Package ‘voice’

April 19, 2023

Type Package

Title Tools for Voice Analysis, Speaker Recognition and Mood Inference

Version 0.4.21

Date 2023-04-20

Maintainer Zabala Filipe J. <filipezabala@gmail.com>

URL https://github.com/filipezabala/voice

BugReports https://github.com/filipezabala/voice/issues

Description Tools for voice analysis, speaker recognition and mood inference. Gathers 'R' and 'Python' tools to solve problems concerning voice and audio in general.

Depends R (>= 4.0.0)

Imports dplyr, R.utils, reticulate, seewave, tibble, tidyselect, tuneR, wrassp, zoo

Suggests gm, knitr

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

VignetteBuilder knitr

Language en-US

NeedsCompilation no

Author Zabala Filipe J. [cre, aut]

Repository CRAN

Date/Publication 2023-04-19 11:00:02 UTC
\textbf{R topics documented:}

\begin{itemize}
  \item \texttt{audio_time} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 2
  \item \texttt{diarize} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 3
  \item \texttt{duration} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 4
  \item \texttt{enrich_rttm} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 5
  \item \texttt{expand_model} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 6
  \item \texttt{extract_features} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 7
  \item \texttt{feat_summary} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 10
  \item \texttt{get_bit} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 12
  \item \texttt{get_dur} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 13
  \item \texttt{get_left} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 14
  \item \texttt{get_right} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 14
  \item \texttt{get samp.rate} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 15
  \item \texttt{get tbeg} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 16
  \item \texttt{get tdur} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 16
  \item \texttt{interp} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 17
  \item \texttt{interp df} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 18
  \item \texttt{interp mc} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 20
  \item \texttt{is mono} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 21
  \item \texttt{mozilla id path} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 22
  \item \texttt{notes} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 22
  \item \texttt{notes freq} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 23
  \item \texttt{read rttm} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 24
  \item \texttt{rm0} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 25
  \item \texttt{smooth df} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 26
  \item \texttt{splitw} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 27
  \item \texttt{tag} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 29
  \item \texttt{write list} \hspace{1cm} \texttt{\ldots} \hspace{1cm} 31
\end{itemize}

\textbf{Index} \hspace{1cm} 33

\begin{tabular}{ll}
\hline
audio\_time & \textit{Returns the total time of audio files in seconds} \\
\hline
\end{tabular}

\textbf{Description}

Returns the total time of audio files in seconds

\textbf{Usage}

\texttt{audio\_time(x, filesRange = NULL, recursive = FALSE)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{x} \hspace{1cm} Either a WAV file or a directory containing WAV files.
  \item \texttt{filesRange} \hspace{1cm} The desired range of directory files (default: NULL, i.e., all files).
  \item \texttt{recursive} \hspace{1cm} Logical. Should the listing recursively into directories? (default: FALSE) Used by \texttt{base::list.files}.
\end{itemize}
**Value**

A tibble containing file name <chr> and audio time <dbl> in seconds.

**Examples**

library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
    pattern <- glob2rx('*.wav'), full.names = TRUE)

# Tibble containing file name and audio time
(at <- voice::audio_time(unique(dirname(path2wav))))
str(at)

---

**diarize**

Who spoke when?

**Description**

Diarization of WAV audios.

**Usage**

\[
diarize(
  fromWav,
  toRttm = NULL,
  autoDir = FALSE,
  pycall = "~/miniconda3/envs/pyvoice38/bin/python3.8",
  token = NULL
)
\]

**Arguments**

- **fromWav**: Either a file or a directory containing WAV files.
- **toRttm**: A directory to write RTTM files. If the default toRttm = NULL is used, './voiceAudios/rttm' is created and used.
- **autoDir**: Logical. Must the directories tree be created? Default: FALSE. See 'Details'.
- **pycall**: Python call. See [https://github.com/filipezabala/voice](https://github.com/filipezabala/voice) for details.
- **token**: Access token needed to instantiate pretrained speaker diarization pipeline from pyannote.audio. #1. Visit [https://hf.co/pyannote/speaker-diarization](https://hf.co/pyannote/speaker-diarization) and accept user conditions. #2. Visit [https://hf.co/pyannote/segmentation](https://hf.co/pyannote/segmentation) and accept user conditions. #3. Visit [https://hf.co/settings/tokens](https://hf.co/settings/tokens) to create an access token. More details at [https://github.com/pyannote/pyannote-audio](https://github.com/pyannote/pyannote-audio).
### duration

**Details**

When `autoDir = TRUE`, the following directories are created: `../mp3`, `../rttm`, `../split` and `../musicxml`. Use `getwd()` to find the parent directory `../`.

**Value**

RTTM files in NIST standard. See `voice::read_rttm`.

**Examples**

```r
## Not run:
library(voice)

wavDir <- list.files(system.file('extdata', package = 'wrassp'),
                     pattern = glob2rx('*.wav'), full.names = TRUE)

voice::diarize(fromWav = unique(dirname(wavDir)),
               toRttm = tempdir(),
               token = NULL) # Must enter a token! See documentation.

(rttm <- dir(tempdir(), '[Rr][Tt][Tt][Mm]$', full.names = TRUE))

file.info(rttm)
## End(Not run)
```

---

### duration

**Description**

Duration of sequences

**Usage**

```
duration(x, windowShift = 5)
```

**Arguments**

- `x` A vector containing symbols and NA.
- `windowShift` Window shift to duration in ms (default: 5.0).

