong <- querxc(qword = 'type:song type:female', download = FALSE)

## End(Not run)
```

`read_wave` *A wrapper for tuneR's readWave that read sound files listed within selection tables*

Description

`read_wave` A wrapper for tuneR's `readWave` function that read sound files listed within selection tables

Usage

```
read_wave(X, index, from = X$start[index], to = X$end[index], header = FALSE, path = NULL)
```

Arguments

`X`	'data.frame', 'selection_table' or 'extended_selection_table' containing columns for sound file name (sound.files), selection number (selec), and start and end

	time of signals (start and end). Low and high frequency columns are optional. Default is NULL.
index	Index of the selection in 'X' that will be read. Ignored if 'X' is NULL.
from	Where to start reading, in seconds. Default is X$start[index].
to	Where to stop reading, in seconds. Default is X$end[index].
header	If TRUE, only the header information of the Wave object is returned, otherwise (the default) the whole Wave object.
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.

Details

The function is a wrapper for readWave that read sound files listed within selection tables ignores file extension mismatches, a common mistake when reading wave files. It is also used internally by warbleR functions to read wave objects from extended selection tables (see selection_table for details).

Value

An object of class "Wave".

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

Examples

```
{
# First set temporary folder
# setwd(tempdir())

# write wave files with lower case file extension
data(list = c("Phae.long1"))
writeWave(Phae.long1,"Phae.long1.wav")

read_wave(X = selec.table, index  =  1)
}
```

---

rm_sil *Remove silence in wave files*

Description

rm_sil

Usage

```
rm_sil(path = NULL, min.sil.dur = 2, img = TRUE, it = "jpeg", flim = c(0, 12),
flist = NULL, parallel = 1, pb = TRUE)
```

Arguments

path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
min.sil.dur	Numeric. Controls the minimum duration of silence segments that would be removed.
img	Logical argument. If FALSE, image files are not produced. Default is TRUE.
it	A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".
flim	A numeric vector of length 2 indicating the highest and lowest frequency limits (kHz) of the spectrogram as in spectro. Default is c(0,12). Ignored if 'img = FALSE'.
flist	character vector or factor indicating the subset of files that will be analyzed. If not provided then all wave files in the working directory (or path) will be processed.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
pb	Logical argument to control progress bar and messages. Default is TRUE.

Details

The function removes silence segments (i.e. segments with very low amplitude values) from wave files.

Value

Sound files for which silence segments have been removed are saved in the new folder "removed_silence_files". If 'img = TRUE' then spectrogram images highlighting the silence segments that were removed are also saved.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

fixwavs, autodetec,

Examples

```
{
# Set temporary working directory
# setwd(tempdir())

# save sound file examples
data(list = c("Phae.long1", "Phae.long2","selec.table"))
sil <- silence(samp.rate = 22500, duration = 3, xunit = "time")

wv1 <- pastew(pastew(Phae.long1, sil, f = 22500, output = "Wave"),
Phae.long2, f = 22500, output = "Wave")

#check silence in between amplitude peaks
env(wv1)

 #save wave file
 writeWave(object = wv1, filename = "wv1.wav", extensible = FALSE)

#remove silence
rm_sil(flist = "wv1.wav")

# OR this is tempdir was used instead
# rm_sil(path = tempdir(), flist = "wv1.wav")

#check this floder
getwd()
}
```

`selec.table` *Data frame of selections (i.e. selection table).*

Description

A data frame containing the start, end, low and hig frequency of *Phaethornis longirostris* (Long-billed Hermit) songs from the example sound files included in this package. Same data than 'selec_table'. 'selec.table' will be reomved in future package version.

Usage

```
data(selec.table)
```

Format

A data frame with 11 rows and 6 variables:

sound.files recording names

channel channel in which signal is found

selec selection numbers within recording

start start times of selected signal

end end times of selected signal

bottom.freq lower limit of frequency range

top.freq upper limit of frequency range

sel.comment selection comments

rec.comment recording comments

Source

Marcelo Araya Salas, warbleR

`selection_table` *Create 'selection_table' and 'extended_selection_table' objects*

Description

`selection_table` converts data frames into an object of classes 'selection_table' or 'extended_selection_table'.

Usage

```
selection_table(X, max.dur = 10, path = NULL, whole.recs = FALSE,
extended = FALSE, confirm.extended = TRUE, mar = 0.1, by.song = NULL,
pb = TRUE, parallel = 1, ...)
```

Arguments

`X`	data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "selec": unique selection identifier (within a sound file), 3) "start": start time and 4) "end": end time of selections. Columns for 'top.freq', 'bottom.freq' and 'channel' are optional. Alternatively, a 'selection_table' class object can be input to double check selections. The ouptut of `manualoc` or `autodetec` can be used as the input object for other `warbleR` functions.
`max.dur`	the maximum duration of expected for a selection (ie. end - start). If surpassed then an error message will be generated. Useful for detecting errors in selection tables.
`path`	Character string containing the directory path where the sound files are located. If `NULL` (default) then the current working directory is used.
`whole.recs`	Logical. If `TRUE` the function will create a selection table for all sound files in the working directory (or "path") with 'start = 0' and 'end = wavdur()'. Default is if `FALSE`.
`extended`	Logical. If `TRUE`, the function will create an object of class 'extended_selection_table' which included the wave objects of the selections as an additional attribute ('wave.objects') to the data set. This is and self-contained format that does not require the original sound files for running most acoustic analysis in `warbleR`. This can largely faciliate the storing and sharing of (bio)acoustic data. Default is if `FALSE`. An extended selection table won't be created if there is any issue with the selection. See 'details'.

confirm.extended
: Logical. If TRUE then the size of the 'extended_selection_table' will be estimated and the user will be asked for confirmation (in the console) before proceeding. Ignored if 'extended' is FALSE. This is used to prevent generating objects too big to be dealth with by R. See 'details' for more information about extended selection table size.

mar
: Numeric vector of length 1 specifying the margins (in seconds) adjacent to the start and end points of the selections when creating extended selection tables. Default is 0.1. Ignored if 'extended' is FALSE.

by.song
: Character string with the column name containing song labels. If provided a wave object containing for all selection belonging to a single song would be saved in the extended selection table (hence only applicable for extended selection tables). Note that the function assumes that song labels are not repeated within a sound file. If NULL (default), wave objects are created for each selection (e.g. by selection). Ignored if extended = FALSE.

pb
: Logical argument to control progress bar and messages. Default is TRUE.

parallel
: Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).

...
: Additional arguments to be passed to checksels for customizing checking routine.

Details

This function creates and object of class 'selection_table' or 'extended_selection_table' (if extended = TRUE, see below). First, the function checks:

- 1) if the selections listed in the data frame correspond to .wav files in the working directory
- 2) if the sound files can be read and if so,
- 3) if the start and end time of the selections are found within the duration of the sound files

If no errors are found the a selection table or extended selection table will be generated. Note that the sound files should be in the working directory (or the directory provided in 'path'). This is useful for avoiding errors in dowstream functions (e.g. specan, xcorr, catalog, dfDTW). Note also that corrupt files can be fixed using fixwavs ('sox' must be installed to be able to run this function). The 'selection_table' class can be input in subsequent functions.

When extended = TRUE the function will generate an object of class 'extended_selection_table' which will also contains the wave objects for each of the selections in the data frame. This transforms selection tables into self-contained objects as they no longer need the original sound files to run acoustic analysis. This can largely faciliate the storing and sharing of (bio)acoustic data. Extended selection table size will be a function of the number of selections nrow(X), sampling rate, selection duration and margin duration. As a guide, a selection table with 1000 selections similar to the ones in 'selec.table' (mean duration ~0.15 seconds) at 22.5 kHz sampling rate and the default margin (mar = 0.1) will generate a extended selection table of ~31 MB (~310 MB for a 10000 rows selection table). You can check the size of the output extended selection table with the object.size function. Note that extended selection table created 'by.song' could be considerable larger.

Value

An object of class selection_table which includes the original data frame plus the following additional attributes:

- 1) A data frame with the output of checksels run on the input data frame. If a extended selection table is created it will also include the original values in the input data frame for each selections. This are used by downstream warbleR functions to improve efficiency and avoid errors due to missing or mislabeled data, or selection out of the ranges of the original sound files.
- 2) A list indicating if the selection table has been created by song (see 'by.song argument).
- 3) If a extended selection table is created a list containing the wave objects for each selection (or song if 'by.song').

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

checkwavs

Examples

```
{
# First set temporary folder
 setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

# make selection table
st <- selection_table(X = selec.table)

is_selection_table(st)

#' # make extended selection table
st <- selection_table(X = selec.table, extended = TRUE, confirm.extended = FALSE)

is_extended_selection_table(st)

### make extended selection by song
# create a song variable
selec.table$song <- as.numeric(selec.table$sound.files)

st <- selection_table(X = selec.table, extended = TRUE, confirm.extended = FALSE, by.song = "song")
}
```

selec_table *Data frame of selections (i.e. selection table).*

Description

A data frame containing the start, end, low and hig frequency of *Phaethornis longirostris* (Long-billed Hermit) songs from the example sound files included in this package. Same data than 'selec.table'.

Usage

```
data(selec_table)
```

Format

A data frame with 11 rows and 6 variables:

sound.files recording names

channel channel in which signal is found

selec selection numbers within recording

start start times of selected signal

end end times of selected signal

bottom.freq lower limit of frequency range

top.freq upper limit of frequency range

sel.comment selection comments

rec.comment recording comments

Source

Marcelo Araya Salas, warbleR

seltailor *Interactive view of spectrograms to tailor selections*

Description

`seltailor` produces an interactive spectrographic view (similar to `manualoc`) in which the start/end times and frequency range of acoustic signals listed in a data frame can be adjusted.

Usage

