Package 'seqArchR'

January 24, 2025

Type Package

Title Identify Different Architectures of Sequence Elements

Version 1.10.0

Description seqArchR enables unsupervised discovery of _de novo_ clusters with characteristic sequence architectures characterized by position-specific motifs or composition of stretches of nucleotides, e.g., CG-richness. seqArchR does _not_ require any specifications w.r.t. the number of clusters, the length of any individual motifs, or the distance between motifs if and when they occur in pairs/groups; it directly detects them from the data. seqArchR uses non-negative matrix factorization (NMF) as its backbone, and employs a chunking-based iterative procedure that enables processing of large sequence collections efficiently. Wrapper functions are provided for visualizing cluster architectures as sequence logos.

License GPL-3 | file LICENSE

URL https://snikumbh.github.io/seqArchR/, https://github.com/snikumbh/seqArchR

BugReports https://github.com/snikumbh/seqArchR/issues

SystemRequirements Python (>= 3.5), scikit-learn (>= 0.21.2), packaging

Depends R (>= 4.2.0)

Imports utils, graphics, cvTools (>= 0.3.2), MASS, Matrix, methods, stats, cluster, matrixStats, fpc, cli, prettyunits, reshape2 (>= 1.4.3), reticulate (>= 1.22), BiocParallel, Biostrings, grDevices, ggplot2 (>= 3.1.1), ggseqlogo (>= 0.1)

Suggests cowplot, hopach (>= 2.42.0), BiocStyle, knitr (>= 1.22), rmarkdown (>= 1.12), testthat (>= 3.0.2), covr, vdiffr (>= 0.3.0)

VignetteBuilder knitr

biocViews MotifDiscovery, GeneRegulation, MathematicalBiology, SystemsBiology, Transcriptomics, Genetics, Clustering, DimensionReduction, FeatureExtraction, DNASeq

Encoding UTF-8

LazyData false

RoxygenNote 7.2.1

git_url https://git.bioconductor.org/packages/seqArchR

git_branch RELEASE_3_20

git_last_commit c6daa9f

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2025-01-23

Author Sarvesh Nikumbh [aut, cre, cph]
(<https://orcid.org/0000-0003-3163-4447>)

Maintainer Sarvesh Nikumbh <sarvesh.nikumbh@gmail.com>

Contents

`collate_clusters` *Collate sequence IDs from an existing clustering according to a new, given clustering of the existing clusters*

Description

Collate sequences original divided across n clusters into a new set of m clusters. These ‘m‘ clusters obtained by clustering the original ‘n‘ clusters. Assume a collection of 100 sequences across seven existing clusters. These seven clusters are collated to obtain three resulting clusters. Collating 100 sequences distributed across the seven clusters into the resulting three clusters can be achieved with collate clusters.

Usage

```
collate_clusters(to_clust, orig_clust)
```

Arguments

to_clust	A list giving clustering of factors. In other words this is the clustering of clusters of sequences given in 'orig_clust'
orig_clust	A list of sequence IDs in existing clusters

Value

A list with sequence IDs collated by the specified clustering

Examples

```
set.seed(123)
n <- 7; nn <- 100
orig_clust_labels <- ceiling(n * runif(nn))
orig_clust <- seqArchR::get_seqs_clust_list(orig_clust_labels)

to_clust <- list(c(1,4), c(2,3,5), c(6,7))

collate_clusters(to_clust = to_clust, orig_clust = orig_clust)
```

---

collate_seqArchR_result

Collate raw clusters at the chosen iteration of seqArchR result

---

Description

We use hierarchical clustering for reordering/collating raw clusters from seqArchR's given iteration.

Usage

```
collate_seqArchR_result(
  result,
  iter = length(result$seqsClustLabels),
  clust_method = "hc",
  aggl_method = "ward.D",
  dist_method = "euclid",
  regularize = FALSE,
  topn = 50,
  collate = TRUE,
  return_order = FALSE,
  flags = list(debugFlag = FALSE, verboseFlag = TRUE),
  ...
)
```

Arguments

result	The seqArchR result object.
iter	Specify clusters at which iteration of seqArchR are to be reordered/collated. Default is the last iteration of the seqArchR result object.
clust_method	Specify 'hc' for hierarchical clustering. Currently, only hierarchical clustering is supported.
aggl_method	One of linkage values as specified for hierarchical clustering with hclust. Default is 'ward.D'.
dist_method	Distance measure to be used with hierarchical clustering. Available options are "euclid" (default), "cor" for correlation, "cosangle" for cosine angle.
regularize	Logical. Specify TRUE if regularization is to be performed before comparison. Default is FALSE. Also see argument 'topN'.
topn	Use only the top N dimensions of each basis vector for comparing them. Note that since each basis vector has 4L or 16L (mono- or dinucleotides) dimensions, each dimension is a combination of nucleotide and its position in the sequence. This argument selects the top N dimensions of the basis vector. This is ignored when argument 'regularize' is FALSE.
collate	Logical. Specify TRUE if collation using hierarchical agglomerative clustering is to be performed, otherwise FALSE.
return_order	Logical. Use this argument when you want hierarchical clustering to be performed but not collation of clusters. Therefore, setting return_order to TRUE will return the hierarchical clustering object itself. This enables custom downstream processing/analysis.
flags	Pass the flags object similar to the flags in configuration of the seqArchR result object.
...	ignored

Value

When 'collate' is TRUE, a list with the following elements is returned:

basisVectorsCLust A list storing collation information of the basis vectors, i.e, IDs of basis vectors that were collated into one.

clusters A list of sequences in each collated cluster.

seqClustLabels Cluster labels for all sequences according to the collated clustering.

