

# Package ‘vocaldia’

May 8, 2020

**License** GPL-3

**LazyData** true

**Type** Package

**Title** Create and Manipulate Vocalisation Diagrams

**Version** 0.8.3

**Date** 2020-05-07

**Description** Create adjacency matrices of vocalisation graphs from dataframes containing sequences of speech and silence intervals, transforming these matrices into Markov diagrams, and generating datasets for classification of these diagrams by 'flattening' them and adding global properties (functionals) etc. Vocalisation diagrams date back to early work in psychiatry (Jaffe and Feldstein, 1970) and social psychology (Dabbs and Ruback, 1987) but have only recently been employed as a data representation method for machine learning tasks including meeting segmentation (Luz, 2012) <doi:10.1145/2328967.2328970> and classification (Luz, 2013) <doi:10.1145/2522848.2533788>.

**Depends** R (>= 3.0.0)

**Suggests** igraph, foreign

**Imports** graphics, stats, utils

**RoxygenNote** 7.1.0

**Encoding** UTF-8

**URL** <https://git.ecdf.ed.ac.uk/sluzfil/vocaldia>

**BugReports** <https://git.ecdf.ed.ac.uk/sluzfil/vocaldia/issues>

**NeedsCompilation** no

**Author** Saturnino Luz [aut, cre]

**Maintainer** Saturnino Luz <luzs@acm.org>

**Repository** CRAN

**Date/Publication** 2020-05-08 11:10:02 UTC

**R topics documented:**

vocaldia-package	2
anonymise	3
appendSpeechRate	4
atddia	5
getEntropy	6
getIDs	6
getPauseType	7
getPID	8
getPofAgivenB	8
getSampledVocalCountMatrix	9
getSampledVocalMatrix	10
getSilences	11
getSyllablesAndSilences	12
getTranscript	12
getTurnTakingMatrix	13
getTurnTakingProbMatrix	14
getTurnType	15
identifyGrpVocalisations	16
identifyPauses	17
identifyVocalisations	17
igraph.vocaldia	18
makeSessionDataSet	19
makeVocalStatsDataset	20
matrixExp	21
namePauses	21
plot.matrixseries	22
plot.vocaldia	23
printARFFfile	24
read.cha	25
startmatrix	26
staticMatrix	27
toDotNotation	28
vocmatrix	29
write.vocaldia	30
<b>Index</b>	<b>31</b>

**Description**

Create adjacency matrices of vocalisation graphs from dataframes containing sequences of speech and silence intervals, transforming these matrices into Markov diagrams, and generating datasets for classification of these diagrams by 'flattening' them and adding global properties (functionals) etc. Vocalisation diagrams date back to early work in psychiatry (Jaffe and Feldstein, 1970) and social psychology (Dabbs and Ruback, 1987) but have only recently been employed as a data representation method for machine learning tasks including meeting segmentation (Luz, 2012) <doi:10.1145/2328967.2328970> and classification (Luz, 2013) <doi:10.1145/2522848.2533788>.

**Author(s)**

Saturnino Luz <luzs@acm.org>

**References**

S. Luz. Automatic identification of experts and performance prediction in the multimodal math data corpus through analysis of speech interaction. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction, ICMI'13*, pages 575–582, New York, NY, USA, 2013. ACM.

S. Luz. The non-verbal structure of patient case discussions in multidisciplinary medical team meetings. *ACM Transactions on Information Systems*, 30(3):17:1–17:24, 2012

Dabbs, J. M. J. and Ruback, B. Dimensions of group process: Amount and structure of vocal interaction. *Advances in Experimental Social Psychology* 20, 123-169, 1987.

Jaffe, J. and Feldstein, S. Rhythms of dialogue. ser. *Personality and Psychopathology*. Academic Press, New York, 1976.

**See Also**

Useful links:

- <https://git.ecdf.ed.ac.uk/sluzfil/vocaldia>
- Report bugs at <https://git.ecdf.ed.ac.uk/sluzfil/vocaldia/issues>

---

anonymise

*anonymise: anonymise a vocalisation diagram*

---

**Description**

Anonymise a vocalisation diagram

**Usage**

```

anonymise(vd)

## S3 method for class 'vocaldia'
anonymise(vd)

## Default S3 method:
anonymise(vd)

```

**Arguments**

vd                    a vocalisation diagram (vocaldia object)

**Details**

"anonymise" a vocaldia turn taking probability matrix by replacing speaker names by variables  $s_1, \dots, s_n$  s.t.  $s_1$  is the speaker who spoke the least and  $s_n$  the one who did the most talking.