```
seltailor(X = NULL, wl = 512, flim = c(0,22), wn = "hanning", mar = 0.5,
 osci = TRUE, pal = reverse.gray.colors.2, ovlp = 70, auto.next = FALSE, pause = 1,
  comments = TRUE, path = NULL, frange = FALSE, fast.spec = FALSE, ext.window = TRUE,
   width = 15, height = 5, index = NULL, collevels = NULL,
   title = c("sound.files", "selec"), ts.df = NULL, col = "#E37222",
   alpha = 0.7, auto.contour = FALSE, ...)
```

Arguments

X	'selection_table' object or data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "selec": number of the selections, 3) "start": start time of selections, 4) "end": end time of selections. The ouptut of manualoc or autodetec can be used as the input data frame. Other data frames can be used as input, but must have at least the 4 columns mentioned above. Notice that, if an output file ("seltailor_output.csv") is found in the working directory it will be given priority over an input data frame.
wl	A numeric vector of length 1 specifying the spectrogram window length. Default is 512.
flim	A numeric vector of length 2 specifying the frequency limit (in kHz) of the spectrogram, as in the function spectro. Default is c(0,22).
wn	A character vector of length 1 specifying the window function (by default "hanning"). See function ftwindow for more options.
mar	Numeric vector of length 1. Specifies the margins adjacent to the start and end points of the selections to define spectrogram limits. Default is 0.5.
osci	Logical argument. If TRUE adds a oscillogram whenever the spectrograms are produced with higher resolution (see seltime). Default is TRUE. The external program must be closed before resuming analysis. Default is NULL.
pal	A color palette function to be used to assign colors in the plot, as in spectro. Default is reverse.gray.colors.2. See Details.
ovlp	Numeric vector of length 1 specifying the percent overlap between two consecutive windows, as in spectro. Default is 70.
auto.next	Logical argument to control whether the functions moves automatically to the next selection. The time interval before moving to the next selection is controled by the 'pause' argument. Ignored if ts.df = TRUE.
pause	Numeric vector of length 1. Controls the duration of the waiting period before moving to the next selection (in seconds). Default is 1.
comments	Logical argument specifying if 'sel.comment' (when in data frame) should be included in the title of the spectrograms. Default is TRUE.
path	Character string containing the directory path where the sound files are located.
frange	Logical argument specifying whether limits on frequency range should be recorded. If NULL (default) then only the time limits are recorded.
fast.spec	Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options

become unavailable, as sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.

ext.window — Logical. If TRUE then and external graphic window is used. Default dimensions can be set using the 'width' and 'height' arguments. Default is TRUE.

width — Numeric of length 1 controling the width of the external graphic window. Ignored if ext.window = FALSE. Default is 15.

height — Numeric of length 1 controling the height of the external graphic window. Ignored if ext.window = FALSE. Default is 5.

index — Numeric vector indicating which selections (rows) of 'X' should be tailored. Default is NULL. Ignored when the process is resumed. This can be useful when combined with filtersels) output (see 'index' argument in filtersels).

collevels — Numeric. Set of levels used to partition the amplitude range (see spectro).

title — Character vector with the names of the columns to be included in the title for each selection.

ts.df — Optional. Data frame with frequency contour time series of signals to be tailored. If provided then 'autonext' is set to FALSE. Default is NULL. The data frame must include the 'sound.files' and 'selec' columns for the same selections included in 'X'.

col — Character vector defining the color of the points when 'ts.df' is provided. Default is "#E37222" (orange).

alpha — Numeric of length one to adjust transparency of points when adjusting frequency contours.

auto.contour — Logical. If TRUE contours are displayed automatically (without having to click on 'contour'). Note that adjusting the selection box (frequency/time limits) won't be available. Default is FALSE. Ignored if 'ts.df' is not provided.

... — Additional arguments to be passed to the internal spectrogram creating function for customizing graphical output. The function is a modified version of spectro, so it takes the same arguments.

Details

This function produces an interactive spectrographic view in which users can select new time/frequency coordinates the selections. 4 "buttons" are provided at the upper right side of the spectrogram that allow to stop the analysis ("stop"), go to the next sound file ("next"), return to the previous selection ("previous") or delete the current selection ("delete"). An additional "button" ("contour") to tailored frequency contour is shown when 'ts.df' is provided. When a unit has been selected, the function plots dotted lines in the start and end of the selection in the spectrogram (or a box if frange = TRUE). Only the last selection is kept for each selection that is adjusted. The function produces a .csv file (seltailor_output.csv) with the same information than the input data frame, except for the new time coordinates, plus a new column (X$tailored) indicating if the selection has been tailored. The file is saved in the working directory and is updated every time the user moves into the next sound file (next sel "button") or stop the process (Stop "button"). It also return the same data frame as and

object in the R environment. If no selection is made (by clicking on the 'next' button) the original time/frequency coordinates are kept. When resuming the process (after "stop" and re-running the function in the same working directory), the function will continue working on the selections that have not been analyzed. The function also displays a progress bar right on top of the sepctrogram. The zoom can be adjusted by setting the `mar` argument. To fix contours a data.frame containing the 'sound.files' and 'selec' columns as in 'X' as well as the frequency values at each contour step must be provided. The function plots points correponding to the time/frequency coordinates of each element of the contour. Cliking on the spectrogram will substitute the frequency value of the points. The contour point closest in time to the "click" will be replaced by the frequency value of the "click".

Value

data frame similar to X with the and a .csv file saved in the working directory with start and end time of selections.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

manualoc

Examples

```
## Not run:
#Set temporary working directory
# setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

seltailor(X =  selec.table, flim = c(1,12), wl = 300, auto.next = TRUE)

# Read output .csv file
seltailor.df <- read.csv("seltailor_output.csv")
seltailor.df

# check this directory for .csv file after stopping function
getwd()

## End(Not run)
```

sig2noise *Measure signal-to-noise ratio*

Description

sig2noise measures signal-to-noise ratio across multiple files.

Usage

```
sig2noise(X, mar, parallel = 1, path = NULL, pb = TRUE, type = 1, eq.dur = FALSE,
in.dB = TRUE, before = FALSE, lim.dB = TRUE, bp = NULL, wl = 10)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame with results from manualoc or any data frame with columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end).
mar	numeric vector of length 1. Specifies the margins adjacent to the start and end points of selection over which to measure noise.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing). It can also be set globally using the 'parallel' option (see warbleR_options).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used. It can also be set globally using the 'wav.path' option (see warbleR_options).
pb	Logical argument to control if progress bar is shown. Default is TRUE. It can also be set globally using the 'pb' option (see warbleR_options).
type	Numeric. Determine the formula to be used to calculate the signal-to-noise ratio (S = signal , N = background noise): • 1: ratio of S mean amplitude envelope to N mean amplitude envelope (mean(env(S))/mean(env(N))) • 2: ratio of S amplitude envelope quadratic mean to N amplitude envelope quadratic mean (rms(env(S))/rms(env(N))) • 3: ratio of the difference between S amplitude envelope quadratic mean and N amplitude envelope quadratic mean to N amplitude envelope quadratic mean ((rms(env(S)) - rms(env(N)))/rms(env(N)))
eq.dur	Logical. Controls whether the noise segment that is measured has the same duration than the signal (if TRUE, default FALSE). If TRUE then 'mar' argument is ignored.
in.dB	Logical. Controls whether the signal-to-noise ratio is returned in decibels (20*log10(SNR)). Default is TRUE.
before	Logical. If TRUE noise is only measured right before the signal (instead of before and after). Default is FALSE.

lim.dB — Logical. If TRUE the lowest signal-to-noise would be limited to -40 dB (if in.dB = TRUE). This would remove NA's that can be produced when noise segments have a higher amplitude than the signal itself. Default is TRUE.

bp — Numeric vector of length 2 giving the lower and upper limits of a frequency bandpass filter (in kHz). Default is NULL.

wl — A numeric vector of length 1 specifying the window length of the spectrogram for applying bandpass. Default is 10. Ignored if bp = NULL. It can also be set globally using the 'wl' option (see warbleR_options). Note that lower values will increase time resolution, which is more important for signal-to-noise ratio calculations.

Details

Signal-to-noise ratio (SNR) is a measure of the level of a desired signal compared to background noise. The function divides the mean amplitude of the signal by the mean amplitude of the background noise adjacent to the signal. A general margin to apply before and after the acoustic signal must be specified. Setting margins for individual signals that have been previously clipped from larger files may take some optimization, as for calls within a larger file that are irregularly separated. When margins overlap with another acoustic signal nearby, the signal-to-noise ratio (SNR) will be inaccurate. Any SNR less than or equal to one suggests background noise is equal to or overpowering the acoustic signal. snrspecs can be used to troubleshoot different noise margins.

Value

Data frame similar to autodetec output, but also includes a new variable with the signal-to-noise values.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) and Grace Smith Vidaurre

Source

https://en.wikipedia.org/wiki/Signal-to-noise_ratio

Examples

```
{
# First set temporary folder
# setwd(tempdir())

data(list = c("Phae.long1","selec.table"))
writeWave(Phae.long1, "Phae.long1.wav") #save sound files

# specifying the correct margin is important
# use snrspecs to troubleshoot margins for sound files
sig2noise(selec.table[grep("Phae.long1", selec.table$sound.files), ], mar = 0.2)

# this smaller margin doesn't overlap neighboring signals
sig2noise(selec.table[grep("Phae.long1", selec.table$sound.files), ], mar = 0.1)
```

```
}
```

`sim.coor.sing` *Simulated coordinated singing events.*

Description

`sim.coor.sing` are selections of simulated interactive singing events.The simulated events use the mean and standard deviation of real lekking *Phaethornis longirostris* (Long-billed Hermit hummingbird) songs and intervals between songs (e.i gaps). Three events are simulated: overlapping signals (ovlp), alternating signals (altern) and non-synchronized signals (uncoor). Will be replaced by 'sim_coor_sing' in future package versions.

Usage

```
data(sim.coor.sing)
```

Format

sim.coor.sing Simulated coordinated singing events that overlap and do not overlap most of the time, for use with `coor.test`

`sim_coor_sing` *Simulated coordinated singing events.*

Description

`sim_coor_sing` same data set as 'sim.coor.sing'. Selections of simulated interactive singing events.The simulated events use the mean and standard deviation of real lekking *Phaethornis longirostris* (Long-billed

Hermit hummingbird) songs and intervals between songs (e.i gaps). Three events are simulated: overlapping signals (ovlp), alternating signals (altern) and non-synchronized signals (uncoor).

