Package: rSWeeP (via r-universe)

September 30, 2024

Type Package

Title Spaced Words Projection (SWeeP)

Version 1.17.3

Date 2024-04-28

Maintainer Camila P Perico <camilapp94@gmail.com>

Description ``Spaced Words Projection (SWeeP)" is a method for representing biological sequences using vectors preserving inter-sequence comparability.

Depends foreach, doParallel, parallel, Biostrings, methods, utils

Imports tools, stringi,

Suggests Rtsne, ape, Seurat, knitr, rmarkdown, tictoc, BiocStyle, testthat (>= 3.0.0)

Encoding UTF-8

RoxygenNote 7.3.2

VignetteBuilder knitr

Config/testthat/edition 3

License GPL (>= 2)

Repository https://bioc.r-universe.dev

RemoteUrl https://github.com/bioc/rSWeeP

RemoteRef HEAD

RemoteSha a17986595e4db32a6d1c1813191b943733c71326

Contents

rSWeeP-package *rSWeeP – Alignment-free method for vectorising biological sequences*

Description

The ´rSWeeP´ package is an R implementation of the Spaced Words Projection (SWeeP) method (De Pierri, 2019). The main function of this package is to provide a vector representation of biological sequences (nucleotides or amino acids), and thus favor alignment-free phylogenetic studies. Each sequence provided is represented by a compact numerical vector which is easier to analyze. SWeeP uses k-mers counting for representing the sequences in high dimensional vector (HDV) and then projected into a low dimensional vector (LDV) through random projection using an orthonormal base. The LDV represents the biological sequence and is handable for comparative analisys and machine learning implements.

In addition, the package allows general dimensionality reduction of RNAseq data and matrices.

More information about 'rSWeeP' can be found at <https://github.com/CamilaPPerico/rSWeeP> Tutorials on package use can be found at <https://aibialab.github.io/rSWeeP>

Details

The main functions are SWeeP, SWeePlite and orthBase. Additionally are available extractHDV

Author(s)

Maintainer: Camila P Perico <camilapp94@gmail.com>

Camila P. Perico [cre, aut, cph] Roberto T. Raittz [aut, cph] Danrley R. Fernandes [aut] Mariane G. Kulik [aut]

References

De Pierri, C. R., Voyceik, R., Santos de Mattos, L. G. C., Kulik, M. G., Camargo, J. O., Repula de Oliveira, A. M., ... & Raittz, R. T. (2020). Sweep: representing large biological sequences datasets in compact vectors. Scientific reports, 10(1), 91.

Examples

```
## Not run:

 path <- paste (system.file("examples/aaMitochondrial/",package = "rSWeeP"),'/', sep = '')
  sw = SWeePlite(path,mask=c(1,1,0,1,1),psz=1500,seqtype='AA',ncores=4)

 pathmetadata <- system.file(package = "rSWeeP" , "examples" , "metadata_mitochondrial.csv")
  mt = read.csv(pathmetadata,header=TRUE)
  pca_output <- prcomp (sw$proj , scale = FALSE)
 plot(pca_output$x[,1],pca_output$x[,2] , xlab = 'PC-1' , ylab = 'PC-2', pch=20, col=mt$id)
 legend("bottomright",unique(mt$family),col=as.character(c(1:length(unique(mt$family)))),pch=20)
```

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

---

extractHDV *Function for obtaining the HDV (High Dimensional Vector) matrix*

Description

Function for obtaining the HDV matrix without projecting it low dimensional vector (LDV). Each line of the HDV corresponds to the counting of k-mers of a biological sequence, organized in a structured way.

Usage

```
extractHDV(input, mask = NULL, seqtype = "AA", ...)

## S4 method for signature 'AAStringSet'
extractHDV(
  input,
  mask = NULL,
  seqtype = NULL,
  bin = FALSE,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'DNAStringSet'
extractHDV(
  input,
  mask = NULL,
  seqtype = NULL,
  bin = FALSE,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'RNAStringSet'
extractHDV(
  input,
  mask = NULL,
  seqtype = NULL,
  bin = FALSE,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'BStringSet'
```

```
extractHDV(
  input,
  mask = NULL,
  seqtype = NULL,
  bin = FALSE,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'BString'
extractHDV(
  input,
  mask = NULL,
  seqtype = NULL,
  bin = FALSE,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'character'
extractHDV(
  input,
  mask = NULL,
  seqtype = "AA",
  bin = FALSE,
  extension = "",
  verbose = TRUE
)
```