**Value**

a new vocaldia with speaker names replaced by variables  $s_1, \dots, s_n$  s.t.  $s_1$  is the speaker who spoke the least and  $s_n$  the one who did the most talking.

**Examples**

```

## Not run:
data(vocdia)
x2 <- getSampledVocalMatrix(subset(atddia, id=='Abbott_Maddock_01'),
                           individual=TRUE, nodecolumn='speaker')
anonymise(x2)

## End(Not run)

```

---

appendSpeechRate	<i>appendSpeechRate: append pre-generated speech rate data to given dataframe t</i>
------------------	---

---

**Description**

appendSpeechRate: append pre-generated speech rate data (see audioproc.R)

**Usage**

```
appendSpeechRate(t, file = NULL)
```

**Arguments**

t                    a table read through read.cha  
file                speech rate file

**Value**

dataframe t bound to speech rates per utterance

**Author(s)**

luzs

---

atddia

*A sample Medical Team Meeting dialogue encoded as a vocaldia*

---

**Description**

A dataset containing 38 dialogues (17 control patients, and 21 AD patients) and 7869 vocalisation events.

**Usage**

atddia

**Format**

A data frame with 7869 rows and 7 variables:

**id** The dialogue identifier

**begin** The start time of a speech turn or silence interval

**end** The end time of a speech turn or silence interval

**speaker** An identifier for the speaker of the turn, or Floor for silence.

**role** The speaker's role (patient, interviewer, other, or Floor)

**trans** The transcription of the turn (blanked out for anonymity)

**dx** The diagnosis (ad or nonad)

**Source**

This dataset was generated from the Carolina Conversations Collection, and used in the work described in De La Fuente, Albert and Luz: "Detecting cognitive decline through dialogue processing", 2017. For the full data set, please contact the Medical University of South Carolina (MUSC) <http://carolinaconversations.musc.edu/>

---

getEntropy	<i>getEntropy: safely return the Shannon entropy of a distribution.</i>
------------	---

---

**Description**

Compute the entropy of a distribution.

**Usage**

```
getEntropy(distribution)
```

**Arguments**

distribution    a probability distribution.

**Details**

Compute the entropy of a distribution.

**Value**

a numeric value.

---

getIDs	<i>getIDs get speaker role IDs (PAR, INV) and info from CHA content</i>
--------	---

---

**Description**

getIDs get speaker IDs from CHA content

**Usage**

```
getIDs(text)
```

**Arguments**

text            a string vector containing the lines of a CHA file

**Value**

a vector with participants IDs

**Author(s)**

luzs

---

getPauseType                    *getPauseType: name pause type between two vocalisation events.*

---

### Description

Identify the type of pause between vocalisations.

### Usage

```
getPauseType(prevspeaker, nextspeaker)
```

### Arguments

prevspeaker	speaker of the vocalisation immediately before Floor
nextspeaker	speaker of the vocalisation immediately after Floor

### Details

The type of pause a 'Floor' (silence) event represents can be: 'Pause', 'SwitchingPause', 'Grp-Pause', or 'GrpSwitchingPause'. See (Luz, 2013) for details.

### Value

the pause type.

### See Also

[namePauses](#)

### Examples

```
getPauseType('a', 'b')  
## [1] "SwitchingPause"  
getPauseType('a', 'Grp')  
## [1] "SwitchingPause"  
getPauseType('Grp', 'Grp')  
## [1] "GrpPause"  
getPauseType('Grp', 'a')  
## [1] "GrpSwitchingPause"  
getPauseType('a', 'a')  
##[1] "Pause"
```

---

getPID *getIDs get study-wide unique patient IDs from CHA content*

---

### Description

getPIDs get study-wide unique patient IDs from CHA content

### Usage

getPID(text)

### Arguments

text                    a string vector containing the lines of a CHA file

### Value

a vector with participants IDs

### Author(s)

luzs

---

getPofAgivenB *getPofAgivenB: transtion probability.*

---

### Description

Conditional (transition ) probability

### Usage

getPofAgivenB(a, b, ttarray)

### Arguments

a                    target node  
 b                    source node  
 ttarray            adjacency matrix

### Details

Retrieve  $p(a|b)$ , probability of a transition from b to a in an adjacency matrix

### Value

a transition probability.