Usage

```
data(sim_coor_sing)
```

Format

sim_coor_sing Simulated coordinated singing events that overlap and do not overlap most of the time, for use with `coor.test`

sim_songs *Simulate animal vocalizations*

Description

sim_songs simulate animal vocalizations in a wave object under brownian motion frequency drift.

Usage

```
sim_songs(n = 1, durs = 0.2, harms = 3, amps = c(1, 0.5, 0.2), gaps = 0.1, freqs = 5,
samp.rate = 44.1, sig2 = 0.5, steps = 10, bgn = 0.5, seed = NULL, diff_fun = "GBM",
fin = 0.1, fout = 0.2, shape = "linear", selec_table = FALSE, file_name = NULL,
path = NULL)
```

Arguments

n	Number of song subunits (e.g. elements). Default is 1.
durs	Numeric vector with the duration of subunits in seconds. It should either be a single value (which would be used for all subunits) or a vector of length n.
harms	NUmeric vector of length 1 specifyin the number of harmonics to simulate. 1 indicates that only the fundamental frequency harmonic will be simulated.
amps	Numeric vector with the relative amplitude of each of the harmonics (including the fundamental frequency).
gaps	Nueric vector with the duration of gaps (silence between subunits) in seconds. It should either be a single value (which would be used for all subunits) or a vector of length n + 1.
freqs	Nueric vector with the initial frequency of the subunits (and ending frequency if diff_fun == "BB") in kHz. It should either be a single value (which would be used for all subunits) or a vector of length n.
samp.rate	Numerirc vector of length 1. Sets the sampling frequency of the wave object (in kHz). Default is 44.1.
sig2	Numeric vector of length 1 defining the sigma value of the brownian motion model. Higher values will produce faster frequency modulations. Ignored if diff_fun == "BB". Default is 0.1. Check the BB for more details.
steps	Numeric vector of length 1. Controls the mean number of segments in which each song subunit is split during the brownian motion process. If not all subunits have the same duration, longer units will be split in more steps (although the average duration subunit will have the predefined number of steps). Default is 10.
bgn	Numeric vector of length 1 indicating the background noise level. 0 means no additional noise will 1 means noise at the same amplitude than the song subunits. Default is 0.5.
seed	Numeric vector of length 1. This allows users to get the same results in different runs (using set.seed internally). Default is NULL.

diff_fun	Character vector of length 1 controlling the function used to simulate the brownian motion process of frequency drift across time. Only "BB" and "GBM" are accepted at this time.Check the BB for more details.
fin	Numeric vector of length 1 setting the proportion of the sub-unit to fade-in amplitude (value between 0 and 1). Default is 0.1. Note that 'fin' + 'fout' cannot be higher than 1.
fout	Numeric vector of length 1 setting the proportion of the sub-unit to fade-out amplitude (value between 0 and 1). Default is 0.2. Note that 'fin' + 'fout' cannot be higher than 1.
shape	Character string of length 1 controlling the shape of in and out amplitude fading of the song sub-units ('fin' and 'fout'). "linear" (default), "exp" (exponential), and "cos" (cosine) are currently allowed.
selec_table	Logical. If TRUE a data frame containing the start/end time, and bottom/top frequency of the sub-units is also returned and the wave object is saved as a ".wav" file in the working directory. Defaul is FALSE.
file_name	Character string for naming the ".wav" file. Ignored if 'selec_table' is FALSE. If not provided the date-time stamp will be used.
path	Character string containing the directory path where the sound files are located. Ignored if 'selec_table' is FALSE. If NULL (default) then the current working directory is used.

Details

This functions uses a brownian motion stochastic process to simulate animal vocalizations (i.e. frequency traces across time). Several song subunits (e.g. elements) can be simulated as well as the corresponding harmonics.

Value

A wave object containing the simulated songs. If 'selec_table' is TRUE the function saves the wave object as a '.wav' sound file in the working directory (or 'path') and returns a list including 1) a selection table with the start/end time, and bottom/top frequency of the sub-units and 2) the wave object.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

querxc for for downloading bird vocalizations from an online repository.

Examples

```
{
 # simulate a song with 3 elements and no harmonics
 sm_sng <- sim_songs(n = 3, harms = 1)
```

```
 # plot spectro
 seewave::spectro(sm_sng)

 # simulate a song with 5 elements and 2 extra harmonics
sm_sng2 <- sim_songs(n = 5, harms = 3)

 # plot spectro
 seewave::spectro(sm_sng2)
}
```

---

snrspecs *Spectrograms with background noise margins*

Description

snrspecs creates spectrograms to visualize margins over which background noise will be measured by sig2noise.

Usage

```
snrspecs(X, wl = 512, flim = c(0, 22), wn = "hanning", ovlp = 70,
inner.mar = c(5, 4, 4, 2), outer.mar = c(0, 0, 0, 0), picsize = 1,
res = 100, cexlab = 1, title = TRUE, before = FALSE, eq.dur = FALSE,
  propwidth= FALSE, xl = 1, osci = FALSE, gr = FALSE, sc = FALSE, mar = 0.2,
   snrmar = 0.1, it = "jpeg", parallel = 1, path = NULL, pb = TRUE)
```

Arguments

X	'selection_table', 'extended_selection_table' or data frame with results from manualoc or any data frame with columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end).
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
flim	A numeric vector of length 2 for the frequency limit in kHz of the spectrogram, as in spectro. Default is c(0, 22).
wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 70.
inner.mar	Numeric vector with 4 elements, default is c(5,4,4,2). Specifies number of lines in inner plot margins where axis labels fall, with form c(bottom, left, top, right). See par.
outer.mar	Numeric vector with 4 elements, default is c(0,0,0,0). Specifies number of lines in outer plot margins beyond axis labels, with form c(bottom, left, top, right). See par.

picsize	Numeric argument of length 1, controls relative size of spectrogram. Default is 1.
res	Numeric argument of length 1 that controls image resolution. Default is 100 (faster) although 300 - 400 is recommended for publication/ presentation quality.
cexlab	Numeric vector of length 1 specifying relative size of axis labels. See spectro.
title	Logical argument to add a title to individual spectrograms. Default is TRUE.
before	Logical. If TRUE noise is only measured right before the signal (instead of before and after). Default is FALSE.
eq.dur	Logical. Controls whether the noise segment that is measured has the same duration than the signal (if TRUE, default FALSE). If TRUE then 'snrmar' argument is ignored.
propwidth	Logical argument to scale the width of spectrogram proportionally to duration of the selected call. Default is FALSE.
xl	Numeric vector of length 1, a constant by which to scale spectrogram width if propwidth = TRUE. Default is 1.
osci	Logical argument to add an oscillogram underneath spectrogram, as in spectro. Default is FALSE.
gr	Logical argument to add grid to spectrogram. Default is FALSE.
sc	Logical argument to add amplitude scale to spectrogram, default is FALSE.
mar	Numeric vector of length 1. Specifies the margins adjacent to the start and end points of the selections to define spectrogram limits. Default is 0.2. If snrmar is larger than mar, then mar is set to be equal to snrmar.
snrmar	Numeric vector of length 1. Specifies the margins adjacent to the start and end points of the selections where noise will be measured. Default is 0.1.
it	A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.

Details

This function can be used to test different margins to facilitate accurate SNR measurements when using sig2noise down the line. Setting margins for individual calls that have been previously clipped from larger files may take some optimization, as for calls within a larger file that are irregularly separated. Setting inner.mar to c(4,4.5,2,1) and outer.mar to c(4,2,2,1) works well when picsize = 2 or 3. Title font size, inner.mar and outer.mar (from mar and oma in par) don't work well when osci or sc = TRUE, this may take some optimization by the user.

Value

Spectrograms per selection marked with margins where background noise will be measured.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) and Grace Smith Vidaurre

Source

https://en.wikipedia.org/wiki/Signal-to-noise_ratio

See Also

trackfreqs for creating spectrograms to visualize frequency measurements by specan, specreator for creating spectrograms after using manualoc

Other spectrogram creators: color.spectro, dfDTW, dfts, ffDTW, ffts, sp.en.ts, specreator, trackfreqs

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "selec.table"))
writeWave(Phae.long1, "Phae.long1.wav") #save sound.files
writeWave(Phae.long2, "Phae.long2.wav")

# make Phae.long1 and Phae.long2 spectrograms
# snrmar needs to be smaller before moving on to sig2noise()

snrspecs(selec.table, flim = c(0, 14), inner.mar = c(4,4.5,2,1), outer.mar = c(4,2,2,1),
picsize = 2, res = 300, cexlab = 2, mar = 0.2, snrmar = 0.1, it = "jpeg", wl = 300)

# make only Phae.long1 spectrograms
# snrmar now doesn't overlap neighboring signals

snrspecs(selec.table[grepl(c("Phae.long1"), selec.table$sound.files), ], flim = c(3, 14),
inner.mar = c(4,4.5,2,1), outer.mar = c(4,2,2,1), picsize = 2, res = 300, cexlab = 2,
mar = 0.2, snrmar = 0.01, wl = 300)

#check this folder!!
getwd()

## End(Not run)
```

---

song_param *Measure acoustic parameters at the song level*

Description

song_param measures average or extreme values of acoustic parameters of elements in a song

Usage