Arguments

input	There are two input formats available: (a) 'BStringSet' (variants: 'AAStringSet', 'RNAStringSet', 'DNAStringSet'). Biological sequence format loaded in memory; (b) 'character'. String containing a path to a folder with FASTA files.
mask	readging mask. Default for amino acids is 'c(2,1,2)' and for nucleotides c(5,5,5)#'
seqtype	type of data: AA for amino acid, NT for nucleotide. The default is 'AA'
...	other arguments of the function itself
bin	binary mode (TRUE), or counting mode (FALSE) for HDV construction. Default is FALSE
concatenate	defines whether to treat each sequence individually or to concatenate them into a single sequence. Available only for inputs in biological sequence format. The default is FALSE.
verbose	verbose mode. The default is TRUE
extension	extension of files desired to concatenate (Optional). Available only for input type path to folder with FASTA files.

Value

'extractHDV' returns a 'list' containing:

- HDV: a 'matrix' containing the High Dimensional Vectors of the given FASTAS
- info: aditional information of the process. This object is subdivided in:
  - headers: a 'character' containing the list of samples
  - mask: a 'integer' containing the mask used
  - SequenceType: a 'character' containing the type of the sequence (amino acid: AA, ou nucleotide: NT)
  - extension: a 'character' containing the list of extensions considered
  - concatenate : a boolean corresponding to the concatenation of sequences
  - bin: a 'character' containing if binary or counting
  - version : a character corresponding to the version of the package
  - saturation: a 'vector' containing the filled (non-zero) percentage of the HDV for each sample
  - timeElapsed: a 'double' containing the elapsed time in seconds

Examples

```
# get the path to the folder containing the FASTA files
path = paste (system.file("examples/aaMitochondrial/",package = "rSWeeP"),'/', sep = '')

# define the parameters
mask = c(2,1,2)

# get the vectors that represent the sequences in high dimension (without projection)
HDV = extractHDV(input=path,mask=mask,seqtype='AA',bin=FALSE,extension=c('.faa','.fas','.fasta'))
```

---

orthBase *Generate a orthonormal matrix*

Description

Generate a orthonormal matrix for specified parameters for ´SWeeP´ function

Usage

```
orthBase(lin = NULL, col, seqtype = "AA", mask = c(2, 1, 2), seed = NULL)
```

Arguments

`lin`	Number of rows in the desired matrix.
`col`	Number of columns in the desired matrix, which means projection size (psz)
`seqtype`	type of data: AA for amino acid, NT for nucleotide. Parameter required if a mask is provided. The default is ´AA´
`mask`	reading mask. Use this option or 'lin' option. Default c(2,1,2).
`seed`	provide, if necessary, a seed to generate the matrix. The default is 647474747

Value

An orthonormal matrix (basis) whose dimensions correspond to the given mask to be used and a desired projection size (length of the output vector). The basis must be supplied to the function SWeeP (see examples).

'orthBase' returns a 'list' containing:

- mat: the orthonormal matrix (basis)
- seed: the random seed (metadata to identify the matrix)
- version: the rSWeeP version

Author(s)

Camila P. Perico

See Also

SWeeP

Examples

```
# define the mask - determines the length of input vector (20^4 = 160000)
mask <- c(2,1,2)

# define the length of output vector
psz <- 600

# get the basis matrix to projection
Mybase <- orthBase(mask = mask, col = psz,seqtype='AA')
```

PCCI *PhyloTaxonomic Consistency Cophenetic Index*

Description

Phylogenetic tree evaluation function, estimate of how grouped the samples of the same taxon are in the phylogenetic tree.

Usage

```
PCCI(tr, mt = NULL)
```

Arguments

tr	Phylogenetic tree. If the tree contains sample names in the labels, provide metadata. If it already contains the names of the taxa, just provide the tree.
mt	Metadata. The metadata should have the following format: the first column should contain the names of the samples, exactly as they appear on the tree label; the second column should contain the corresponding taxa. If the tree already has the labels renamed according to the taxon, it is not necessary to provide metadata.

Details

Empty or NA labels are removed from analyses

Value

The PCCI index for each taxon and the mean value

'PCCI' returns a 'list' containing:

- tab: the PCCI value for each taxon in a two-colunm output: taxa and cost
- mean: the mean value of PCCI metric

Author(s)

Camila P. Perico

Examples

```
# Load the sample tree and its metadata
pathtree <- system.file(package = "rSWeeP" , "examples" , "tree_Mitochondrial.tree")
tree = ape::read.tree(pathtree)
pathmetadata <- system.file(package = "rSWeeP" , "examples" , "metadata_mitochondrial.csv")
mt = read.csv(pathmetadata,header=TRUE)

data = data.frame(sp=mt$fileName,family=mt$family)
PCCI(tree,data)
```

PMPG *Percentage of Mono or Paraphyletic Groups*

Description

Phylogenetic tree evaluation function, returns the percentage of Monophyletic and Paraphyletic taxa in the phylogenetic tree.