---

```
getSampledVocalCountMatrix
```

*getSampledVocalCountMatrix: generate vocalisation diagrams*

---

## Description

Generate a count vocalisation diagram through 'sampling'.

## Usage

```
getSampledVocalCountMatrix(  
  cdf,  
  rate = 1,  
  individual = FALSE,  
  noPauseTypes = FALSE,  
  begin = "begin",  
  end = "end",  
  nodecolumn = "role"  
)
```

## Arguments

<code>cdf</code>	a data frame consisting, minimally, of a column for vocalisation/pause start times, a column for end times, and a column identifying the speaker, speaker role or 'Floor' (for silences).
<code>rate</code>	the rate at which to sample the vocalisation events (in seconds)
<code>individual</code>	whether to include individual speakers or group them into a single Vocalisation node
<code>noPauseTypes</code>	if TRUE, ignore distinctions between pauses (SwitchingPause, GrpSwitchingPause, etc)
<code>begin</code>	the name of the column containing the start time of the vocalisation event in a row.
<code>end</code>	the name of the column containing the end time of the vocalisation event in the same row.
<code>nodecolumn</code>	the name of the column containing the node (speaker) name (e.g. 'speaker', 'role').

## Details

A vocalisation diagram (vocaldia) is a representation of a dialogue as a Markov process whose cell  $\langle m,n \rangle$  contains the transition probability from node  $n$  to node  $m$ . This function for 'cases' (an identifier for a case or a vector of identifiers identifying a set of cases) in data frame 'df', obtained by sampling the timeline every 'rate'-th second (see `getSampledVocalCountMatrix`).

**Value**

a vocaldia object, consisting of a vocalisation matrix (vocmatrix) where cell <m,n> contains the counts of transitions from node n to node m, and a table of prior probabilities (stationary distribution) per node.

**See Also**

(Luz, 2013)

**Examples**

```
data(vocdia)
getSampledVocalCountMatrix(subset(atddia,
  id='Abbott_Maddock_01'), nodecolumn='role')
```

---

getSampledVocalMatrix *getSampledVocalCountMatrix: generate vocalisation diagrams*

---

**Description**

Generate a probabilistic vocalisation diagram through 'sampling'.

**Usage**

```
getSampledVocalMatrix(df, ...)
```

**Arguments**

df	a data frame consisting, minimally, of a column for vocalisation/pause start times, a column for end times, and a column identifying the speaker, speaker role or 'Floor' (for silences).
...	general parameter to be passed to <a href="#">getSampledVocalCountMatrix</a>

**Details**

A vocalisation diagram (vocaldia) is a representation of a dialogue as a Markov process whose cell <m,n> contains the transition probability from node n to node m).

**Value**

a vocaldia object, consisting of a vocalisation matrix (vocmatrix) where cell <m,n> contains the transition probability from node n to node m, and a table of prior probabilities (stationary distribution) per node.

**Author(s)**

Saturnino Luz <luzs@acm.org>

## References

S. Luz. Automatic identification of experts and performance prediction in the multimodal math data corpus through analysis of speech interaction. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction, ICMI'13*, pages 575–582, New York, NY, USA, 2013. ACM.

## See Also

[getSampledVocalCountMatrix](#)

## Examples

```
data(vocdia)
getSampledVocalMatrix(subset(atddia,
                             id=='Abbott_Maddock_01'),nodecolumn='speaker', individual=TRUE)
```

---

getSilences	<i>getSilences read silences file</i>
-------------	---------------------------------------

---

## Description

getSilences read silences file

## Usage

```
getSilences(file, sildir = NULL, silsuffix = "c.mp3.csv")
```

## Arguments

file	CSV formatted silences file
sildir	dir where silence files are
silsuffix	## suffix for silence files

## Value

silences dataframe

## Author(s)

luzs

---

getSyllablesAndSilences

*getSyllablesAndSilences: process Praat's grid for syllable nuclei*

---

### Description

getSyllablesAndSilences: process Praat's grid for syllable nuclei, based on De Jong's approach

### Usage

getSyllablesAndSilences(txtgrid)

### Arguments

txtgrid            Path to Praat grid file generated by praat-syllable-syllable-nuclei-v2

### Value

list of syllables and silences

### Author(s)

luzs

### References

De Jong, N. H. and Wempe, T. (2009). Praat script to detect syllable nuclei and measure speech rate automatically. *Behavior Research Methods*, 41(2):385–390, May.