**Value**

A data frame with duration in number of lines/occurrences (`dur_line`), milliseconds (`dur_ms`) and proportional (`dur_prop`).
### Examples

```r
library(voice)
duration(letters)
duration(c('a','a','a', 'letters', 'z'))

nts <- c('NA', 'A3', 'A3', 'A3', 'A#3', 'B3', 'B3', 'C4', 'C4', 'C4', 'C4',
         'C4', 'C4', 'C#4', 'C#4', 'C4', 'C4', 'B3', 'B3', 'A#3', 'NA', 'NA', 'NA', 'NA', 'NA', 'NA', 'NA', 'NA',
         'C#4', 'C#4', 'C#4', 'B3', 'B3', 'A#3', 'A#3', 'A3', 'A3', 'G3', 'G#3', 'G3', 'F#3')
duration(nts)
```

### enrich_rttm

**Enrich RTTM files**

#### Description

Enrich Rich Transcription Time Marked (RTTM) files obtained from `voice::read_rttm`.

#### Usage

```r
enrich_rttm(listRttm, silence.gap = 0.5, as.tibble = TRUE)
```

#### Arguments

- **listRttm**: A list containing RTTM files.
- **silence.gap**: The silence gap (in seconds) between adjacent words in a keyword. Rows with `tdur <= silence.gap` are removed. (default: 0.5)
- **as.tibble**: Logical. Should it return a tibble?

#### Value

A list containing either data frames or tibbles obtained from standard RTTM files. See `voice::read_rttm`.

#### References


#### See Also

`voice::read_rttm`
Expand model

Example

```r
library(voice)
url0 <- 'https://raw.githubusercontent.com/filipezabala/voiceAudios/main/rttm/sherlock0.rttm'
destfile0 <- paste0(tempdir(), '/sherlock0.rttm')
download.file(url0, destfile = destfile0)
url1 <- 'https://raw.githubusercontent.com/filipezabala/voiceAudios/main/rttm/sherlock1.rttm'
destfile1 <- paste0(tempdir(), '/sherlock1.rttm')
download.file(url1, destfile = destfile1)
rttm <- voice::read_rttm(dirname(destfile0))
(er <- voice::enrich_rttm(rttm))
class(er)
lapply(er, class)
```

**Description**

Expand model given `y` and `x` variables.

**Usage**

```r
expand_model(y, x, k)
```

**Arguments**

- `y` The Y variable.
- `x` The X variables.
- `k` Number of additive components.

**Value**

A char vector containing the expanded models.

**Examples**

```r
library(voice)
expand_model('y', LETTERS[1:4], 1)
expand_model('y', LETTERS[1:4], 2)
expand_model('y', LETTERS[1:4], 3)
expand_model('y', LETTERS[1:4], 4)

# multiple models using apply functions
```
nx <- 10 # number of X variables to be used
definitive_var <- lapply(1:mx, expand_model, y = 'y', x = LETTERS[1:mx])
names(definitive_var) <- 1:mx
definitive_var
sum(sapply(definitive_var, length)) # total of definitive_var

extract_features

### Description
Extracts features from WAV audio files.

#### Usage
```r
evaluate_features(x, features = c("f0", "fmt", "rf", "rfp", "rcf", "rfc", "mfcc"),
                  filesRange = NULL,
                  sex = "u",
                  windowShift = 10,
                  numFormants = 8,
                  numcep = 12,
                  dcttype = c("t2", "t1", "t3", "t4"),
                  fbtype = c("mel", "htkmel", "fcmel", "bark"),
                  resolution = 40,
                  usecmp = FALSE,
                  mc.cores = 1,
                  full.names = TRUE,
                  recursive = FALSE,
                  check.mono = FALSE,
                  stereo2mono = FALSE,
                  overwrite = FALSE,
                  freq = 44100,
                  round.to = NULL,
                  verbose = FALSE,
                  pycall = "~/miniconda3/envs/pyvoice38/bin/python3.8"
)
```

#### Arguments
- **x**: A vector containing either files or directories of audio files in WAV format.
- **features**: Vector of features to be extracted. (Default: 'f0', 'fmt', 'rf', 'rcf', 'rfp', 'rfc', 'mfcc'). The 'fmt_praat' feature may take long time processing. The following features may contain a variable number of columns: 'cep', 'dft', 'css' and 'lps'.
filesRange The desired range of directory files (Default: NULL, i.e., all files). Should only be used when all the WAV files are in the same folder.

sex = <code> set sex specific parameters where <code> = ‘f’[emale], ‘m’[ale] or ‘u’[unknown] (Default: ‘u’). Used as ‘gender’ by wrassp::ksvF0, wrassp::forest and wrassp::mhsF0.