Usage

```
PMPG(tr, mt = NULL)
```

Arguments

tr	Phylogenetic tree. If the tree contains sample names in the labels, provide metadata. If it already contains the names of the taxa, just provide the tree.
mt	Metadata. The metadata should have the following format: the first column should contain the names of the samples, exactly as they appear on the tree label; the second column should contain the corresponding taxa. If the tree already has the labels renamed according to the taxon, it is not necessary to provide metadata.

Details

Empty or NA labels are removed from analyses

Value

The ‘PMPG’ returns a ‘list‘ containing:

- tab: a dataframe with a three-colunm output: taxa, mono and para. ‘mono‘ and ‘para‘ columns returns a boolean value.
- percMono: percentage of Monophyletic taxa
- percPara: percentage of Paraphyletic taxa
- mean: the mean value of ‘percMono‘ and ‘percPara‘

Author(s)

Camila P. Perico

Examples

```
# Load the sample tree and its metadata
pathtree <- system.file(package = "rSWeeP" , "examples" , "tree_Mitochondrial.tree")
tree = ape::read.tree(pathtree)
pathmetadata <- system.file(package = "rSWeeP" , "examples" , "metadata_mitochondrial.csv")
mt = read.csv(pathmetadata,header=TRUE)
```

```
data = data.frame(sp=mt$fileName,family=mt$family)
PMPG(tree,data)
```

SWeeP *Spaced Words Projection*

Description

Spaced Words Projection version (SWeeP) is an alignment-free method for the vector representation of the biological sequences (amino acid and nucleotide). The ´SWeeP´ is an R implementation of the SWeeP method (De Pierri, 2020). Each sequence provided is represented by a compact numerical vector which is easy to analyze. The method is based on k-mers counting and random projection. For the analysis of biological sequences, this function requires you to supply the orthonormal matrix, which can be obtained by the 'orthBase' function as in the example. Details of the methodology can be found in the reference (De Pierri, 2020). The function allows general dimensionality reduction of RNAseq data and generic matrices.

Usage

```
SWeeP(input, orthbase, bin = FALSE, ...)

## S4 method for signature 'AAStringSet'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  mask = NULL,
  seqtype = NULL,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'DNAStringSet'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  mask = NULL,
  seqtype = NULL,
  concatenate = FALSE,
  verbose = TRUE
```

```
)

## S4 method for signature 'RNAStringSet'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  mask = NULL,
  seqtype = NULL,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'BStringSet'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  mask = NULL,
  seqtype = NULL,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'BString'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  mask = NULL,
  seqtype = NULL,
  concatenate = FALSE,
  verbose = TRUE
)

## S4 method for signature 'character'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  norm = "none",
  ncores = NULL,
```

```
  extension = "",
  mask = NULL,
  seqtype = "AA",
  lowRAMmode = FALSE,
  verbose = TRUE
)

## S4 method for signature 'array'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  transpose = FALSE,
  RNAseqdata = FALSE,
  norm = "none",
  verbose = TRUE
)

## S4 method for signature 'integer'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  transpose = FALSE,
  RNAseqdata = FALSE,
  norm = "none",
  verbose = TRUE
)

## S4 method for signature 'matrix'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  transpose = FALSE,
  RNAseqdata = FALSE,
  norm = "none",
  verbose = TRUE
)

## S4 method for signature 'dgCMatrix'
SWeeP(
  input,
  orthbase,
  bin = FALSE,
  transpose = FALSE,
  RNAseqdata = FALSE,
  norm = "none",
```

```
  verbose = TRUE
)
```