---

getTranscript

*getTranscript: get transcription lines from .cha content*

---

### Description

getTranscript

### Usage

getTranscript(text)

### Arguments

text                a string vector containing the lines of a CHA file

### Value

a list of transcriptions (participant and interviewer utterances)

**Author(s)**

luzs

---

 getTurnTakingMatrix    *getSampledVocalCountMatrix: generate vocalisation diagrams*


---

**Description**

Generate a vocalisation diagram with absolute vocalisation durations.

**Usage**

```
getTurnTakingMatrix(
  df,
  begin = "begin",
  end = "end",
  nodecolumn = "role",
  individual = FALSE,
  noPauseTypes = FALSE
)
```

**Arguments**

df	a data frame consisting, minimally, of a column for vocalisation/pause start times, a column for end times, and a column identifying the speaker, speaker role or 'Floor' (for silences).
begin	the name of the column containing the start time of the vocalisation event in a row.
end	the name of the column containing the end time of the vocalisation event in the same row.
nodecolumn	the name of the column containing the node (speaker) name (e.g. 'speaker', 'role').
individual	whether to include individual speakers or group them into a single Vocalisation node
noPauseTypes	if TRUE, ignore distinctions between pauses (SwitchingPause, GrpSwitching-Pause, etc)

**Details**

A vocalisation diagram (vocaldia) is a representation of a dialogue as a Markov process whose cell  $\langle m, n \rangle$  contains the transition probability from node  $n$  to node  $m$ ). Unlike [getSampledVocalCountMatrix](#) this function accumulates event durations directly, therefore resulting in no self-transitions (in general).

**Value**

a vocaldia object, consisting of a vocalisation matrix (vocmatrix) where cell <m,n> contains the counts of transitions from node n to node m, and a table of absolute durations of vocalisation events.

**References**

S. Luz. Automatic identification of experts and performance prediction in the multimodal math data corpus through analysis of speech interaction. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction, ICMI'13*, pages 575–582, New York, NY, USA, 2013. ACM.

**See Also**

(Luz, 2013) and [getTurnTakingMatrix](#).

**Examples**

```
x <- subset(atddia, id=='Abbott_Maddock_01')
getTurnTakingMatrix(x)
getTurnTakingMatrix(x, individual=TRUE)
```

---

```
getTurnTakingProbMatrix
```

*getTurnTakingProbMatrix: create a vocaldia from a data.frame.*

---

**Description**

Convert a data frame into a vocalisation diagram using counts rather than sampling.

**Usage**

```
getTurnTakingProbMatrix(df, individual = FALSE, ...)
```

**Arguments**

df	a data frame consisting, minimally, of a column for vocalisation/pause start times, a column for end times, and a column identifying the speaker, speaker role or 'Floor' (for silences).
individual	whether to include individual speakers or group them into a single Vocalisation node
...	other parameters to be passed to <a href="#">getTurnTakingMatrix</a> .

**Details**

Unlike [getSampledVocalMatrix](#), this function is based on transition counts rather than sampled intervals. As a result, where in this version self transitions will always be set to 0 (since a vocalisation by a speaker is never followed by another vocalisation by the same speaker) in the sampled version self transits will usually dominate the distribution, since the speaker who is speaking now is very likely to be the one who were speaking one second ago.

**Value**

a vocaldia object, consisting of a vocalisation matrix (vocmatrix) where cell  $(m, n)$  contains the probabilities  $P(n|m)$  transitions to node  $n$  from node  $m$ , and a table of prior probabilities (stationary distribution) per node.

**See Also**

(Luz, 2013) and [getTurnTakingMatrix](#).

S. Luz. Automatic identification of experts and performance prediction in the multimodal math data corpus through analysis of speech interaction. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction, ICMI'13*, pages 575–582, New York, NY, USA, 2013. ACM.