When 'collate' is FALSE, it returns the already existing basis vectors, each as singleton clusters. The sequence cluster labels and sequence clusters are also handled accordingly. All are available as part of the same list as the earlier case.

When 'return_order' is set to TRUE, the hierarchical clustering result is returned instead.

Examples

```
res <- readRDS(system.file("extdata", "example_seqArchRresult.rds",
        package = "seqArchR", mustWork = TRUE))

# While the default settings for collation use Euclidean distance and
# ward.D agglomeration, one can choose to use different settings, say,
# correlation distance and complete linkage, and also regularizing to use
# only top 50 dimensions (nucleotide-positions combinations)
```

```
collated_res <- collate_seqArchR_result(result = res, iter = 2,
                        aggl_method = "complete", dist_method = "cor",
                        regularize = TRUE, topn = 50)

names(collated_res)
```

`get_clBasVec` *Get functions for seqArchR result object*

Description

Basis vectors' information for the selected iteration.

Usage

```
get_clBasVec(res, iter)

get_clBasVec_k(res, iter)

get_clBasVec_m(res, iter)

get_seqClLab(res, iter = NULL)
```

Arguments

`res`	seqArchR result object.
`iter`	Choose the iteration of seqArchR result to get from.

Value

get_clBasVec A list with two elements 'nBasisVectors' (integer) and 'basisVectors' (matrix).

get_clBasVec_k The number of basis vectors (integer).

get_clBasVec_m The basis vectors' matrix with features along the rows of the matrix.

get_seqClLab A character vector denoting the cluster IDs for each sequence.

Functions

- `get_clBasVec_k`: Get the number of basis vectors (clusters) at the selected iteration.
- `get_clBasVec_m`: The basis vectors matrix at the selected iteration. Note that eatures along rows.
- `get_seqClLab`: Get the cluster IDs for each sequence. Note that order of sequences here is as per the input.

See Also

`seqs_str`

Examples

```
res <- readRDS(system.file("extdata", "example_seqArchRresult.rds",
         package = "seqArchR", mustWork = TRUE))

k <- get_clBasVec_k(res=res, iter=2)

bMat <- get_clBasVec_m(res=res, iter=2)

## cluster labels of sequences from final clustering
scLab <- get_seqClLab(res=res, iter=2)
```

---

get_one_hot_encoded_seqs

Get one-hot encoded sequences

Description

Get the one-hot encoding representation of the given sequences.

Usage

```
get_one_hot_encoded_seqs(seqs, sinuc_or_dinuc = "sinuc")
```

Arguments

seqs	A DNAStringSet object holding the given DNA sequences
sinuc_or_dinuc	character string, 'sinuc' or 'dinuc' to select for mono- or dinucleotide profiles.

Value

A sparse matrix of sequences represented with one-hot-encoding

See Also

prepare_data_from_FASTA for generating one-hot encoding of sequences from a FASTA file

Other input functions: prepare_data_from_FASTA()

Examples

```
fname <- system.file("extdata", "example_data.fa.gz",
                        package = "seqArchR", mustWork = TRUE)


rawSeqs <- prepare_data_from_FASTA(fasta_fname = fname,
                        raw_seq = TRUE)

seqs_dinuc <- get_one_hot_encoded_seqs(seqs = rawSeqs,
                                       sinuc_or_dinuc = "dinuc")
```

get_seqs_clust_list *Retrieve sequence clusters as a list from the sequence labels*

Description

Given the sequence cluster labels from the seqArchR result object, returns the clusters separated as a list.

Usage

```
get_seqs_clust_list(seqs_clust_lab)
```

Arguments

seqs_clust_lab	Sequences with cluster labels as in the seqArchR result object.

Value

A list holding sequence IDs belonging in each cluster.

Examples

```
clustLabels <- sample(seq_len(4), 50, replace = TRUE)
print(clustLabels)
get_seqs_clust_list(clustLabels)
```

make_PWMs *Make a PWM-resembling matrix out of a given n-vector*

Description

The given matrix (or simply a vector) is reshaped to have four rows for four nucleotides and a relevant number of columns.