Arguments

`input`	There are four input formats available: (a) 'BStringSet' (variants: 'AAStringSet', 'RNAStringSet', 'DNAStringSet'). Biological sequence format loaded in memory; (b) 'character'. String containing a path to a folder with FASTA files; (c) 'dgCMatrix'. Expression matrix loaded with Seurat package (mtx pattern). (d) 'matrix' (variants: 'array','integer'). Generic matrix.
`orthbase`	the orthonormal projection matrix generated by the orthBase() function.
`bin`	binary mode (TRUE), or count mode (FALSE) for HDV construction. Default is FALSE.
`...`	other arguments of the function itself
`ncores`	Number of CPU cores used for parallel processing. Default is 2.
`norm`	normalization of HDV. This must be one of 'none', 'log' or 'logNeg'. 'none' is no normalization, 'log' is simple logarithm, ´Neg´ to convert nulls into -1, ´logNeg´ option is indicated for analyzing genes and short sequences. Default is ´none´.
`mask`	reading mask. Available only for inputs in biological sequence format or path for FASTA files. Default for amino acids is 'c(2,1,2)' and for nucleotides c(5,5,5)
`seqtype`	type of data: ´AA´ for amino acid, ´NT´ for nucleotide. Available only for inputs in biological sequence format or path for FASTA files. The default is AA
`concatenate`	defines whether to treat each sequence individually or to concatenate them into a single sequence. Available only for inputs in biological sequence format. The default is FALSE.
`verbose`	verbose mode. The default is TRUE
`extension`	extension of files desired to concatenate (Optional). Available only for input type path to folder with FASTA files.
`lowRAMmode`	lowRAMmode is suitable for reading large files individually, such as complete genomes, when the machine's memory is limited. read one FASTA at a time, recommended for large files such as complete eukaryotic genomes or proteomes. The default is FALSE
`transpose`	If the rows correspond to the samples and the columns correspond to the genes (mtx pattern), use transpose=FALSE. If the columns correspond to the samples, use transpose=TRUE. Available only for inputs of the expression matrix or generic matrix type. The default setting is FALSE
`RNAseqdata`	For RNAseq data use 'TRUE' or apply the parameter 'transpose=TRUE'. Default is FALSE.

Details

The normalization option 'logNeg' applies a simple logarithm to the HDV matrix. Its difference from 'log' is the conversion of zeros to -1 in HDV.

Value

'SWeeP' returns a 'list' containing the following components:

- proj: a 'numeric' matrix with 'm' columns and one line per sequence, each row corresponding to a compact vector
- info: aditional information of the process. This object is subdivided in:
  - ProjectionSize: a 'integer' corresponding to 'psz'
  - bin: bin: a 'boolean' containing if binary (TRUE) or counting (FALSE)
  - mask: a 'vector' containing the mask used
  - SequenceType: a 'character' containing the type of the sequence (amino acid: AA, ou nucleotide: NT)
  - concatenate : a 'boolean' corresponding to the concatenation of sequences
  - version : a 'character' corresponding to the version of the package
  - norm : a 'character' containing the normalization used
  - extension: a 'character' containing the list of extensions considered
  - timeElapsed: a 'double' containing the elapsed time in seconds
  - headers : 'list' of headers for each analyzed sequence

Author(s)

Camila P. Perico

References

De Pierri, C. R., et al. (2020). SWeeP: representing large biological sequences datasets in compact vectors. Scientific reports, 10(1):1–10.

Examples

```
# get the path to the folder containing the FASTA files
path = paste (system.file("examples/aaMitochondrial/",package = "rSWeeP"),'/', sep = '')

# define the parameters
mask = c(2,1,2)
psz = 500

# get the basis matrix to projection
base160k = orthBase(160000,psz)

# get the vectors that represent the sequences
LDV = SWeeP(input=path,orthbase=base160k,extension=c('.faa','.fas','.fasta'),
            mask=mask,seqtype='AA',ncores=2)
```

SWeePlite *Spaced Words Projection lite*

Description

Spaced Words Projection version lite (SWeePlite) is an alignment-free method for the vector representation of the biological sequences (amino acid and nucleotide). Analogous to the ´SWeeP´ function (De Pierri, 2020), ´SWeePlite´ has optimizations in its implementation that allow the use of larger read masks with low RAM consumption. It also eliminates the need to supply the orthonormal matrix (it is generated internally). Each sequence provided is represented by a compact numerical vector which is easy to analyze. The method is based on k-mers counting and random projection. Details of the methodology can be found in the reference (De Pierri, 2020). The function allows general dimensionality reduction of RNAseq data and generic matrices.

