Package ‘dynaSpec’

June 22, 2020

Type Package

Title Dynamic Spectrogram Visualizations

Version 1.0.0

Date 2020-06-14

Description A set of tools to generate dynamic spectrogram visualizations in video format.

License GPL (>= 2)

Imports pbapply, utils, grDevices, stats, seewave, tuneR, grid, png,
NatureSounds, ggplot2

Depends R (>= 3.2.1)

LazyData TRUE

SystemRequirements ffmpeg

URL `https://github.com/maRce10/dynaSpec`

BugReports `https://github.com/maRce10/dynaSpec/issues`

NeedsCompilation no

Suggests parallel, viridis, imager

RoxygenNote 7.1.0

Repository CRAN

Language en-US

Author Marcelo Araya-Salas [aut, cre]
(<https://orcid.org/0000-0003-3594-619X>)

Maintainer Marcelo Araya-Salas <`marcelo.araya@ucr.ac.cr`>

Date/Publication 2020-06-22 13:50:02 UTC

R topics documented:

canyon_wren *Acoustic recording of a* Catherpes mexicanus *(canyon wren) song.*

Description

Acoustic recording of a *Catherpes mexicanus* (canyon wren) song.

Usage

```
data(canyon_wren)
```

Format

One Wave object:

canyon_wren *Catherpes mexicanus* recording

dynaSpec *dynaSpec: Dynamic spectrogram visualization*

Description

A set of tools to generate dynamic spectrogram visualizations in video format.

Details

License: GPL (>= 2)

Author(s)

Marcelo Araya-Salas

Maintainer: Marcelo Araya-Salas (<`marcelo.araya@ucr.ac.cr`>)

scrolling_spectro *Create scrolling dynamic spectrograms*

Description

scrolling_spectro create videos of single row spectrograms scrolling from right to left sync'ed with sound.

Usage

```
scrolling_spectro(wave, file.name = "scroll.spectro.mp4", hop.size = 11.6, wl = NULL,
ovlp = 70, flim = NULL, pal = seewave::reverse.gray.colors.1, speed = 1, fps = 50,
t.display = 1.5, fix.time = TRUE, res = 70,
width = 700, height = 400, parallel = 1, pb = TRUE,
 play = TRUE, loop = 1, lcol = "#07889B99",
 lty = 2, lwd = 2, axis.type = "standard", buffer = 1,
 ggspectro = FALSE, lower.spectro = TRUE, height.prop = c(5, 1), derivative = FALSE,
 osc = FALSE, colwave = "black", colbg = "white",
 spectro.call = NULL, annotation.call = NULL, ...)
```

Arguments

`wave` object of class 'Wave'.

`file.name` Character string with the name of the output video file. Must include the .mp4 extension. Default is 'scroll.spectro.mp4'.

`hop.size` A numeric vector of length 1 specifying the time window duration (in ms). Default is 11.6 ms, which is equivalent to 512 wl for a 44.1 kHz sampling rate. Ignored if 'wl' is supplied.

`wl` A numeric vector of length 1 specifying the window length of the spectrogram, default is NULL. If supplied, 'hop.size' is ignored.

`ovlp` Numeric vector of length 1 specifying the percent overlap between two consecutive windows, as in spectro. Default is 70.

`flim` A numeric vector of length 2 specifying limits in the frequency axis (in kHz). Default is NULL (which means from 0 to Nyquist frequency).

`pal` Character string with the color palette to be used. Default is 'reverse.gray.colors.1'.

`speed` Numeric vector of length 1 indicating the speed at which the sound file will be reproduced (default is 1, normal speed). Values < 1 (but higher than 0) slow down while values > 1 speed up. Note that changes in speed are achieved by modifying the number of frames per second in the output video. Hence, you may want to adjust 'fps' if video quality is considerably affected.

`fps` Numeric vector of length 1 specifying the number of frames per second.

`t.display` Numeric vector of length 1 specifying the time range displayed in the spectrogram.

`fix.time` Logical argument to control if the time axis moves along with the spectrogram or remains fixed. Default is `TRUE` (fixed).