windowShift = <dur> set analysis window shift to <dur>ation in ms (Default: 5.0). Used by wrassp::ksvF0, wrassp::forest, wrassp::mhsF0, wrassp::zcrana, wrassp::rfcana, wrassp::acfana, wrassp::cepstrum, wrassp::dftSpectrum, wrassp::cssSpectrum and wrassp::lpsSpectrum.

numFormants = <num> number of formants (Default: 8). Used by wrassp::forest.

numcep  Number of Mel-frequency cepstral coefficients (cepstra) to return (Default: 12). Used by tuneR::melfcc.

dcttype Type of DCT used. ‘t1’ or ‘t2’, ‘t3’ for HTK ‘t4’ for feacalc (Default: ‘t2’). Used by tuneR::melfcc.


resolution = <freq> set FFT length to the smallest value which results in a frequency resolution of <freq> Hz or better (Default: 40.0). Used by wrassp::cssSpectrum, wrassp::dftSpectrum and wrassp::lpsSpectrum.


mc.cores Number of cores to be used in parallel processing. (Default: 1)

full.names Logical. If TRUE, the directory path is prepended to the file names to give a relative file path. If FALSE, the file names (rather than paths) are returned. (Default: TRUE) Used by base::list.files.

recursive Logical. Should the listing recursively into directories? (Default: FALSE) Used by base::list.files.

check.mono Logical. Check if the WAV file is mono. (Default: TRUE)

stereo2mono (Experimental) Logical. Should files be converted from stereo to mono? (Default: TRUE)

overwrite (Experimental) Logical. Should converted files be overwritten? If not, the file gets the suffix _mono. (Default: FALSE)

freq Frequency in Hz to write the converted files when stereo2mono=TRUE. (Default: 44100)

round.to Number of decimal places to round to. (Default: NULL)

verbose Logical. Should the running status be showed? (Default: FALSE)

pycall Python call. See https://github.com/filipezabala/voice for details.

Details

The feature 'df' corresponds to 'formant dispersion' (df2:df8) by Fitch (1997), 'pf' to formant position (pf1:pf8) by Puts, Apicella & Cárdena (2011), 'rf' to 'formant removal' (rf1:rf8) by Zabala (2023), 'rcf' to 'formant cumulated removal' (rcf2:rcf8) by Zabala (2023) and 'rpf' to 'formant position removal' (rpf2:rpf8) by Zabala (2023).
Value

A Media data frame containing the selected features.

References


Examples

```r
library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
                      pattern = glob2rx('*.wav'), full.names = TRUE)

# minimal usage
M1 <- extract_features(path2wav)
M2 <- extract_features(dirname(path2wav))
identical(M1,M2)
table(basename(M1$wav_path))

# limiting filesRange
M3 <- extract_features(path2wav, filesRange = 3:6)
table(basename(M3$wav_path))
```

---

### feat_summary

<table>
<thead>
<tr>
<th>Features summary</th>
</tr>
</thead>
</table>

**Description**

Returns summary measures of `voice::extract_features`.

**Usage**

```r
feat_summary(
  x,
  groupBy = "wav_path",
  wavPath = unique(x$wav_path),
  wavPathName = "wav_path",
  features = "f0",
  filesRange = NULL,
  sex = "u",
  windowShift = 10,
  numFormants = 8,
  numcep = 12,
  dcttype = c("t2", "t1", "t3", "t4"),
  fctype = c("mel", "htkmel", "fcme", "bark"),
  resolution = 40,
  usecmp = FALSE,
  mc.cores = 1,
  full.names = TRUE,
  recursive = FALSE,
  check.mono = FALSE,
  stereo2mono = FALSE,
  overwrite = FALSE,
  freq = 44100,
)```
round.to = 4,
verbose = FALSE
)

Arguments

x     An Extended data frame to be tagged with media information.
groupBy     A variable to group the summary measures. The argument must be a character
            vector. (Default: groupBy = 'wav_path').
wavPath     A vector containing the path(s) to WAV files. May be both as dirname or
            basename formats.
wavPathName     A string containing the WAV path name. (Default: wavPathName = 'wav_path').
ftrues     Vector of features to be extracted. (Default: 'f0').
filesRange     The desired range of directory files (default: NULL, i.e., all files). Should only be
            used when all the WAV files are in the same folder.
sex     = <code> set sex specific parameters where <code> = 'f'[emale], 'm'[ale] or
            'u'[known] (Default: 'u'). Used as 'gender' by wrassp::ksvF0, wrassp::forest
            and wrassp::mhsF0.
windowShift     = <dur> set analysis window shift to <dur>ation in ms (Default: 5.0). Used by
            wrassp::ksvF0, wrassp::forest, wrassp::mhsF0, wrassp::zcrana, wrassp::rfcana,
            wrassp::acfana, wrassp::cepstrum, wrassp::dftSpectrum, wrassp::cssSpectrum
            and wrassp::lpsSpectrum.
numFormants     = <num> <num>ber of formants (Default: 8). Used by wrassp::forest.
numcep     Number of Mel-frequency cepstral coefficients (cepstra) to return (Default: 12).
            Used by tuneR::melfcc.
dcttype     Type of DCT used. 't1' or 't2', 't3' for HTK 't4' for feacalc (Default: 't2').
            Used by tuneR::melfcc.
fbtype     Auditory frequency scale to use: 'mel', 'bark', 'htkmel', 'fcme1' (Default: 'mel').
            Used by tuneR::melfcc.
resolution     = <freq> set FFT length to the smallest value which results in a frequency reso-
            lution of <freq> Hz or better (Default: 40.0). Used by wrassp::cssSpectrum,
            wrassp::dftSpectrum and wrassp::lpsSpectrum.
usecmp     Logical. Apply equal-loudness weighting and cube-root compression (PLP instead
            of LPC) (Default: FALSE). Used by tuneR::melfcc.
mc.cores     Number of cores to be used in parallel processing. (Default: 1)
full.names     Logical. If TRUE, the directory path is prepended to the file names to give a relative
            file path. If FALSE, the file names (rather than paths) are returned. (Default:
            TRUE). Used by base::list.files.
recursive     Logical. Should the listing recursively into directories? (Default: FALSE) Used
            by base::list.files.
check.mono     Logical. Check if the WAV file is mono. (Default: TRUE)
stereo2mono     (Experimental) Logical. Should files be converted from stereo to mono? (De-
            fault: TRUE)
overwrite (Experimental) Logical. Should converted files be overwritten? If not, the file gets the suffix _mono. (Default: FALSE)

freq Frequency in Hz to write the converted files when stereo2mono=TRUE. (Default: 44100)

round.to Number of decimal places to round to. (Default: NULL)

verbose Logical. Should the running status be showed? (Default: FALSE)

Details

filesRange should only be used when all the WAV files are in the same folder.

Value

A tibble data frame containing summarized numeric columns using (1) mean, (2) standard deviation, (3) variation coefficient, (4) median, (5) interquartile range and (6) median absolute deviation.

Examples