```
song_param(X = NULL, weight = NULL, song_colm = "song",
mean_indx = NULL, min_indx = NULL, max_indx = NULL, sd = FALSE,
parallel = 1, pb = TRUE, na.rm = FALSE)
```

Arguments

X	'selection_table', 'extended_selection_table' (created 'by.song') or data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "selec": number of the selections, 3) "start": start time of selections, 4) "end": end time of selections. The ouptut of manualoc or autodetec can be used as the input data frame. Other data frames can be used as input, but must have at least the 4 columns mentioned above.
weight	Character vector defining 1 or more numeric vectors to weight average measurements (i.e. song parameters). Default is NULL. Names of numeric columns in 'X' can also be used. See weighted.mean. for more details. To use unweighted average set 'weight' to NULL.
song_colm	Character string with the column name containing song labels. Note that the function assumes that song labels are not repeated within a sound file.
mean_indx	Numeric vector with the index of the columns that will be averaged. If NULL the mean of all numeric columns in 'X' is returned.
min_indx	Numeric vector with the index of the columns for which the minimum value is needed. Default is NULL.
max_indx	Numeric vector with the index of the columns for which the maximum value is needed. Default is NULL. If NULL the mean of all numeric columns in 'X' is returned.
sd	Logical value indicating whether standard deviaton is also returned for variables in which averages are reported. Default is FALSE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
pb	Logical argument to control progress bar and messages. Default is TRUE.
na.rm	Logical value indicating whether 'NA' values should be ignored for calculations.

Details

The function removes silence segments (i.e. segments with very low amplitude values) from wave files.

Value

A data frame similar to the input 'X' data frame, containing the mean values for numeric acoustic parameters. Parameters to average can be defined with 'mean_indx' (otherwhise all numeric parameters are used). Parameters can be weighted by other parameters in the data set (e.g. duration, frequency range). Note that the functions works by default on songs, but can be used at other hierarchical levels (e.g. syllables, singing bouts). This function assumes that song labels are not repeated within a sound file.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

fixwavs, autodetec,

Examples

```
{
# Set temporary working directory

# get warbleR sound file examples
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

# add a 'song' column
selec.table$song <- rep(1:4, each = 3)[1:11]

# measure acoustic parameters
sp <- specan(selec.table, bp = c(1, 11), 300, fast = TRUE)

# add song data
sp <- merge(sp, selec.table, by = c("sound.files", "selec"))

# caculate song-level parameters for all numeric parameters
song_param(X = sp, song_colm = "song", parallel = 1, pb = TRUE)

# caculate song-level parameters selecting parameters with mean_indx
song_param(X = sp, song_colm = "song",mean_indx = 5:10, parallel = 1, pb = TRUE)

# caculate song-level parameters for selecting parameters with mean_indx, max_indx
# and min_indx and weighted by duration
song_param(X = sp, weight = "duration", song_colm = "song",
mean_indx = 5:6, min_indx = 7:8, max_indx = 9:10, parallel = 1, pb = TRUE)

# with two weights
song_param(X = sp, weight = c("duration", "dfrange"), song_colm = "song",
mean_indx = 5:9, parallel = 1, pb = TRUE)

# with two weights no progress bar
song_param(X = sp, weight = c("duration", "dfrange"), song_colm = "song",
mean_indx = 5:9, parallel = 1, pb = FALSE)
}
```

sp.en.ts *Extract the spectral entropy across signals as a time series*

Description

sp.en.ts spectral entropy across signals as a time series. of signals selected by manualoc or autodetec.

Usage

```
sp.en.ts(X, wl = 512, length.out = 20, wn = "hanning", ovlp = 70, bp = NULL,
  threshold = 15, img = TRUE, parallel = 1, path = NULL, img.suffix = "sp.en.ts",
   pb = TRUE, clip.edges = FALSE, leglab = "sp.en.ts", sp.en.range = c(2, 10), ...)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame containing columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). The ouptut of manualoc or autodetec can be used as the input data frame.
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512. Note that this is particularly important for measuring spectral entropy. Low values (~100) generate a very detail contour of the variation in spectral entropy that is probably not useful for assesing signal similarity.
length.out	A character vector of length 1 giving the number of measurements of spectral entropy desired (the length of the time series).
wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 70.
bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz). Default is NULL.
threshold	amplitude threshold (%) for dominant frequency detection. Default is 15.
img	Logical argument. If FALSE, image files are not produced. Default is TRUE.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located.
img.suffix	A character vector of length 1 with a sufix (label) to add at the end of the names of image files.
pb	Logical argument to control progress bar. Default is TRUE.
clip.edges	Logical argument to control whether edges (start or end of signal) in which amplitude values above the threshold were not detected will be removed. If TRUE this edges will be excluded and signal contour will be calculated on the remainging values. Default is FALSE.

leglab	A character vector of length 1 or 2 containing the label(s) of the frequency contour legend in the output image.
sp.en.range	Numeric vector of length 2. Range of frequency in which to display the entropy values on the spectrogram (when img = TRUE). Default is c(2, 10). Negative values can be used in order to stretch more the range.
...	Additional arguments to be passed to trackfreqs for customizing graphical output.

Details

This function spectral entropy across signals as a time series. The function uses the approx function to interpolate values between spectral entropy measures (calculated with csh). If there are no frequencies above the amplitude theshold at the begining or end of the signals then NAs will be generated. On the other hand, if there are no frequenciesabove the amplitude theshold in between signal segments in which amplitude was detected then the values of this adjacent segments will be interpolated to fill out the missing values (e.g. no NAs in between detected amplitude segments). Missing values at the start of end can be removed with "clip.edges".

Value

A data frame with the dominant frequency values measured across the signals. If img is TRUE it also produces image files with the spectrograms of the signals listed in the input data frame showing the location of the dominant frequencies (see trackfreqs description for more details).

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

specreator for creating spectrograms from selections, snrspecs for creating spectrograms to optimize noise margins used in sig2noise

Other spectrogram creators: color.spectro, dfDTW, dfts, ffDTW, ffts, snrspecs, specreator, trackfreqs

Examples

```
{
# set the temp directory
# setwd(tempdir())

#load data
data(list = c("Phae.long1", "Phae.long2",  "Phae.long3",  "Phae.long4","selec.table"))
writeWave(Phae.long2, "Phae.long2.wav") #save sound files
writeWave(Phae.long1, "Phae.long1.wav")
writeWave(Phae.long3, "Phae.long3.wav") #save sound files
writeWave(Phae.long4, "Phae.long4.wav")

# without clip edges
sp.en.ts(X = selec.table, threshold = 10, bp = NULL, clip.edges = FALSE, length.out = 10,
```

```
 type = "b", sp.en.range = c(-25, 10))

# with clip edges and length.out 10
sp.en.ts(X = selec.table, threshold = 10, bp = c(2, 12), clip.edges = TRUE, length.out = 10)

}
```

---

specan *Measure acoustic parameters in batches of sound files*

Description

specan measures acoustic parameters on acoustic signals for which the start and end times are provided.

Usage

```
specan(X, bp = c(0,22), wl = 512, wl.freq = NULL, threshold = 15,
 parallel = 1, fast = TRUE, path = NULL, pb = TRUE, ovlp = 50, ff.method = "seewave",
wn = "hanning", fsmooth = 0.1)
```

Arguments

X	'selection_table', 'extended_selection_table' or data frame with the following columns: 1) "sound.files": name of the .wav files, 2) "sel": number of the selections, 3) "start": start time of selections, 4) "end": end time of selections. The ouptut of manualoc or autodetec can be used as the input data frame.
bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz) or "frange" to indicate that values in bottom.freq and top.freq columns will be used as bandpass limits. Default is c(0, 22).Lower limit of bandpass is not applied to fundamental frequencies.
wl	A numeric vector of length 1 specifying the spectrogram window length. Default is 512. See 'wl.freq' for setting windows length independenlty in the frequency domain.
wl.freq	A numeric vector of length 1 specifying the window length of the spectrogram for measurements on the frecuency spectrum. Default is 512. Higher values would provide more accurate measurements. Note that this allows to increase measurement precision independently in the time and frequency domain. If NULL (default) then the 'wl' value is used.
threshold	amplitude threshold (%) for fundamental frequency and dominant frequency detection. Default is 15.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
fast	Logical. If TRUE (default) then the peakf acoustic parameter (see below) is not computed, which substantially increases performance (~9 times faster).

path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar and messages. Default is TRUE.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, used for fundamental frequency (using fund or FF) and dominant frequency (using dfreq). Default is 50.
ff.method	Character. Selects the method used to calculate the fundamental frequency. Either 'tuneR' (using FF) or 'seewave' (using fund). Default is 'seewave'. Use trackfreqs to decide which method works the best. 'tuneR' performs faster (and seems to be more accurate) than 'seewave'.
wn	Character vector of length 1 specifying window name. Default is hanning'. See function ftwindow for more options.
fsmooth	A numeric vector of length 1 to smooth the frequency spectrum with a mean sliding window (in kHz) used for mean peak frequency detection. This help to average amplitude "hills" to minimize the effect of amplitude modulation. Default is 0.1.

Details

The ouptut of manualoc or autodetec can be used directly without any additional modification. The function measures 29 acoustic parameters (if fast = TRUE) on each selection in the data frame. Most parameters are produced internally by specprop, fpeaks, fund, and dfreq from the package seewave and FF from the package tuneR. NAs are produced for fundamental and dominant frequency measures when there are no amplitude values above the threshold.