**Examples**

```
x <- subset(atddia, id=='Abbott_Maddock_01')
getTurnTakingProbMatrix(x)
getTurnTakingProbMatrix(x, individual=TRUE)
```

---

getTurnType

*getTurnType: return type of turn*

---

**Description**

Identify turn types

**Usage**

```
getTurnType(
  df,
  i,
  individual = FALSE,
  nodecolumn = "speaker",
  noPauseTypes = F
)
```

**Arguments**

df	a data frame consisting, minimally, of a column for vocalisation/pause start times, a column for end times, and a column identifying the speaker, speaker role or 'Floor' (for silences).
i	the identifier (index number) whose type will be returned
individual	if TRUE, return the identifier, a Pause or Grp
nodecolumn	the name of the column containing the node (speaker) name (e.g. 'speaker', 'role').
noPauseTypes	if TRUE, ignore distinctions between pauses (SwitchingPause, GrpSwitchingPause, etc)

**Details**

Return one of Vocalisation, GrpVocalisation, ... or identifier.

**Value**

a string containing the turn type or identifier.

**Examples**

```
data(vocdia)
atddia[1:10,]
getTurnType(atddia, 3, nodecolumn='role') ## a vocalisation
getTurnType(atddia, 4, nodecolumn='role') ## a pause
```

---

identifyGrpVocalisations

*identifyGrpVocalisations: replace appropriate vocalisation types*

---

**Description**

Identify group vocalisations

**Usage**

```
identifyGrpVocalisations(vocvector)
```

**Arguments**

vocvector      a character vector containing a sequence of vocalisation events

**Details**

Standardise identifier for group vocalisations

**Value**

A vector with all events replaced by the appropriate type identifier.

**Examples**

```
data(vocdia)
identifyGrpVocalisations(atddia$speaker[1:60])
```



---

identifyPauses      *identifyPauses: label pauses according to type.*

---

**Description**

Assign types to the pauses (Floor events) in a sequence

**Usage**

```
identifyPauses(vocvector)
```

**Arguments**

vocvector      a character vector containing a sequence of vocalisation events

**Details**

Identify the pauses in a vector as one of the pauses in pauseTypes

**Value**

A vector with all Floor events replaced by the appropriate pause type identifier.

**Examples**

```
data(vocdia)
identifyPauses(atddia$speaker[1:60])
```

---

identifyVocalisations      *identifyVocalisations: replace appropriate vocalisation types*

---

**Description**

Identify switching vocalisations

**Usage**

```
identifyVocalisations(vocvector, idswitchvoc = T)
```

**Arguments**

vocvector      a character vector containing a sequence of vocalisation events  
idswitchvoc    if TRUE distinguish between SwitchingVocalisation and Vocalisation.



**Examples**

```
data(vocdia)
if (require('igraph'))
  igraph.vocaldia(getSampledVocalMatrix(subset(atddia,
                                             id=='Abbott_Maddock_01'),
                                       individual=TRUE, nodecolumn='speaker'))
```

---

makeSessionDataSet	<i>makeSessionDataSet: create a data frame for a session (e.g. cookie scene description) based on .cha transcription files</i>
--------------------	--

---

**Description**

makeSessionDataSet: create a data frame for a session (e.g. cookie scene description)

**Usage**

```
makeSessionDataSet(
  f,
  sildir = NULL,
  silsuffix = "c.mp3.csv",
  srdir = "../data/ADReSS/speech_rate/",
  srsuffix = "sra",
  sprate = T
)
```

**Arguments**

f	CHA file to read
sildir	directory where silence profiles are stored
silsuffix	suffix for silence files
srdir	directory where speech rate csv (1 value per utterance) files are stored
srsuffix	the suffix of the speech rate files (default: sre)
sprate	estimate speech rate? (default: TRUE)

**Value**

a speech session data frame

**Author(s)**

luzs

---

makeVocalStatsDataset *makeVocalStatsDataset: create a dataset of vocalisation statistics (1 row per patient)*

---

## Description

Build a data frame createwith vocalisation statistics

## Usage

```
makeVocalStatsDataset(
  dir = c("data/Pitt/Dementia/cookie", "data/Pitt/Control/cookie"),
  sildir = NULL,
  silsuffix = "c.mp3.csv",
  srdir = "data/Pitt/speech_rate/",
  srsuffix = "sra",
  sprate = T
)
```