```r
library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'), pattern = glob2rx('*.wav'), full.names = TRUE)

# creating Extended synthetic data
E <- dplyr::tibble(subject_id = c(1,1,1,2,2,2,3,3,3), wav_path = path2wav)

# minimal usage
feat_summary(E)

# canonical data
feat_summary(E, groupBy = 'subject_id')
```

---

get_bit

Get bit rate

Description

Get bit rate from WAV file.

Usage

```r
get_bit(x)
```

Arguments

- **x**: Wave object from ‘tuneR::readWave’. 

**get_dur**

**Value**

Integer indicating the bit rate from a WAV file.

**Examples**

```r
library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
                      pattern = glob2rx('*\.wav'), full.names = TRUE)

rw <- tuneR::readWave(path2wav[1])
voice::get_bit(rw)

rwl <- lapply(path2wav, tuneR::readWave)
sapply(rwl, voice::get_bit)
```

---

**get_dur**

_Time duration_

**Description**

Get time duration from WAV file.

**Usage**

```r
get_dur(x)
```

**Arguments**

- `x` Wave object from `tuneR::readWave`.

**Value**

Numeric indicating the time duration in seconds from a WAV file.

**Examples**

```r
library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
                      pattern = glob2rx('*\.wav'), full.names = TRUE)

rw <- tuneR::readWave(path2wav[1])
voice::get_dur(rw)

rwl <- lapply(path2wav, tuneR::readWave)
sapply(rwl, voice::get_dur)
```
get_left

**Get left channel**

**Description**
Get left channel from WAV file.

**Usage**

```r
get_left(x)
```

**Arguments**

- `x` Wave object from `tuneR::readWave`.

**Value**
Numeric vector indicating the left channel from a WAV file.

**Examples**

```r
library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
                       pattern = glob2rx('*.wav'), full.names = TRUE)

rw <- tuneR::readWave(path2wav[1])
l <- voice::get_left(rw)
head(l)
length(l)
```

---

get_right

**Get right channel**

**Description**
Get right channel from WAV file.

**Usage**