Value

Data frame with 'sound.files' and 'selec' as in the input data frame, plus the following acoustic parameters:

- duration: length of signal (in s)
- meanfreq: mean frequency. Weighted average of frequency by amplitude (in kHz)
- sd: standard deviation of frequency weighted by amplitude
- freq.median: median frequency. The frequency at which the signal is divided in two frequency intervals of equal energy (in kHz)
- freq.Q25: first quartile frequency. The frequency at which the signal is divided in two frequency intervals of 25% and 75% energy respectively (in kHz)
- freq.Q75: third quartile frequency. The frequency at which the signal is divided in two frequency intervals of 75% and 25% energy respectively (in kHz)
- freq.IQR: interquartile frequency range. Frequency range between 'freq.Q25' and 'freq.Q75' (in kHz)
- time.median: median time. The time at which the signal is divided in two time intervals of equal energy (in s)
- time.Q25: first quartile time. The time at which the signal is divided in two time intervals of 25% and 75% energy respectively (in s). See acoustat

- time.Q75: third quartile time. The time at which the signal is divided in two time intervals of 75% and 25% energy respectively (in s). See acoustat
- time.IQR: interquartile time range. Time range between 'time.Q25' and 'time.Q75' (in s). See acoustat
- skew: skewness. Asymmetry of the spectrum (see note in specprop description)
- kurt: kurtosis. Peakedness of the spectrum (see note in specprop description)
- sp.ent: spectral entropy. Energy distribution of the frequency spectrum. Pure tone ~ 0; noisy ~ 1. See sh
- time.ent: time entropy. Energy distribution on the time envelope. Pure tone ~ 0; noisy ~ 1. See th
- entropy: spectral entropy. Product of time and spectral entropy sp.ent * time.ent. See H
- sfm: spectral flatness. Similar to sp.ent (Pure tone ~ 0; noisy ~ 1). See sfm
- meanfun: average of fundamental frequency measured across the acoustic signal
- minfun: minimum fundamental frequency measured across the acoustic signal
- maxfun: maximum fundamental frequency measured across the acoustic signal
- meandom: average of dominant frequency measured across the acoustic signal
- mindom: minimum of dominant frequency measured across the acoustic signal
- maxdom: maximum of dominant frequency measured across the acoustic signal
- dfrange: range of dominant frequency measured across the acoustic signal
- modindx: modulation index. Calculated as the cumulative absolute difference between adjacent measurements of dominant frequencies divided by the dominant frequency range. 1 means the signals is not modulated.
- startdom: dominant frequency measurement at the start of the signal
- enddom: dominant frequency measurement at the end of the signal
- dfslope: slope of the change in dominant frequency through time ((enddom-startdom)/duration). Units are kHz/s.
- peakf: peak frequency. Frequency with highest energy (only generated if fast = FALSE)
- meanpeakf: Mean peak frequency. Frequency with highest energy from the mean spectrum (see meanspec). Typically more consistent than peakf.
- bottom.freq: low frequency. Low limit of frequency range (only generated if frange.detec = TRUE)
- top.freq: high frequency. High limit of frequency range (only generated if frange.detec = TRUE)

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) and Grace Smith Vidaurre

Examples

```
{
# First set temporary folder
# setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
```

```
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

a <- specan(X = selec.table, bp = c(0, 22))

# using a diferent threshold
a <- specan(X = selec.table, bp = c(0, 22), threshold = 20)
# View(a)

}
```

---

specreator *Spectrograms of selected signals*

Description

specreator creates spectrograms of signals selected by manualoc or autodetec.

Usage

```
specreator(X, wl = 512, flim = c(0, 22), wn = "hanning", pal
  = reverse.gray.colors.2, ovlp = 70, inner.mar = c(5, 4, 4, 2), outer.mar =
  c(0, 0, 0, 0), picsize = 1, res = 100, cexlab = 1, title = TRUE,
  propwidth = FALSE, xl = 1, osci = FALSE, gr = FALSE,  sc = FALSE, line = TRUE,
  col = adjustcolor("#E37222", 0.6), lty = 3, mar = 0.05, it = "jpeg",
  parallel = 1, path = NULL, pb = TRUE, fast.spec = FALSE, by.song = NULL,
  sel.labels = "selec", ...)
```

Arguments

X	'selection_table', 'extended_selection_table' or data frame containing columns for sound file name (sound.files), selection number (selec), and start and end time of signals (start and end). Low and high frequency columns are optional. The ouptut of manualoc or autodetec can be used as the input data frame. If using an 'extended_selection_table' the sound files are not required (see selection_table).
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
flim	A numeric vector of length 2 for the frequency limit (in kHz) of the spectrogram, as in spectro. Default is c(0, 22).
wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
pal	A color palette function to be used to assign colors in the plot, as in spectro. Default is reverse.gray.colors.2.
ovlp	Numeric vector of length 1 specifying the percent overlap between two consecutive windows, as in spectro. Default is 70.

inner.mar	Numeric vector with 4 elements, default is c(5,4,4,2). Specifies number of lines in inner plot margins where axis labels fall, with form c(bottom, left, top, right). See par.
outer.mar	Numeric vector with 4 elements, default is c(0,0,0,0). Specifies number of lines in outer plot margins beyond axis labels, with form c(bottom, left, top, right). See par.
picsize	Numeric argument of length 1. Controls relative size of spectrogram. Default is 1. Ignored when propwidth is TRUE.
res	Numeric argument of length 1. Controls image resolution. Default is 100 (faster) although 300 - 400 is recommended for publication/ presentation quality.
cexlab	Numeric vector of length 1 specifying the relative size of axis labels. See spectro.
title	Logical argument to add a title to individual spectrograms. Default is TRUE.
propwidth	Logical argument to scale the width of spectrogram proportionally to duration of the selection. Default is FALSE.
xl	Numeric vector of length 1. A constant by which to scale spectrogram width if propwidth = TRUE. Default is 1.
osci	Logical argument to add an oscillogram underneath spectrogram, as in spectro. Default is FALSE.
gr	Logical argument to add grid to spectrogram. Default is FALSE.
sc	Logical argument to add amplitude scale to spectrogram, default is FALSE.
line	Logical argument to add red lines at start and end times of selection (or box if bottom.freq and top.freq columns are provided). Default is TRUE.
col	Color of 'line'. Default is 'adjustcolor("red2", alpha.f = 0.7)'.'
lty	Type of 'line' as in par. Default is 1.
mar	Numeric vector of length 1. Specifies the margins adjacent to the start and end points of selections, dealineating spectrogram limits. Default is 0.05.
it	A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE.
fast.spec	Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.

by.song	Character string with the column name containing song labels. If provide a single spectrogram containinig all elements for each song will be produce. Note that the function assumes that song labels are not repeated within a sound file. If NULL (default), spectrograms are produced for single selections.
sel.labels	Character string with the name of the column for selection labeling. Ignored if 'by.song' is NULL. Default is 'selec'. Set to NULL to remove labels.
...	Additional arguments to be passed to the internal spectrogram creating function for customizing graphical output. The function is a modified version of spectro, so it takes the same arguments.

Details

This function provides access to batch process of (a modified version of) the spectro function from the 'seewave' package. The function creates spectrograms for visualization of vocalizations. Setting inner.mar to c(4,4.5,2,1) and outer.mar to c(4,2,2,1) works well when picsize = 2 or 3. Title font size, inner.mar and outer.mar (from mar and oma) don't work well when osci or sc = TRUE, this may take some optimization by the user. Setting 'fast' argument to TRUE significantly increases speed, although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels.

Value

Image files containing spectrograms of the signals listed in the input data frame.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) and Grace Smith Vidaurre

See Also

trackfreqs for creating spectrograms to visualize frequency measurements by specan, snrspecs for creating spectrograms to optimize noise margins used in sig2noise

Other spectrogram creators: color.spectro, dfDTW, dfts, ffDTW, ffts, snrspecs, sp.en.ts, trackfreqs

Examples

```
{
# First set empty folder
# setwd(tempdir())

# load and save data
data(list = c("Phae.long1", "Phae.long2","selec.table"))
writeWave(Phae.long1, "Phae.long1.wav") #save sound files
writeWave(Phae.long2, "Phae.long2.wav")

# make spectrograms
specreator(X = selec.table, flim = c(0, 11), res = 300, mar = 0.05, wl = 300)

# check this folder
```

```
getwd()
}
```

---

spec_param *Plot a mosaic of spectrograms with varying display parameters*

Description

spec_param plots a mosaic of spectrograms with varying display parameters to facilitate selection of display parameters

Usage

```
spec_param(X, length.out = 5, ovlp = 90, wl = c(100, 1000), wn = "hanning",
 collev.min = -40, pal = "reverse.gray.colors.2", path = NULL, rm.axes = TRUE, ...)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame with a single row and columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). Default is NULL.
length.out	Numeric vector of length 1 controling the number of sublevels of the numeric arguments for which a range has been provided. Ranges are allowed for 'ovlp', 'wl', and 'collev.min' arguments.
ovlp	Numeric vector of length 1 or 2 specifying % of overlap (or lower/upper values the desired range) between two consecutive windows, as in spectro. Default is 90.
wl	A numeric vector of length 1 or 2 specifying the window length (length 1) or the lower and upper range limits of the desired window length range (length 2) for creating spectrograms. Default is c(100, 1000).
wn	Character vector specifying the window function names to be used. Several names can be provided. See ftwindow for name options. Default is "hanning". If "all", then all window functions available are used.
collev.min	A (negative) numeric vector of length 1 or 2. Determines the first argument to use in 'collevels' for the internal spectrogram creating function. This replaces the first element in the 'collevels' as in spectro. Note that 'collevels' is not available in this function spec_param.
pal	Color palette function for spectrogram. Default is "reverse.gray.colors.2". Several palettes can be provided in a character vector. Note that, contrary to other warbleR and seewave functions, the palette most be provided as character string rather than as a function. See spectro for more palettes.
path	Character string containing the directory path where the sound file are located.
rm.axes	Logical. If TRUE frequency and time axes are excluded. Default is TRUE.
...	Additional arguments to be passed to catalog function for customizing graphical output. Check out catalog for more details.

Details

This functions aims to simplify the selection of spectrogram parameters. The function plots, for a single selection, a mosaic of spectrograms with varying display parameters. For numeric arguments the upper and lower limits of a range can be provided. The following arguments accept can have varying values:

- `wl`: Windows length (numeric range)
- `ovlp`: Overlap (numeric range)
- `collev.min`: Minimum value of the color levels (numeric range)
- `wn`: window function names (character)
- `pal`: palette (character)

Outputs are similar to those of `catalog`. The output image files can be put together in a single pdf file with `catalog2pdf`. We recommend using low resolution (~60-100) and smaller dimensions (width & height < 10) if aiming to generate pdfs (otherwise pdfs could be pretty big).

Value

Image files with spectrograms of whole sound files in the working directory. Multiple pages can be returned, depending on the length of each sound file.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

https://marce10.github.io/2017/03/17/Creating_song_catalogs.html, https://marce10.github.io/2017/07/31/Updates_on_catalog_function.html & `catalog2pdf`

Examples

```
## Not run:
# Set temporary working directory
# setwd(tempdir())
# save sound file examples
data(list = c("Phae.long1", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")

# variable collevels
spec_param(X = selec.table, wl = 164, ovlp = c(90), wn = c("flattop"),
length.out = 16, nrow = 4, ncol = 4, width = 20, height = 11.3, rm.axes = TRUE,
cex = 1, box = F, collev.min = c(-20, -150))