## Arguments

dir	a string or vector containing the location (directory path) of the DementiaBank transcript files (.cha files)
sildir	directory where silence csv files are stored
silsuffix	the suffix of the silence profile files 'c.mp3.csv'. The format of such files should be the format used by Audacity label files, i.e. 'start time, end time, label' (without header), where 'label' should be 'silence'
srdir	directory where speech rate csv (1 value per utterance) files are stored
srsuffix	the suffix of the speech rate files (default: sre)
sprate	compute speech rate? (not in use yet)

## Value

a session's vocalisation feature stats

## Examples

```
## Not run:
makeVocalStatsDataset(dir=c('ADReSS-IS2020-data/train/transcription/cc/',
                           'ADReSS-IS2020-data/train/transcription/cd/'),
                      sildir='ADReSS/silence/',
                      srdir='ADReSS/speech_rate/',
                      silsuffix='.wav-sil.csv')
```

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

---

matrixExp	<i>matrixExp: raise matrix to exp.</i>
-----------	--

---

**Description**

Matrix exponentials

**Usage**

```
matrixExp(matrix, exp, mmatrix = matrix)
```

**Arguments**

matrix	a matrix
exp	the power to which matrix will be raised
mmatrix	a placeholder.

**Details**

A (sort of) exponential function for matrix multiplication (to be used with [staticMatrix](#)).

**Value**

matrix<sup>exp</sup>

**Examples**

```
data(vocdia)
matrixExp(vocmatrix$ttarray, 3)
```

---

namePauses	<i>namePauses: name pause types.</i>
------------	--------------------------------------

---

**Description**

Replace identified pause pause types in data frame.

**Usage**

```
namePauses(df, nodecolumn = "role")
```

**Arguments**

df	a data frame consisting, minimally, of a column for vocalisation/pause start times, a column for end times, and a column identifying the speaker, speaker role or 'Floor' (for silences).
nodecolumn	the name of the column containing the node (speaker) name (e.g. 'speaker', 'role').

**Details**

replace all 'Floor' speakers in df by 'Pause', 'SwitchingPause' etc, and return a new data fame containing pause types in place of 'Floor' (see markov.R, identifyPauses() for a better implementation)

**Value**

a data.frame with pauses in nodecolumn replaced by different pause types.

**See Also**

[identifyPauses](#) for a better implementation

**Examples**

```
data(vocdia)
x <- subset(atddia, id=='Abbott_Maddock_01')
x[1:15,1:6]
namePauses(x)[1:15,1:6]
```

---

plot.matrixseries      *plotConvergence: plots Markov diagram convergence.*

---

**Description**

Visualise convergence properties of vocalisation graphs

**Usage**

```
## S3 method for class 'matrixseries'
plot(x, ..., par = list(), interact = F)
```

**Arguments**

x	an object of class matrixseries; a list where the $i^{th}$ element corresponds to $M^i$ .
...	extra graphics parameters for plot.
par	graphic parameters alist
interact	if TRUE, pauses the drawing after each node.

**Details**

A 'toy' for visualisation of convergence properties of vocalisation graphs. Plot the convergence paths of each Vocalisation event (i.e. each row-column transition probability, grouped by colour according to the inciding node)

**Value**

the matrixseries

**Examples**

```
data(vocdia)
plot(staticMatrix(vocmatrix$ttarray, digits=4, history=TRUE))
```

---

plot.vocaldia      *plot.vocaldia*

---

**Description**

Plot a vocalisation diagram

**Usage**

```
## S3 method for class 'vocaldia'
plot(x, ...)
```

**Arguments**

x                    a vocalisation diagram  
...                   arguments for the layout algorithm

**Details**

Plot a vocalisation diagram

**Value**

NULL

**Examples**

```
data(vocdia)
if (require('igraph'))
  plot(getSampledVocalMatrix(subset(atddia, id=='Abbott_Maddock_01'),
                             individual=TRUE, nodecolumn='speaker'))
```

---

printARFFfile

*printARFFfile: Create arff files by creating and flattening vocaldias*


---

## Description

Generate ARFF files from vocalisation diagrams

## Usage

```
printARFFfile(
  df,
  ids = c(),
  idcolumn = "id",
  noPauseTypes = F,
  sampled = 0,
  individual = TRUE,
  nodecolumn = "role",
  classcolumn = "dx",
  file = ""
)
```