```r
get_right(x)
```

**Arguments**

- `x` Wave object from `tuneR::readWave`.
get_samp.rate

Value
Numeric vector indicating the right channel from a WAV file.

Examples
library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
                      pattern <- glob2rx('*.wav'), full.names = TRUE)

rw <- tuneR::readWave(path2wav[1])
r <- voice::get_right(rw)
head(r)
length(r)

________________________________________________________________________________

get_samp.rate                  Get sample rate

Description
Get sample rate from WAV file.

Usage
get_samp.rate(x)

Arguments
x Wave object from `tuneR::readWave`.

Value
Integer indicating the sample rate from a WAV file.

Examples
library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
                      pattern <- glob2rx('*.wav'), full.names = TRUE)

rw <- tuneR::readWave(path2wav[1])
voice::get_samp.rate(rw)

rwl <- lapply(path2wav, tuneR::readWave)
sapply(rwl, voice::get_samp.rate)
get_tbegin  

Time beginning

Description
Get time beginning from a data frame in RTTM standard.

Usage
get_tbegin(x)

Arguments
x  
A data frame in RTTM standard. See 'voice::read_rttm'.

Value
Numeric vector containing the time beginning in seconds.

Examples
library(voice)

url0 <- 'https://raw.githubusercontent.com/filipezabala/voiceAudios/main/rttm/sherlock0.rttm'
download.file(url0, destfile = paste0(tempdir(), '/sherlock0.rttm'))

rttm <- voice::read_rttm(tempdir())
(gtbn <- voice::get_tbegin(rttm$sherlock0.rttm))
class(gtbn)

get_tduration  

Time duration

Description
Get time duration from a data frame in RTTM standard.

Usage
get_tduration(x)

Arguments
x  
A data frame in RTTM standard. See 'voice::read_rttm'.

Value
Numeric vector containing the time duration in seconds.
Examples

library(voice)

url0 <- 'https://raw.githubusercontent.com/filipezabala/voiceAudios/main/rttm/sherlock0.rttm'
download.file(url0, destfile = paste0(tempdir(), '/sherlock0.rttm'))

rttm <- voice::read_rttm(tempdir())
(gtd <- voice::get_tdur(rttm$sherlock0.rttm))
class(gtd)

interp  Interpolate vectors

Description

Interpolate vectors, compressing to compact.to fraction. May remove zeros.

Usage

interp(
y, compact.to, drop.zeros = FALSE, to.data.frame = FALSE, round.off = NULL, weight = NULL
)

Arguments

y  A vector or time series.
compact.to  Proportion of remaining points after compaction, between (including) 0 and 1. If equals to 1 and keep.zeros = TRUE, the original vector is presented.
drop.zeros  Logical. Drop repeated zeros? Default: FALSE.
to.data.frame  Logical. Convert to data frame? Default: FALSE.
round.off  Number of decimal places of the interpolated y Default: NULL.
weight  Vector of weights with same length of y. Default: NULL.

Value

A list of interpolated x and y values with length near to compact.to*length(y).

See Also

rm0, interp_mc, interp_df
Examples

    library(voice)

    v1 <- 1:100
    (c1 <- interp(v1, compact.to = 0.2))
    length(c1$y)
    plot(1:100, type = 'l')
    points(c1$x, c1$y, col='red')

    # with weight
    (c2 <- interp(v1, compact.to = 0.2, weight = rev(v1)))
    plot(c1$y)
    points(c2$y, col = 'red')

    (v2 <- c(1:5, rep(0,10), 1:10, rep(0,5), 10:20, rep(0,10)))
    length(v2)
    interp(v2, 0.1, drop.zeros = TRUE, to.data.frame = FALSE)
    interp(v2, 0.1, drop.zeros = TRUE, to.data.frame = TRUE)
    interp(v2, 0.2, drop.zeros = TRUE)
    interp(v2, 0.2, drop.zeros = FALSE)

    (v3 <- c(rep(0,10), 1:20, rep(0,3)))
    (c3 <- interp(v3, 1/3, drop.zeros = FALSE, to.data.frame = FALSE))
    lapply(c3, length)
    plot(v3, type = 'l')
    points(c3$x, c3$y, col = 'red')

    (v4 <- c(rnorm(1:100)))
    (c4 <- interp(v4, 1/4, round.off = 3))

interp_df

Inperpolate data frames

Description

Interpolate data frames using multicore, compressing to compact.to fraction. May remove zeros.

Usage

    interp_df(
        x,
        compact.to,
        id = colnames(x)[1],
        colnum = NULL,
        drop.x = TRUE,
        drop.zeros = FALSE,
        to.data.frame = TRUE,
        round.off = NULL,
        weight = NULL,
Arguments

x  A data frame.
compact.to  Proportion of remaining points after interpolation. If equals to 1 and keep.zeros = TRUE, the original vector is presented.
id  The identification column. Default: colname of the first column of x.
colnum  A char vector indicating the numeric colnames. If NULL, uses the columns of the numeric class.