# variable overlap and wn
spec_param(X = selec.table, wl = 164, ovlp = c(50, 90),
wn = c("hanning", "hamming", "rectangle", "bartlett", "blackman", "flattop"),
length.out = 7, nrow = 6, ncol = 7, width = 20, height = 11.3, rm.axes = TRUE,
cex = 1, box = F)
```

```
# variable wl and wn
spec_param(X = selec.table, wl = c(100, 1000), ovlp = c(50, 90), wn = "all",
length.out = 5, nrow = 10, ncol = 14, width = 20, height = 11.3, rm.axes = TRUE,
cex = 0.7)

# variable wl, collev.min and wn
spec_param(X = selec.table, wl = c(100, 1000), ovlp = 90,
wn = c("hanning", "hamming", "rectangle"), collev.min = c(-110, -25),
length.out = 3, nrow = 10, ncol = 14, width = 20, height = 11.3, rm.axes = TRUE,
 cex = 0.7)

 # variable wl, wn and pal
 spec_param(X = selec.table, wl = c(100, 1000), ovlp = 90,
 wn = c("hanning", "hamming", "rectangle"),
 pal = c("reverse.gray.colors.2", "reverse.topo.colors",
 "reverse.terrain.colors", "reverse.cm.colors"),
 length.out = 4, nrow = 5, ncol = 10, width = 20, height = 11.3,
  rm.axes = TRUE, cex = 0.7, lab.mar = 2)

  # wl, wn and pal
  spec_param(X = selec.table, wl = c(100, 1000), ovlp = 90,
   wn = c("hanning", "hamming", "rectangle"),
  pal = c("reverse.gray.colors.2", "reverse.topo.colors",
  "reverse.terrain.colors", "reverse.cm.colors"),
  length.out = 4, nrow = 5, ncol = 10, width = 20, height = 11.3, rm.axes = TRUE,
   cex = 0.7, group.tag = "wn",  spec.mar = 0.4, lab.mar = 0.8, box = FALSE,
   tag.pal = list(reverse.cm.colors))

check this floder
getwd()

## End(Not run)
```

---

trackfreqs *Spectrograms with frequency measurements*

Description

trackfreqs creates spectrograms to visualize dominant and fundametal frequency measurements (contours) of signals selected by manualoc or autodetec.

Usage

```
trackfreqs(X, wl = 512, wl.freq = 512, flim = c(0, 22), wn = "hanning", pal =
  reverse.gray.colors.2, ovlp = 70, inner.mar = c(5, 4, 4, 2), outer.mar =
  c(0, 0, 0, 0), picsize = 1, res = 100, cexlab = 1, title = TRUE, propwidth = FALSE,
  xl = 1, osci = FALSE, gr = FALSE, sc = FALSE, bp = c(0, 22), cex = c(0.6, 1),
  threshold = 15, threshold.time = NULL, threshold.freq = NULL, contour = "both",
  col = c("#E37222B3", "#07889BB3"), pch = c(21, 24),  mar = 0.05, lpos = "topright",
  it = "jpeg", parallel = 1, path = NULL, img.suffix = NULL, custom.contour = NULL,
```

```
pb = TRUE, type = "p", leglab = c("Ffreq", "Dfreq"), col.alpha = 0.6, line = TRUE,
  fast.spec = FALSE, ff.method = "seewave", frange.detec = FALSE,
  fsmooth = 0.1, widths = c(2, 1), freq.continuity = NULL, clip.edges = 2, ...)
```

Arguments

X	object of class 'selection_table', 'extended_selection_table' or data frame containing columns for sound file name (sound.files), selection number (selec), and start and end time of signal (start and end). The ouptut of manualoc or autodetec can be used as the input data frame.
wl	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
wl.freq	A numeric vector of length 1 specifying the window length of the spectrogram for measurements on the frecuency spectrum. Default is 512. Higher values would provide more accurate measurements.
flim	A numeric vector of length 2 for the frequency limit of the spectrogram (in kHz), as in spectro. Default is c(0, 22).
wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options.
pal	A color palette function to be used to assign colors in the plot, as in spectro. Default is reverse.gray.colors.2.
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 70.
inner.mar	Numeric vector with 4 elements, default is c(5,4,4,2). Specifies number of lines in inner plot margins where axis labels fall, with form c(bottom, left, top, right). See par.
outer.mar	Numeric vector with 4 elements, default is c(0,0,0,0). Specifies number of lines in outer plot margins beyond axis labels, with form c(bottom, left, top, right). See par.
picsize	Numeric argument of length 1. Controls relative size of spectrogram. Default is 1.
res	Numeric argument of length 1. Controls image resolution. Default is 100 (faster) although 300 - 400 is recommended for publication/ presentation quality.
cexlab	Numeric vector of length 1 specifying the relative size of axis labels. See spectro.
title	Logical argument to add a title to individual spectrograms. Default is TRUE.
propwidth	Logical argument to scale the width of spectrogram proportionally to duration of the selected call. Default is FALSE.
xl	Numeric vector of length 1. A constant by which to scale spectrogram width. Default is 1.
osci	Logical argument to add an oscillogram underneath spectrogram, as in spectro. Default is FALSE.
gr	Logical argument to add grid to spectrogram. Default is FALSE.
sc	Logical argument to add amplitude scale to spectrogram, default is FALSE.

bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz) or "frange" to indicate that values in bottom.freq and top.freq columns will be used as bandpass limits. Default is c(0, 22).
cex	Numeric vector of length 2, specifies relative size of points plotted for frequency measurements and legend font/points, respectively. See spectro.
threshold	amplitude threshold (%) for fundamental and dominant frequency detection as well as frequency range from the spectrum (see 'frange.detec'). Default is 15. WILL BE DEPRECATED. Use 'threshold.time' and 'threshold.time' instead.
threshold.time	amplitude threshold (%) for the time domain. Use for fundamental and dominant frequency detection. If NULL (default) then the 'threshold' value is used.
threshold.freq	amplitude threshold (%) for the frequency domain. Use for frequency range detection from the spectrum (see 'frange.detec'). If NULL (default) then the 'threshold' value is used.
contour	Character vector, one of "df", "ff" or "both", specifying whether the dominant or fundamental frequencies or both should be plotted. Default is "both".
col	Vector of length 1 or 2 specifying colors of points plotted to mark fundamental and dominant frequency measurements respetively (if both are plotted). Default is c("#E37222B3","#07889BB3").
pch	Numeric vector of length 1 or 2 specifying plotting characters for the frequency measurements. Default is c(21, 24).
mar	Numeric vector of length 1. Specifies the margins adjacent to the selections to set spectrogram limits. Default is 0.05.
lpos	Character vector of length 1 or numeric vector of length 2, specifiying position of legend. If the former, any keyword accepted by xy.coords can be used (see below). If the latter, the first value will be the x coordinate and the second value the y coordinate for the legend's position. Default is "topright".
it	A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
img.suffix	A character vector of length 1 with a suffix (label) to add at the end of the names of image files. Default is NULL.
custom.contour	A data frame with frequency contours for exactly the same sound files and selection as in X. The frequency values are assumed to be equally spaced in between the start and end of the signal. The first 2 colums of the data frame should contain the 'sound.files' and 'selec' columns and should be identical to the corresponding columns in X (same order).
pb	Logical argument to control progress bar. Default is TRUE.
type	A character vector of length 1 indicating the type of frequency contour plot to be drawn. Possible types are "p" for points, "l" for lines and "b" for both.
leglab	A character vector of length 1 or 2 containing the label(s) of the frequency contour legend in the output image.

col.alpha — A numeric vector of length 1 within [0,1] indicating how transparent the lines/points should be.

line — Logical argument to add red lines (or box if bottom.freq and top.freq columns are provided) at start and end times of selection. Default is TRUE.

fast.spec — Logical. If TRUE then image function is used internally to create spectrograms, which substantially increases performance (much faster), although some options become unavailable, as collevels, and sc (amplitude scale). This option is indicated for signals with high background noise levels. Palette colors gray.1, gray.2, gray.3, topo.1 and rainbow.1 (which should be imported from the package monitoR) seem to work better with 'fast' spectograms. Palette colors gray.1, gray.2, gray.3 offer decreasing darkness levels.

ff.method — Character. Selects the method used to calculate the fundamental frequency. Either 'tuneR' (using FF) or 'seewave' (using fund). Default is 'seewave'. 'tuneR' performs faster (and seems to be more accurate) than 'seewave'.

frange.detec — Logical. Controls whether frequency range of signal is automatically detected using the frange.detec function. If so, the range is used as the bandpass filter (overwriting 'bp' argument). Default is FALSE.

fsmooth — A numeric vector of length 1 to smooth the frequency spectrum with a mean sliding window (in kHz) used for frequency range detection (when frange.detec = TRUE). This help to average amplitude "hills" to minimize the effect of amplitude modulation. Default is 0.1.

widths — Numeric vector of length 2 to control the relative widths of the spectro (first element) and spectrum (second element, (when frange.detec = TRUE)).

freq.continuity — Numeric vector of length 1 to control whether dominant frequency detections outliers(i.e that differ from the frequency of the detections right before and after) would be removed. Should be given in kHz. Default is NULL.

clip.edges — Integer vector of length 1 to control if how many 'frequency-wise discontinuous' detection would be remove at the start and end of signals (see 'freq.continuity' argument). Default is 2. Ignored if freq.continuity = NULL.

... — Additional arguments to be passed to the internal spectrogram creating function for customizing graphical output. The function is a modified version of spectro, so it takes the same arguments.

Details

This function provides visualization of frequency measurements as the ones made by specan, dfts, ffts, dfDTW and ffDTW. Frequency measures can be made by the function or input by the user (see 'custom.contour' argument). If frange = TRUE the function uses frange.detec to detect the frequency range. In this case the graphical output includes a frequency spectrum showing the detection threshold.

Value

Spectrograms of the signals listed in the input data frame showing the location of the dominant and fundamental frequencies.

Author(s)

Grace Smith Vidaurre and Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

specreator for creating spectrograms from selections, snrspecs for creating spectrograms to optimize noise margins used in sig2noise

Other spectrogram creators: color.spectro, dfDTW, dfts, ffDTW, ffts, snrspecs, sp.en.ts, specreator

Examples