## Arguments

df	df a data frame consisting, minimally, of a column for vocalisation/pause start times, a column for end times, and a column identifying the speaker, speaker role or 'Floor' (for silences).
ids	Ids of dialogues to generate (as defined in column named idcolumn)
idcolumn	the name of the column containing the dialogue id
noPauseTypes	if TRUE, ignore distinctions between pauses (SwitchingPause, GrpSwitchingPause, etc)
sampled	if >0 use <a href="#">getSampledVocalMatrix</a> with rate=sampled
individual	whether to include individual speakers or group them into a single Vocalisation node
nodecolumn	the name of the column containing the node (speaker) name (e.g. 'speaker', 'role').
classcolumn	the name of the column containing the target class (or value).
file	name of ARFF file to be generated, or "" (print to console).

## Details

Use this function to generate turn-taking diagrams in ARFF format for



## References

S. Luz. Automatic identification of experts and performance prediction in the multimodal math data corpus through analysis of speech interaction. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction, ICMI'13*, pages 575–582, New York, NY, USA, 2013. ACM.

## See Also

[getSampledVocalCountMatrix](#), [getTurnTakingProbMatrix](#).

## Examples

```
data(vocdia)
atdarff <- tempfile(pattern='vocaldia-', fileext='arff')
printARFFfile(atddia, individual=TRUE, classcolumn='dx',
              file=atdarff, noPauseTypes=FALSE)
library("foreign")
x1 <- read.arff(atdarff)
x1[1:3,]
## remove empty columns
x1[,c(unlist(apply(x1[1:(ncol(x1)-1)],2,sum)!=0), TRUE)]
```

---

read.cha	<i>read.cha read CHA transcription file (format used by DementiaBank)</i>
----------	---

---

## Description

read.cha: read CHA transcription file (format used by DementiaBank)

## Usage

```
read.cha(file, sildir = NULL, silsuffix = "c.mp3.csv")
```

## Arguments

file	.cha file to read
sildir	silences directory
silsuffix	silence files suffix

## Value

a list containing the PID, a dataframe containing the speaker IDs and demographics, and a dataframe containing the speaker IDs, transcribed utterances, start and end times, speech rates etc.

## Author(s)

luzs

---

startmatrix	<i>startmatrix: return the first matrix of a converging series.</i>
-------------	---

---

## Description

Access initial matrix in a `matrixseries`

## Usage

```
startmatrix(mseries)

## Default S3 method:
startmatrix(mseries)

## S3 method for class 'matrixseries'
startmatrix(mseries)
```

## Arguments

`mseries` a `matrixseries` object

## Details

Access initial matrix in a `matrixseries`

## Value

the initial matrix.

## Examples

```
## Not run:
data(vocdia)
x2 <- staticMatrix(vocmatrix$ttarray, digits=4, history=TRUE)
## original matrix
startmatrix(x2)

## End(Not run)
```

---

staticMatrix                    *staticMatrix Iterate until transition probabilities converge (or give up).*

---

### Description

Compute the stationary distribution for a Markov diagram

### Usage

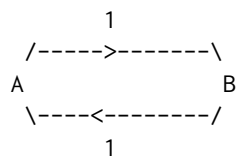
```
staticMatrix(matrix, limit = 1000, digits = 4, history = F)
```

### Arguments

matrix	an adjacency matrix of transition probabilities
limit	maximum number of iterations until we give up on convergence
digits	the number of decimal places to compare
history	if TRUE, keep track of all matrix products

### Details

Return static matrix (i.e. the stationary distribution) for the Markov process represented by the given adjacency matrix. In the particular case of vocaldia's, each column should roughly correspond to the amount of time a speaker held the floor for). Of course, not all Markov chains converge, an example being:



which gives

$$\begin{array}{l}
 . \quad | \ 0 \ 1 \ | \quad \quad \quad | \ 0x0+1x1 \ 0x1+1x0 \ | \quad | \ 1 \ 0 \ | \\
 . \ M = | \ 1 \ 0 \ | \quad \text{and} \quad M^2 = | \ 1x0+0x1 \ 1x1+1x0 \ | = | \ 0 \ 1 \ |
 \end{array}$$