drop.x  Logical. Drop columns containing .x? Default: TRUE.
drop.zeros  Logical. Drop repeated zeros or keep 1 zero per null set? Default: FALSE.
to.data.frame  Logical. Should return a data frame? If FALSE returns a list. Default: TRUE.
round.off  Number of decimal places of the interpolated y. Default: NULL.
weight  Vector of weights with same length of y. Default: NULL.
mc.cores  The number of cores to mclapply. Default: 1.

Value

A data frame of interpolated values with nrow near to compact.to*length(x).

See Also

interp, interp_mc

Examples

library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
    pattern = glob2rx(*.wav), full.names = TRUE)

# getting Media data frame via lean call
M <- extract_features(dirname(path2wav), features = c('f0','fmt'),
    mc.cores = 1, verbose = FALSE)

(cM.df <- interp_df(M[,-(1:2)], 0.1, mc.cores = 1))
(cM.df2 <- interp_df(M[,-(1:2)], 0.1, drop.x = FALSE, mc.cores = 1))

dim(M)
dim(cM.df)
dim(cM.df2)
(cM.list <- interp_df(M[,-(1:2)], 0.1, to.data.frame = FALSE, mc.cores = 1))
interp_mc

Interpolate vectors using multicore

Description

Interpolate vectors using multicore

Usage

interp_mc(
  y, 
  compact.to, 
  drop.zeros = FALSE, 
  to.data.frame = FALSE, 
  round.off = NULL, 
  weight = NULL, 
  mc.cores = 1
)

Arguments

y A numeric vector, matrix or data frame.
compact.to Proportion of remaining points after compression. If equals to 1 and keep.zeros = TRUE, the original vector is presented.
drop.zeros Logical. Drop repeated zeros? Default: FALSE.
to.data.frame Logical. Convert to data frame? Default: FALSE.
round.off Number of decimal places of the interpolated y. Default: NULL.
weight Vector of weights with same length of y. Default: NULL.
mc.cores The number of cores to mclapply. Default: 1.

Value

A list of x and y convoluted values with length near to compact.to*length(y).

See Also

rm0, interp, interp_df

Examples

library(voice)
# Same result of interp() function if x is a vector
interp(1:100, compact.to = 0.1, drop.zeros = TRUE, to.data.frame = FALSE)
interp_mc(1:100, compact.to = 0.1, drop.zeros = TRUE, to.data.frame = FALSE)
interp(1:100, compact.to = 0.1, drop.zeros = TRUE, to.data.frame = TRUE)
is_mono <- interp_mc(1:100, compact.to = 0.1, drop.zeros = TRUE, to.data.frame = TRUE)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
                       pattern = glob2rx('*.wav'), full.names = TRUE)

# getting Media data frame
M <- voice::extract_features(dirname(path2wav), mc.cores = 1, verbose = FALSE)
M.num <- M[, -(1:3)]
nrow(M.num)
cml <- interp_mc(M.num, compact.to = 0.1, drop.zeros = TRUE,
                 to.data.frame = FALSE, mc.cores = 1)
names(cml)
lapply(cml$f0, length)

is_mono  Verify if an audio is mono

Description
Verify if an audio is mono

Usage
is_mono(x)

Arguments
x  Path to WAV audio file.

Value
Logical. ‘TRUE’ indicates a mono (one-channel) file. ‘FALSE’ indicates a non-mono (two-channel) file.

Examples
library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
                       pattern = glob2rx('*.wav'), full.names = TRUE)

is_mono(path2wav[1])
sapply(path2wav, is_mono)
mozilla_id_path

Sample IDs and paths

Description

A dataset containing sample IDs and paths from Ardila et al (2019) 'Common voice: A massively-multilingual speech corpus', used in Zabala (2023) 'voice: new approaches to audio analysis'. The considered sample contains 34,425 rows associated with 838 IDs (p_s = 2.4%).

Usage

mozilla_id_path

References


See Also

extract_features.

Examples

library(voice)
mozilla_id_path

notes

Assign notes to frequencies

Description

Returns a vector of notes for equal-tempered scale, A4 = 440 Hz.

Usage

notes(x, method = "spn", moving.average = FALSE, k = 11)

Arguments

x

Numeric vector of frequencies in Hz.

method

Method of specifying musical pitch. (Default: spn, i.e., Scientific Pitch Notation).

moving.average

Logical. Must apply moving average? (Default: FALSE).

k

Integer width of the rolling window used if moving.average is TRUE. (Default: 11).
The symbol '#' is being used to represent a sharp note, the higher in pitch by one semitone on Scientific Pitch Notation (SPN).

A vector containing the notes for equal-tempered scale, A4 = 440 Hz. When `method = 'spn'` the vector is of class 'ordered factor'. When `method = 'octave'` the vector is of class 'factor'. When `method = 'midi'` the vector is of class 'integer'.

https://pages.mtu.edu/~suits/notefreqs.html

Examples