```
{
#Set temporary working directory
# setwd(tempdir())

#load data
data("Cryp.soui")
writeWave(Cryp.soui, "Cryp.soui.wav") #save sound files

#autodetec location of signals
ad <- autodetec(threshold = 6, bp = c(1, 3), mindur = 1.2,
maxdur = 3, img = FALSE, ssmooth = 600, wl = 300, flist = "Cryp.soui.wav")

#track dominant frequency graphs with freq reange detection
trackfreqs(X = ad[!is.na(ad$start),], flim = c(0, 5), ovlp = 90, it = "tiff",
bp = c(1, 3), contour = "df", wl = 300, frange = TRUE)

#using users frequency data (custom.contour argument)
#first get contours using dfts
df <- dfts(X = ad[!is.na(ad$start),], flim = c(0, 5), ovlp = 90, img = FALSE,
bp = c(1, 3),  wl = 300)

# now input the dfts output into trackfreqs
trackfreqs(X = ad[!is.na(ad$start),], custom.contour = df ,flim = c(0, 5), ovlp = 90, it = "tiff")

# Check this folder
getwd()

#track both frequencies
trackfreqs(X = ad[!is.na(ad$start),], flim = c(0, 5), ovlp = 90, it = "tiff",
bp = c(1, 3), contour = "both", wl = 300)

}
```

---

track_harm *Track harmonic frequency contour*

Description

track_harm tracks the frequency contour of the dominant harmonic.

Usage

```
track_harm(wave, f, wl = 512, wn = "hanning", ovlp = 0, fftw = FALSE, at = NULL,
tlim = NULL, threshold = 10, bandpass = NULL, clip = NULL, plot = TRUE,
xlab = "Times (s)", ylab = "Frequency (kHz)",  ylim = c(0, f/2000),
adjust.wl = FALSE, dfrq = FALSE, ...)
```

Arguments

wave	A 'wave' object produced by readWave or similar functions.
f	Sampling frequency of the wave object (in Hz). Does not need to be specified if embedded in wave.
wl	A numeric vector of length 1 specifying the window length for the FFT, default is 512.
wn	Character vector of length 1 specifying window name. Default is "hanning". See function ftwindow for more options. This is used for calculating the frequency spectrum (using meanspec) and producing the spectrogram (using spectro, if plot = TRUE).
ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive time windows, as in spectro. Default is 0.
fftw	if TRUE calls the function FFT of the library fftw. See Notes of the spectro function. Default is FALSE.
at	Time position where the harmonic frequency contour has to be computed (in seconds). Default is NULL.
tlim	time range in which to measure frequency contours. Default is NULL (which means it will measure across the entire wave object).
threshold	Amplitude threshold (%) for dominant frequency and detection. Default is 10.
bandpass	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz).
clip	A numeric value to select dominant frequency values according to their amplitude in reference to a maximal value of 1 for the whole signal (has to be >0 & < 1).
plot	Logical, if TRUE plots the dominant frequency against time. Default is TRUE.
xlab	Label of the time axis.
ylab	Label of the frequency axis.
ylim	A numeric vector of length 2 for the frequency limit of the spectrogram (in kHz), as in spectro. Default is c(0, f/2000).
adjust.wl	Logical. If TRUE the 'wl' is reset to be equal at the number of samples in a selections if the samples are less than 'wl'. Default is FALSE.
dfrq	Logical. If TRUE seewave's dfreq is used instead. Default is FALSE.
...	Additional arguments to be passed to the plotting function.

Details

This is a modified version of seewave's dfreq function that allows to track the frequency contour of a dominant harmonic even when the highest amplitude jumps between harmonics. The arguments and default values of the original dfreq function have been kept unchanged to facilitate switching between the 2 functions.

Author(s)

Jerome Sueur, modified by Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

trackfreqs for tracking frequencies iteratively on selections tables.

Examples

```
{
#Set temporary working directory
# setwd(tempdir())

#load data

# Check this folder
getwd()

#track both frequencies

}
```

---

try_na *Wrapper for "try" function*

Description

try_na wrapper for try function that returns an NA if an error is found.

Usage

```
try_na(expr, silent = TRUE, outFile)
```

Arguments

expr	An R expression to try.
silent	Logical to control whether the report of error messages is suppressed. Default is TRUE.
outFile	A connection, or a character string naming the file to print to (via cat(*, file = outFile)); used only if silent is false, as by default.

Details

This is a wrapper on that returns an 'NA' if any error occurs during the evaluation of a expression. See `try` for details.

Value

Returns an 'NA' if any error occurs during the evaluation of a expression. If not, it will return the result of the evaluation.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

Examples

```
{
# try a function that does not exists to produce an error
try_na(crazy78(12))

# try a real function (no error)
try_na(mean(1:5))
}
```

---

warbleR *warbleR: A package to streamline bioacoustic analysis*

Description

warbleR is a package designed to streamline analysis of animal acoustic signals in R. This package allows users to collect open-access avian vocalizations data or input their own data into a workflow that facilitates spectrographic visualization and measurement of acoustic parameters. warbleR makes fundamental sound analysis tools from the R package seewave, as well as new tools not yet offered in the R environment, readily available for batch process analysis. The functions facilitate searching and downloading avian vocalizations from 'Xeno-Canto' http://www.xeno-canto.org/, creating maps of 'Xeno-Canto' recordings, converting .mp3 files to .wav files, checking .wav files, automatically detecting acoustic signals, selecting them manually, printing spectrograms of whole recordings or individual signals, measuring signal to noise ratio, cross-correlation and performing acoustic measurements.

The warbleR package offers three overarching categories of functions:

- Obtaining avian vocalization data
- Sound file management
- Streamlined (bio)acoustic analysis in R

Details

License: GPL (>= 2)

Obtaining avian vocalization data

querxc: Download recordings and metadata from 'Xeno-Canto'

sim_songs: Simulate animal vocalizations

Managing sound files

selection_table: Create 'selection_table' class objects

mp32wav: Convert several .mp3 files in working directory to .wav format

checksels: Check whether selections can be read by subsequent functions

checkwavs: Check whether .wav files can be read by subsequent functions and the minimum windows length ("wl" argument) that can be used

fixwavs: Fix .wav files to allow importing them into R

wavdur: Determine the duration of sound files

cut_sels: Cut selections from a selection table into individual sound files

rm_sil: Remove silence segments from wave files

consolidate: Consolidate sound files into a single folder

selection_table: Create double-checked and self-contained selection tables

fix_extended_selection_table: Fix attributes of extended selection tables

Exploring/analyzing signal structure

autodetec: Automatically detect start and end of acoustic signals

manualoc: Interactive spectrographic view to measure start and end of acoustic signals

autodetec: Automatic detection of acoustic signals based on ampltiude

seltailor: Interactive view of spectrograms to tailor start and end of selections

sig2noise: Measure signal-to-noise ratio across multiple files

trackfreqs: Create spectrograms to visualize frequency measurements

filtersels: Filter selection data frames based on filtered image files

frange: Detect frequency range iteratively from signals in a selection table

frange.detec: Detect frequency range in a Wave object

specan: Measure acoustic parameters on selected acoustic signals

xcorr: Pairwise cross-correlation of multiple signals

dfts: Extract the dominant frequency values across the signal as a time series

ffts: Extract the fundamental frequency values across the signal as a time series

sp.en.ts: Extract the spectral entropy values across the signal as a time series

dfDTW: Calculate acoustic dissimilarity using dynamic time warping on dominant frequency contours

ffDTW: Calculate acoustic dissimilarity using dynamic time warping on fundamental frequency contours

compare.methods: Produce graphs to visually assess performance of acoustic distance measurements

coor.test: Assess statistical significance of singing coordination

ovlp_sels: Find selections that overlap in time within a given sound file

track_harm: Track harmonic frequency contour

Graphical outputs

xcmaps: Create maps to visualize the geographic spread of 'Xeno-Canto' recordings

catalog: Produce a vocalization catalog with spectrograms in and array with several rows and columns

catalog2pdf: Combine catalog images to single pdf files

coor.graph: Creat graphs of coordinated singing

color.spectro: Highlight spectrogram regions

xcorr.graph: Pairwise cross-correlation of multiple signals

lspec: Produce spectrograms of whole recordings split into multiple rows

lspec2pdf: Combine lspec images to single pdf files

specreator: Create spectrograms of manualoc selections

snrspecs: Create spectrograms to visualize margins over which noise will be measured by sig2noise

Author(s)

Marcelo Araya-Salas & Grace Smith Vidaurre

Maintainer: Marcelo Araya-Salas (<araya-salas@cornell.edu>)

---

warbleR_options *Setting warbleR options*

Description

warbleR_options sets global parameters for warbleR functions

Usage

```
warbleR_options(...)
```

Arguments

...	Arguments in 'parameter = value" form, or a list of tagged values. The tags (i.e. parameters) must come from the list of parameters described below.

Details

The function aims to simplify the use of parameters that apply to many warbleR functions (i.e. global parameters) by setting a default value that will be used to any function in downstream analyses. Tags that are set with warbleR_options will be used iby the functions that share those arguments. However, if an argument is set within a function call it will overwrite the values set by warbleR_options. Hence, the functions remain 'flexible' as their parameters can also be modified 'on the fly'. The following tags are available:

- `bp`: Numeric vector of length 2 giving the lower and upper limits of a frequency bandpass filter (in kHz).
- `collevels`: A numeric vector of length 3. Specifies levels to partition the amplitude range of the spectrogram (in dB) as in `spectro`. The more levels the higher the resolution of the spectrogram. THe lower the first value the darker the spectrograms.
- `flim`: A numeric vector of length 2 for the frequency limit in kHz of the spectrogram, as in `spectro`.
- `it`: A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted.
- `osci`: Logical argument to add an oscillogram underneath spectrogram, as in `spectro`.
- `pal`: A color palette function to be used to assign colors in the plot, as in `spectro`.
- `parallel`: Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used in iterative functions.
- `pb`: Logical argument to control whether progress bar is used.
- `res`: Numeric argument of length 1. Controls image resolution in all image creating functions.
- `wav.path`: Character string containing the directory path where the sound files are located. Used as 'path' in all functions in which sound files are read.
- `wl`: A numeric vector of length 1 specifying the window length for creating spectrogram (either for plotting or for measuring spectrogram parameters).
- `wn`: Character vector of length 1 specifying the window name for creating spectrogram (either for plotting or for measuring spectrogram parameters). See function `ftwindow` for options.

Value

When parameters are set by warbleR_options, their former values are returned in an invisible named list. Such a list can be passed as an argument to pboptions to restore the parameter values. If the function is called with no arguments the current option values are printed.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

Examples