`res` Numeric vector of length 1 specifying the resolution of the image files (see `png`).

`width` Numeric vector of length 1 specifying width of the video frame in pixels (see `png`). Default is 700.

`height` Numeric vector of length 1 specifying height of the video frame in pixels (see `png`). Default is 400.

`parallel` Numeric vector of length 1. Controls whether parallel computing is applied by specifying the number of cores to be used. Default is 1 (i.e. no parallel computing).

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

`play` Logical argument to control if the video is played after generated. Default is `TRUE`.

`loop` Logical argument to control if the video is formatted to be played in a loop (i.e. if ends at the start of the clip).

`lcol` Character string with the color to be used for the vertical line at which sounds are played. Default is `"#07889B99"`.

`lty` Character string to control the type of the line at which sounds are played. Line types can either be specified as an integer (0=blank, 1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as one of the character strings "blank", "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash", where "blank" uses 'invisible lines' (i.e., does not draw them).Default is 2.

`lwd` Character string to control the width of the line at which sounds are played. Default is 2.

`axis.type` Character string to control the style of spectrogram axes. Currently there are 3 options: #'

- `standard`: Both Y and X axes are printed as in the default `spectro` view.
- `minimal`: Single lines are used to denote the range defined by 1 s and 1 kHz for the X and Y axes respectively.
- `none`: No axis is printed (also removes ticks, tick labels, and axis labels).

`buffer` Numeric vector of length 1 (> 0) specifying the time to delay the start of the spectrogram scrolling (in seconds). Default is 1. Not available when loop is > 1.

`ggspectro` Logical argument to control if a ggspectro (`ggspectro`) is used instead. Note that there is much less control on display parameters when `ggpsectro = TRUE`. Default is `FALSE`.

`lower.spectro` Logical argument to control if a spectrogram of the full wave object is plotted at the bottom of the graph. Default is `TRUE`.

`height.prop` Numeric vector of length 2 to control the relative height of the scrolling and lower spectro, respectively. Default is `c(5,1)`. Ignored if `lower.spectro = FALSE`.

`derivative` Logical argument to control if spectral derivative is used instead of spectrogram (as in Sound Analysis Pro, see `deriche`). Default is `FALSE`.

`osc` Logical argument to control if the oscillogram is plotted at the bottom of the spectrogram. Default is `FALSE`. Note that 'osc' and 'lower.spectro' are mutually exclusive.

`colwave` Character string to control the color of the oscillogram. Default is 'black'.

`colbg` Character string to control the background color. Default is 'white'.

`spectro.call` A call from a spectrogram creating function (i.e. `spectro`, `color_spectro`) generated by the function `call`. This call will replace the internal spectrogram creating call. Default is `NULL`.

`annotation.call`

A call from `text` generated by the function `call`. The call should also include the argmuents 'start' and 'end' to indicate the time at which the labels are displayed (in s).'fading' is optional and allows fade-in and fade-out effects on labels (in s as well). The position ('x' and 'y' arguments) should be between 0 and 1: `x = 0,y = 0` corresponds to the bottom left and `x = 1,y = 1` corresponds to the top right position.

`...` Additional arguments to be passed to `spectro` for customizing spectrograms. Note that 'scale' cannot be included.

Details

The function creates videos (mp4 format) of single row spectrograms scrolling from right to left. The audio is sync'ed with the spectrograms.

Value

A video file in mp4 format in the working directory with the scrolling spectrogram.

Author(s)

Marcelo Araya-Salas (<marcelo.araya@ucr.ac.cr>)

References

Araya-Salas M. (2020). *dynaSpec: dynamic spectrogram visualizations in R*. R package version 1.0.0.

See Also

`spectro`

Examples

```
## Not run:
# load example data
data(list = c("Phae.long1"))

# run function
scrolling_spectro(wave = Phae.long1, wl = 300, ovlp = 90,
fps = 50, t.display = 1.5, collevels = seq(-40, 0, 5),
 pal = reverse.heat.colors, grid = FALSE, flim = c(1, 10),
 res = 120)

## End(Not run)
```

Index