```r
library(voice)
notes(c(220,440,880))
notes(c(220,440,880), method = 'octave')
notes(c(220,440,880), method = 'midi')
```

Description

Returns a tibble of frequencies on Scientific Pitch Notation (SPN) for equal-tempered scale, A4 = 440 Hz.

Usage

```r
notes_freq()
```
read_rttm

References

https://pages.mtu.edu/~suits/notefreqs.html

See Also

notes

Examples

library(voice)
notes_freq()

read_rttm Read RTTM files

Description

Read Rich Transcription Time Marked (RTTM) files in fromRttm directory.

Usage

read_rttm(fromRttm)

Arguments

fromRttm A directory/folder containing RTTM files.

Details

The Rich Transcription Time Marked (RTTM) files are space-delimited text files containing one turn per line defined by NIST - National Institute of Standards and Technology. Each line containing ten fields:

- type Type: segment type; should always by SPEAKER.
- file File ID: file name; basename of the recording minus extension (e.g., rec1_a).
- chn1 Channel ID: channel (1-indexed) that turn is on; should always be 1.
- tbeg Turn Onset – onset of turn in seconds from beginning of recording.
- tdur Turn Duration – duration of turn in seconds.
- ortho Orthography Field – should always by <NA>.
- stype Speaker Type – should always be <NA>.
- name Speaker Name – name of speaker of turn; should be unique within scope of each file.
- conf Confidence Score – system confidence (probability) that information is correct; should always be <NA>.
- slat Signal Lookahead Time – should always be <NA>.
Compress zeros.

Description

Transforms \( n \) sets of \( m > n \) zeros (alternated with sets of non zeros) into \( n \) sets of \( n \) zeros.

Usage

\( \text{rm0}(y) \)

Arguments

\( y \) A vector or time series.

Value

Vector with \( n \) zeros.
smooth_df

Examples

library(voice)

(v0 <- c(1:20,rep(0,10)))
(r0 <- rm0(v0))
length(v0)
length(r0)
sum(v0 == 0)

(v1 <- c(rep(0,10),1:20))
(r1 <- rm0(v1))
length(r1)

(v2 <- rep(0,10))
(r2 <- rm0(v2))
length(r2)

(v3 <- c(0:10))
(r3 <- rm0(v3))
length(r3)

(v4 <- c(rep(0,10),1:10,rep(0,5),10:20,rep(0,10)))
(r4 <- rm0(v4))
length(r4)
sum(v4 == 0)

smooth_df

Smooth numeric variables in a data frame

Description

Smooth numeric variables in a data frame

Usage

smooth_df(x, k = 11, id = colnames(x)[1], colnum = NULL, mc.cores = 1)

Arguments

x A data frame.
k Integer width of the rolling window. Default: 11.
id The identification column. Default: colname of the first column of x.
colnum A char vector indicating the numeric colnames. If NULL, uses the columns of the numeric class.
mc.cores The number of cores to mclapply. By default uses 1.
splitw

Value

Vector of interpolated values with length near to \texttt{compact.to*length(x)}.

See Also

\texttt{extract_features}

Examples

library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
        pattern = glob2rx('*.wav'), full.names = TRUE)

# minimal usage
M <- extract_features(path2wav, features = c('f0', 'fmt'))
(Ms <- smooth_df(M[-(1:2)]))
dim(M)
dim(Ms)

----------

\textbf{splitw} \hspace{4cm} \textit{Split Wave}

Description

Split WAV files either in \texttt{fromWav} directory or using (same names) RTTM files/subdirectories as guidance.

Usage

\texttt{splitw(}
    \texttt{fromWav,}
    \texttt{fromRttm = NULL,}
    \texttt{toSplit = NULL,}
    \texttt{autoDir = FALSE,}
    \texttt{subDir = FALSE,}
    \texttt{output = "wave",}
    \texttt{filesRange = NULL,}
    \texttt{full.names = TRUE,}
    \texttt{recursive = FALSE,}
    \texttt{silence.gap = 0.5}
\texttt{)}
splitw

Arguments

fromWav  Either WAV file or directory containing WAV files.
fromRttm  Either RTTM file or directory containing RTTM files. Default: NULL.
toSplit  A directory to write generated files. Default: NULL.
autoDir  Logical. Must the directories tree be created? Default: FALSE. See 'Details'.
subDir  Logical. Must the splitted files be placed in subdirectories? Default: FALSE.
output  Character string, the class of the object to return, either 'wave' or 'list'.
filesRange  The desired range of directory files (default: NULL, i.e., all files). Must be TRUE only if fromWav is a directory.
full.names  Logical. If TRUE, the directory path is prepended to the file names to give a relative file path. If FALSE, the file names (rather than paths) are returned. (default: TRUE) Used by base::list.files.
recursive  Logical. Should the listing recursively into directories? (default: FALSE) Used by base::list.files. Inactive if fromWav is a file.
silence.gap  The silence gap (in seconds) between adjacent words in a keyword. Rows with tdur <= silence.gap are removed. (default: 0.5)

Details

When autoDir = TRUE, the following directories are created: '..mp3', '..rttm', '..split' and '..musicxml'. Use getwd() to find the parent directory '..'.

Value

Splited audio files according to the correspondent RTTM file(s). See 'voice::diarize'.

See Also

voice::diarize

Examples

## Not run:
library(voice)

urlWav <- 'https://raw.githubusercontent.com/filipezabala/voiceAudios/main/wav/sherlock0.wav'
destWav <- paste0(tempdir(), '/sherlock0.wav')
download.file(urlWav, destfile = destWav)

urlRttm <- 'https://raw.githubusercontent.com/filipezabala/voiceAudios/main/rttm/sherlock0.rttm'
destRttm <- paste0(tempdir(), '/sherlock0.rttm')
download.file(urlRttm, destfile = destRttm)

splitDir <- paste0(tempdir(), '/split')