```
{
# First set temporary folder
# setwd(tempdir())
```

```
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4", "selec.table"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")
writeWave(Phae.long4,"Phae.long4.wav")

# sig2noise with progress bar (by default is TRUE)
a <- sig2noise(X = selec.table, mar = 0.1)

# set progress bar to FALSE with warbleR_options
warbleR_options(pb = FALSE)

# sig2noise without progress bar
a <- sig2noise(X = selec.table, mar = 0.1)

# sig2noise with progress bar by setting it within the function call (overwritting options)
a <- sig2noise(X = selec.table, pb = TRUE, mar = 0.1)

# sig2noise without progress bar using warbleR_options setting again
a <- sig2noise(X = selec.table, mar = 0.1)
}
```

---

wavdur *Measure the duration of sound files*

Description

wavdur measures the duration of sound files in '.wav' format

Usage

```
wavdur(files = NULL, path = NULL)
```

Arguments

files	Character vector with the names of the sound files to be measured. The sound files should be in the working directory or in the directory provided in 'path'.
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.

Details

This function returns the duration (in seconds) of sound files.

Value

A data frame with the duration (in seconds) of the sound files.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

Examples

```
{
# Set temporary working directory
# setwd(tempdir())

data(list = c("Phae.long1", "Phae.long2", "Phae.long3"))
writeWave(Phae.long1,"Phae.long1.wav")
writeWave(Phae.long2,"Phae.long2.wav")
writeWave(Phae.long3,"Phae.long3.wav")

wavdur()
}
```

---

xcmaps *Maps of 'Xeno-Canto' recordings by species*

Description

xcmaps creates maps to visualize the geographic spread of 'Xeno-Canto' recordings.

Usage

```
xcmaps(X, img = TRUE, it = "jpeg", res = 100, labels = F)
```

Arguments

X	Data frame output from querxc.
img	A logical argument specifying whether an image file of each species map should be returned, default is TRUE.
it	A character vector of length 1 giving the image type to be used. Currently only "tiff" and "jpeg" are admitted. Default is "jpeg".
res	Numeric argument of length 1. Controls image resolution. Default is 100 (faster) although 300 - 400 is recommended for publication/ presentation quality.
labels	A logical argument defining whether dots depicting recording locations are labeled. If TRUE then the Recording_ID is used as label.

Details

This function creates maps for visualizing the geographic spread of recordings from the open-access online repository 'Xeno-Canto' (http://www.xeno-canto.org/). The function takes the output of querxc as input. Maps can be displayed in the graphic devide or saved as images in the working directory.

Value

A map of 'Xeno-Canto' recordings per species (image file), or a faceted plot of species map(s) in the active graphic device.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>) and Grace Smith Vidaurre

Examples

```
## Not run:
# search in xeno-canto
X <- querxc("Phaethornis anthophilus", download = FALSE)

#create image in R graphic device
xcmaps(X, img = FALSE)

#or save it as a file in the working directory
xcmaps(X)


## End(Not run)
```

---

xcorr *Spectrogram cross-correlation*

Description

`xcorr` estimates the similarity of two spectrograms by means of cross-correlation

Usage

```
xcorr(X, wl =512, frange= NULL, ovlp=90, dens=0.9, bp= NULL, wn='hanning',
cor.method = "pearson", parallel = 1, path = NULL, pb = TRUE, na.rm = FALSE,
 dfrange = FALSE, cor.mat = TRUE)
```

Arguments

`X`	'selection_table', 'extended_selection_table' or data frame containing columns for sound files (sound.files), selection number (selec), and start and end time of signal (start and end).
`wl`	A numeric vector of length 1 specifying the window length of the spectrogram, default is 512.
`frange`	A numeric vector of length 2 setting the upper and lower frequency limits (in kHz) in which to compare the signals. Must be provided. The `dfts` function can be used to determine this parameter if `dfrange = TRUE`. This method is more adequate for pure tone signals. Default is `NULL`.

ovlp	Numeric vector of length 1 specifying % of overlap between two consecutive windows, as in spectro. Default is 90. High values of ovlp slow down the function but produce more accurate results.
dens	Numeric vector of length 1 specifying the approximate density of points in which to sample amplitude. See makeTemplate. Deafult is 0.9.
bp	A numeric vector of length 2 for the lower and upper limits of a frequency bandpass filter (in kHz) in which to detect dominant frequency. Only applied when frange is NULL. Default is NULL.
wn	A character vector of length 1 specifying the window name as in ftwindow.
cor.method	A character vector of length 1 specifying the correlation method as in cor.
parallel	Numeric. Controls whether parallel computing is applied. It specifies the number of cores to be used. Default is 1 (i.e. no parallel computing).
path	Character string containing the directory path where the sound files are located. If NULL (default) then the current working directory is used.
pb	Logical argument to control progress bar. Default is TRUE. Note that progress bar is not accurate as the number of pairwise comparisons decreases on each iteration. The first iteration runs n-1 comparisons while the last one only 1 (n = nrow(X)).
na.rm	Logical. If TRUE all NAs produced when pairwise cross-correlations failed are removed from the results. This means that all selections with at least 1 cross-correlation that failed are excluded.
dfrange	Logical. If TRUE the dfts function can is used to determine the frequency range in which to compare signals.
cor.mat	Logical. If TRUE only the correlation matrix is returned. Default is TRUE.

Details

This function calculates the pairwise similarity of multiple signals by means of spectrogram cross-correlation. This method "slides" one spectrogram over the other calculating a correlation of the amplitude values at each step. The function runs pairwise cross-correlations on several signals and returns a list including the correlation statistic for each "sliding" step as well as the maximum (peak) correlation for each pairwise comparison. To accomplish this the margins of the signals are expanded by half the duration of the signal both before and after the provided time coordinates. The correlation matrix could have NA's if some of the pairwise correlation did not work (common when sound files have been modified by band-pass filters). This function is a modified version of the corMatch and makeTemplate from the awesome R package 'monitoR'.

Value

If corr.mat is TRUE the function returns a matrix with the maximum (peak) correlation for each pairwise comparison. Otherwise it will return a list that includes 1) a data frame with the correlation statistic for each "sliding" step, 2) a matrix with the maximum correlation for each pairwise comparison, and 3) the frequency range.

Author(s)

Marcelo Araya-Salas <araya-salas@cornell.edu>)

Source

H. Khanna, S.L.L. Gaunt & D.A. McCallum (1997). Digital spectrographic cross-correlation: tests of sensitivity. Bioacoustics 7(3): 209-234

See Also

xcorr.graph

Examples

```
{
#First set temporary working directory
# setwd(tempdir())

#load data
data(list = c("Phae.long1", "Phae.long2", "Phae.long3", "Phae.long4","selec.table"))
writeWave(Phae.long1, "Phae.long1.wav") #save sound files
writeWave(Phae.long2, "Phae.long2.wav")
writeWave(Phae.long3, "Phae.long3.wav")
writeWave(Phae.long4, "Phae.long4.wav")

xcor <- xcorr(X = selec.table, wl = 300, frange = c(2, 9), ovlp = 90,
dens = 1, wn = 'hanning', cor.method = "pearson")

}
```

---

xcorr.graph *Pairwise plots of spectrogram cross-correlation scores*

Description

xcorr.graph generates pairwise plots showing the spectrogram cross-correlation scores against the time sliding.

Usage

```
xcorr.graph(X, cex.cor = 1, cex.lab = 1, cex.axis.lab = 1, rel.cex = FALSE, labs = NULL)
```

Arguments

X	Output from xcorr function.
cex.cor	A numeric vector of length 1 giving the amount by which correlation scores (in the upper triangle of the multipannel plot) should be magnified. Default is 1.
cex.lab	A numeric vector of length 1 giving the amount by which signal selection labels (in diagonal of the multipannel plot) should be magnified. Default is 1.
cex.axis.lab	A numeric vector of length 1 giving the amount by which the axis labels should be magnified. Default is 1.

rel.cex	Logical. Controls whether the size of the correlation scores (in the upper triangle of the multipannel plot) should be relative to the correlation score.
labs	Alternative selection labels. If not provided the combined name of sound files and selection numbers are used as labels. Default is FALSE.

Details

This function generates pairwise plots of the spectrogram cross-correlation scores by sliding step. The function takes the output of xcorr (when cor.mat is FALSE in xcorr) as input. The colors of the lines in the lower triangle of the plot matrix represent the strenght of the similarity between the two signals. The x axis shows the time difference between the two signals for each sliding step (0 means perfectly centered signals). Note that large number of signals may not display well in the default graphic device. In such cases saving the plot as and image file is adviced.

Author(s)

Marcelo Araya-Salas (<araya-salas@cornell.edu>)

See Also

xcorr

Examples

```
{
#load data
#First set temporary working directory]
# setwd(tempdir())

#load data
data(list = c("Phae.long1", "Phae.long2", "selec.table"))
writeWave(Phae.long1, "Phae.long1.wav") #save sound files
writeWave(Phae.long2, "Phae.long2.wav")

 #run cross correlation first
 xcor<-xcorr(X = selec.table[1:5,], wl =300, frange= c(2, 9), ovlp=90, dens=0.8,
 wn= "hanning", cor.method = "pearson", cor.mat = FALSE)

 #plot pairwise scores
  #xcorr.graph(X = xcor, cex.cor = 2, cex.lab = 1, rel.cex = FALSE)
}
```

Index