### Value

a matrixseries object; that is, a list where each element is either the initial matrix or the product of the two preceding matrices

**Examples**

```

data(vocdia)
x2 <- staticMatrix(vocmatrix$ttarray, digits=4, history=TRUE)
## original matrix
round(x2[[1]],3)
## stationary matrix (M^139)
round(x2[[length(x2)]],3)

```

---

toDotNotation

*toDotNotation: conver vocaldia to graphviz dot notation*


---

**Description**

Create vocalisation diagram to file in dot (graphviz) notation

**Usage**

```

toDotNotation(
  vd,
  individual = T,
  varsizenode = T,
  shape = "circle",
  fontsize = 16,
  rankdir = "LR",
  nodeattribs = "fixedsize=true;",
  comment = ""
)

```

**Arguments**

vd	a vocalisation diagram
individual	if TRUE write individual node names
varsizenode	if true set varsizenode in dot
shape	node shape
fontsize	font size
rankdir	direction of ranking (LR, RF etc)
nodeattribs	attributes for node
comment	comments

**Details**

Create a vocalisation diagram in dot notation

**Value**

character data containing the diagram in dot format.

**See Also**

graphviz manual

**Examples**

```
data(vocdia)
toDotNotation(getSampledVocalMatrix(subset(atddia,
                                         id=='Abbott_Maddock_01'),
                                     individual=TRUE, nodecolumn='speaker'))
```

---

vocmatrix

*A sample vocalisation matrix*

---

**Description**

A vocaldia object containing a 3-speaker dialogue

**Usage**

```
vocmatrix
```

**Format**

A list containing 2 arrays

**ttarray** The vocaldia adjacency matrix

**tdarray** The proportional durations (stationary probabilities) of each event (node)

**Source**

This dataset was generated from the Multimodal Learning Analytics dataset, for the eponymous ICMI'13 Grand Challenge. The use these vocaldias were put to is described in Luz (2013). The full dataset and code is available at <https://gitlab.scss.tcd.ie/saturnino.luz/icmi-mla-challenge>

**References**

S. Luz. Automatic identification of experts and performance prediction in the multimodal math data corpus through analysis of speech interaction. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction, ICMI'13*, pages 575–582, New York, NY, USA, 2013. ACM.

---

write.vocaldia	<i>write.vocaldia</i>
----------------	-----------------------

---

**Description**

Write vocalisation diagram to file in dot (graphviz) notation

**Usage**

```
write.vocaldia(vd, file = "", ...)
```

**Arguments**

vd	a vocalisation diagram
file	name of file to which dot diagram will be written.
...	arguments passed on to toDotNotation. If "", write to STDOUT.

**Details**

Write a vocalisation diagram

**Value**

NULL

**Examples**

```
data(vocdia)
write.vocaldia(getSampledVocalMatrix(subset(atddia,
                                         id=='Abbott_Maddock_01'),
                                         individual=TRUE, nodecolumn='speaker'),
               file=tempfile(pattern='vocaldia-', fileext='.dot'))
```

# Index

## \*Topic **datasets**

- atddia, 5
- vocmatrix, 29
  
- anonymise, 3
- appendSpeechRate, 4
- atddia, 5
  
- getEntropy, 6
- getIDs, 6
- getPauseType, 7
- getPID, 8
- getPofAgivenB, 8
- getSampledVocalCountMatrix, 9, 10, 11, 13, 25
- getSampledVocalMatrix, 10, 14, 24
- getSilences, 11
- getSyllablesAndSilences, 12
- getTranscript, 12
- getTurnTakingMatrix, 13, 14, 15
- getTurnTakingProbMatrix, 14, 25
- getTurnType, 15
  
- identifyGrpVocalisations, 16
- identifyPauses, 17, 22
- identifyVocalisations, 17
- igraph.vocaldia, 18
  
- makeSessionDataSet, 19
- makeVocalStatsDataset, 20
- matrixExp, 21
  
- namePauses, 7, 21
  
- plot.matrixseries, 22
- plot.vocaldia, 23
- printARFFfile, 24
  
- read.cha, 25
  
- startmatrix, 26
  
- staticMatrix, 21, 27
  
- toDotNotation, 28
  
- vocaldia (vocaldia-package), 2
- vocaldia-package, 2
- vocmatrix, 29
  
- write.vocaldia, 30