Usage

```
SWeePlite(input, psz, bin = FALSE, ncores = NULL, ...)

## S4 method for signature 'AAStringSet'
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  concatenate = FALSE,
  mask = NULL,
  seqtype = NULL,
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'DNAStringSet'
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  concatenate = FALSE,
  mask = NULL,
  seqtype = NULL,
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'RNAStringSet'
```

```
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  concatenate = FALSE,
  mask = NULL,
  seqtype = NULL,
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'BStringSet'
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  concatenate = FALSE,
  mask = NULL,
  seqtype = NULL,
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'BString'
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
  concatenate = FALSE,
  mask = NULL,
  seqtype = NULL,
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'character'
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  norm = "none",
```

```
  mask = NULL,
  extension = "",
  seqtype = "AA",
  lowRAMmode = TRUE,
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'array'
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  transpose = FALSE,
  RNAseqdata = FALSE,
  norm = "none",
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'integer'
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  transpose = FALSE,
  RNAseqdata = FALSE,
  norm = "none",
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'matrix'
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  transpose = FALSE,
  RNAseqdata = FALSE,
  norm = "none",
  nk = 15000,
  verbose = TRUE
)

## S4 method for signature 'dgCMatrix'
```

```
SWeePlite(
  input,
  psz = 1369,
  bin = FALSE,
  ncores = NULL,
  transpose = FALSE,
  RNAseqdata = FALSE,
  norm = "none",
  nk = 15000,
  verbose = TRUE
)
```

Arguments

input	There are four input formats available: (a) 'BStringSet' (variants: 'AAStringSet', 'RNAStringSet', 'DNAStringSet'). Biological sequence format loaded in memory; (b) 'character' String containing a path to a folder with FASTA files; (c) 'dgCMatrix' Expression matrix loaded with Seurat package (mtx pattern). (d) 'matrix' (variants: 'array','integer'). Generic matrix.
psz	projection size. Default 1369
bin	binary mode (TRUE), or counting mode (FALSE) for HDV construction. Default is FALSE.
ncores	Number of CPU cores used for parallel processing. Default is 2.
...	other arguments of the function itself
norm	normalization of HDV. This must be one of 'none', 'log' or 'logNeg'. 'none' is no normalization, 'log' is simple logarithm, ´Neg´ to convert nulls into -1, ´logNeg´ option is indicated for analyzing genes and short sequences. Default is ´none´.
concatenate	defines whether to treat each sequence individually or to concatenate them into a single sequence Available only for inputs in biological sequence format. The default is FALSE.
mask	reading mask. Available only for inputs in biological sequence format or path for FASTA files. Default c(2,1,2)
seqtype	type of data: ´AA´ for amino acid, ´NT´ for nucleotide. Available only for inputs in biological sequence format or path for FASTA files. The default is AA
nk	Step size of HDV for parallel loop. Default is 50000.
verbose	verbose mode. The default is TRUE
extension	extension of files desired to concatenate (Optional). Available only for input type path to folder with FASTA files.
lowRAMmode	lowRAMmode is suitable for reading large files individually, such as complete genomes, when the machine's memory is limited. read one FASTA at a time, recommended for large files such as complete eukaryotic genomes or proteomes. The default is FALSE

`transpose` If the rows correspond to the samples and the columns correspond to the genes (mtx pattern), use transpose=FALSE. If the columns correspond to the samples, use transpose=TRUE. Available only for inputs of the expression matrix or generic matrix type. The default setting is FALSE

`RNAseqdata` For RNAseq data use 'TRUE' or apply the parameter 'transpose=TRUE'. Default is FALSE.

Details

The normalization option 'logNeg' applies a simple logarithm to the HDV matrix. Its difference from 'log' is the conversion of zeros to -1 in HDV.

Value

'SWeePlite' returns a 'list' containing the following components:

- proj: a 'numeric' matrix with 'm' columns and one line per sequence, each row corresponding to a compact vector
- info: aditional information of the process. This object is subdivided in:
  - ProjectionSize: a 'integer' corresponding to 'psz'
  - bin: bin: a 'boolean' containing if binary (TRUE) or counting (FALSE)
  - mask: a 'vector' containing the mask used
  - SequenceType: a 'character' containing the type of the sequence (amino acid: AA, ou nucleotide: NT)
  - concatenate : a 'boolean' corresponding to the concatenation of sequences
  - version : a 'character' corresponding to the version of the package
  - norm : a 'character' containing the normalization used
  - extension: a 'character' containing the list of extensions considered
  - timeElapsed: a 'double' containing the elapsed time in seconds
  - headers : list of headers for each analyzed sequence

Author(s)

Camila P. Perico

References

De Pierri, C. R., et al. (2020). SWeeP: representing large biological sequences datasets in compact vectors. Scientific reports, 10(1):1–10.

Examples

```
# get the path to the folder containing the FASTA files
path = paste (system.file("examples/aaMitochondrial/",package = "rSWeeP"),'/', sep = '')

# define the parameters
mask = c(2,1,2)
psz = 1369
```

```
# get the vectors that represent the sequences
LDV = SWePlite(input=path,extension=c('.faa','.fas','.fasta'),
                psz = psz,mask=mask,bin=FALSE,seqtype='AA',ncores=2)
```

Index