dir.create(splitDir)

splitw(destWav, fromRttm = destRttm, toSplit = splitDir)
Tag a data frame with media information

Description

Tag a data frame with media information

Usage

tag(
  x, 
  groupBy = "wav_path", 
  wavPath = unique(x$wav_path), 
  wavPathName = "wav_path", 
  tags = c("feat_summary"), 
  sortByGroupBy = TRUE, 
  filesRange = NULL, 
  features = "f0", 
  sex = "u", 
  windowShift = 5, 
  numFormants = 8, 
  numcep = 12, 
  dcttype = c("t2", "t1", "t3", "t4"), 
  fbtype = c("mel", "htkmel", "fcme1", "bark"), 
  resolution = 40, 
  usecmp = FALSE, 
  mc.cores = 1, 
  full.names = TRUE, 
  recursive = FALSE, 
  check.mono = FALSE, 
  stereo2mono = FALSE, 
  overwrite = FALSE, 
  freq = 44100, 
  round.to = 4, 
  verbose = FALSE
)

Arguments

x An Extended data frame to be tagged with media information. See references.

groupBy A variable to group the summary measures. The argument must be a character vector. (Default: groupBy = 'wav_path').
wavPath  
A vector containing the path(s) to WAV files. May be both as dirname or basename formats.

wavPathName  
A string containing the WAV path name. (Default: wavPathName = 'wav_path').

tags  
Tags to be added to x. See Details. (Default: 'feat_summary').

sortByGroupBy  
Logical. Should the function sort the Extended data frame x by groupBy? (Default: sortByGroupBy = TRUE).

filesRange  
The desired range of directory files. Should only be used when all the WAV files are in the same folder. (Default: NULL, i.e., all files).

features  
Vector of features to be extracted. (Default: 'f0').

sex  
= <code> set sex specific parameters where <code> = 'f'[emale], 'm'[ale] or 'u'[nknown] (default: 'u'). Used as 'gender' by wrassp::ksvF0, wrassp::forest and wrassp::mhsF0.

windowShift  
= <dur> set analysis window shift to <dur>ation in ms (default: 5.0). Used by wrassp::ksvF0, wrassp::forest, wrassp::mhsF0, wrassp::zcrana, wrassp::rfcana, wrassp::acfana, wrassp::cepstrum, wrassp::dftSpectrum, wrassp::cssSpectrum and wrassp::lpsSpectrum.

numFormants  
= <num> <num>ber of formants (Default: 8). Used by wrassp::forest.

numcep  
Number of Mel-frequency cepstral coefficients (cepstra) to return (Default: 12). Used by tuneR::melfcc.

dcttype  
Type of DCT used. 't1' or 't2', 't3' for HTK 't4' for feacalc (Default: 't2'). Used by tuneR::melfcc.

fbtype  
Auditory frequency scale to use: 'mel', 'bark', 'htkmel', 'fcmel' (Default: 'mel'). Used by tuneR::melfcc.

resolution  
= <freq> set FFT length to the smallest value which results in a frequency resolution of <freq> Hz or better (Default: 40.0). Used by wrassp::cssSpectrum, wrassp::dftSpectrum and wrassp::lpsSpectrum.

usecmp  

mc.cores  
Number of cores to be used in parallel processing. (Default: 1)

full.names  
Logical. If TRUE, the directory path is prepended to the file names to give a relative file path. If FALSE, the file names (rather than paths) are returned. (Default: TRUE) Used by base::list.files.

recursive  
Logical. Should the listing recursively into directories? (Default: FALSE) Used by base::list.files.

check.mono  
Logical. Check if the WAV file is mono. (Default: TRUE)

stereo2mono  
(Experimental) Logical. Should files be converted from stereo to mono? (Default: TRUE)

overwrite  
(Experimental) Logical. Should converted files be overwritten? If not, the file gets the suffix _mono. (Default: FALSE)

freq  
Frequency in Hz to write the converted files when stereo2mono=TRUE. (Default: 44100)

round.to  
Number of decimal places to round to. (Default: NULL)

verbose  
Logical. Should the running status be showed? (Default: FALSE)
write_list

Details

filesRange should only be used when all the WAV files are in the same folder.

Value

A tibble data frame containing summarized numeric columns using (1) mean, (2) standard deviation, (3) variation coefficient, (4) median, (5) interquartile range and (6) median absolute deviation.

Examples

library(voice)

# get path to audio file
path2wav <- list.files(system.file('extdata', package = 'wrassp'),
  pattern = glob2rx('*.wav'), full.names = TRUE)

# creating Extended synthetic data
E <- dplyr::tibble(subject_id = c(1,1,1,2,2,2,3,3,3),
  wav_path = path2wav)
E

# minimal usage
tag(E)

# canonical data
tag(E, groupBy = 'subject_id')

# limiting filesRange
tag(E, filesRange = 3:6)

# more features
Et <- tag(E, features = c('f0', 'fmt', 'rf', 'rcf', 'rpf', 'rfc', 'mfcc'),
  groupBy = 'subject_id')
Et
str(Et)

---

**write_list**

*Writes a list to a path*

**Description**

Writes a list to a path

**Usage**

write_list(x, path)
Arguments

- x: A list.
- path: A full path to file.

Value

A file named 'list.txt' in 'path'.

Examples

```r
## Not run:
library(voice)

pts <- list(x = cars[,1], y = cars[,2])
listFile <- paste0(tempdir(), '/quotesingle.Var/list.txt/quotesingle.Var')
voice::write_list(pts, listFile)
file.info(listFile)
system(paste0('head ', listFile))

## End(Not run)
```
Index

audio_time, 2

diarize, 3
duration, 4

enrich_rttm, 5
expand_model, 6
extract_features, 7, 22

feat_summary, 10

get_bit, 12
get_dur, 13
get_left, 14
get_right, 14
get_samp_rate, 15
get_tbeg, 16
get_tdur, 16

interp, 17
interp_df, 18
interp_mc, 20
is_mono, 21

mozilla_id_path, 22

notes, 22
notes_freq, 23

read_rttm, 24
rm0, 25

smooth_df, 26
splitw, 27

tag, 29

write_list, 31