Usage

```
make_PWMs(vec, add_pseudo_counts = TRUE, scale = TRUE, sinuc = TRUE)
```

Arguments

vec	A vector that will be reshaped into a PWM matrix of DNA sequences. Note that the matrix is formed by row.
add_pseudo_counts	Logical, taking values TRUE or FALSE, specifying whether or not pseudo-counts are added to the matrix.
scale	Logical, taking values TRUE or FALSE, specifying whether or not the matrix is scaled column-wise, i.e., all columns summed to 1.
sinuc	Logical. Specify TRUE for mononucleotides (default), FALSE to for dinucleotides.

Value

A PWM. If sinuc is 'TRUE', the PWM has 4 rows corresponding to the 4 nucleotides (A, C, G, T) and the relevant number of columns (i.e., number of elements in given vector/4). If dinucleotide is selected, by setting 'sinuc' to 'FALSE', the PWM has 16 rows corresponding to the dinucleotide combinations of the four nucleotides (A, C, G, T) and the relevant number of columns (i.e., number of elements in given vector/16).

Examples

```
## Mononucleotides case
## Make a dummy PWM of dimensions 4 * 10 from a vector
vec <- runif(4*10)
pwm <- seqArchR::make_PWMs(vec = vec, add_pseudo_counts = FALSE)

## Dinucleotides case
res <- readRDS(system.file("extdata", "example_seqArchRresult.rds",
         package = "seqArchR", mustWork = TRUE))

pwm <- seqArchR::make_PWMs(get_clBasVec_m(res,iter=1)[,1],
                        add_pseudo_counts = FALSE, sinuc = FALSE)
```

---

plot_arch_for_clusters

Plot cluster architectures as sequence logos.

Description

Given a collection of FASTA sequences as a DNAStringSet object, and the clusters information, this function plots the architectures for all clusters. If a name for the PDF file is provided, the resulting set of architecture sequence logos are saved as a multi-page PDF.

Usage

```
plot_arch_for_clusters(
  seqs,
  clust_list,
  pos_lab = NULL,
  xt_freq = 5,
  set_titles = TRUE,
  pdf_width = 11,
  pdf_height = 2,
  pdf_name = NULL,
  show = FALSE,
  ...
)
```

Arguments

seqs	Sequences as a DNAStringSet.
clust_list	Clusters as a list of sequence IDs in each cluster.

pos_lab	Labels for sequence positions, should be of same length as that of the sequences. Default value is NULL, when the positions are labeled from 1 to the length of the sequences.
xt_freq	Frequency of x-axis ticks.
set_titles	Specify TRUE if titles are to be written for the plots. With FALSE, there are no titles for the plots. The title for each plot includes the current cluster number, total number of clusters, start and end sequence numbers in the collection.
pdf_width, pdf_height	Width and height in inches of the PDF file. Default values are 11 and 2.
pdf_name	Specify the PDF filename.
show	Set TRUE if plot should be immediately shown/plotted. Default is TRUE. By setting FALSE, one can simply collect the list of plots and use any other approach to arrange/display them. See examples.
...	Additional args passed to plot_ggseqlogo_of_seqs.

Value

A list of (ggplot2-based) sequence logo plots is returned. When a valid file name is specified, the list of plots is also written to the PDF file (one plot per page).

Examples

```
res <- readRDS(system.file("extdata", "example_seqArchRresult.rds",
         package = "seqArchR", mustWork = TRUE))

# Default position labels 1 to length of the sequences.
# Can also set pos_lab based on biology, e.g., use -50 to 49 denoting
# 50 basepairs upstream and 49 downstream of the transcription start site
# located at position 0.
arch_pl <- plot_arch_for_clusters(seqs = seqs_str(res),
                                  clust_list = res$clustSol$clusters,
                                  pos_lab = NULL,
                                  pdf_name = NULL,
                                  fixed_coord = TRUE)


# Using cowplot::plot_grid
arch_pl <- plot_arch_for_clusters(seqs = seqs_str(res),
                                  clust_list = res$clustSol$clusters,
                                  pos_lab = seq(100),
                                  method = "bits",
                                  pdf_name = NULL, show = FALSE)
cowplot::plot_grid(plotlist = arch_pl, ncol=1)

# Plotting architecture sequence logos with probability instead of
# information content
arch_pl <- plot_arch_for_clusters(seqs = seqs_str(res),
                                  clust_list = res$clustSol$clusters,
                                  pos_lab = seq(100),
                                  method = "prob",
                                  pdf_name = NULL, show = FALSE)
cowplot::plot_grid(plotlist = arch_pl, ncol=1)
```

`plot_ggseqlogo_of_seqs` *Plot sequence logo of a collection of sequences*

Description

A wrapper to ggseqlogo plotting. Given a collection of sequences, this function plots the sequence logo.

Usage

```
plot_ggseqlogo_of_seqs(
  seqs,
  pos_lab = NULL,
  xt_freq = 5,
  method = "bits",
  title = NULL,
  bits_yax = "full",
  fixed_coord = FALSE
)
```

Arguments

`seqs`	Collection of sequences as a `DNAStringSet` object.
`pos_lab`	Labels for sequence positions, should be of same length as that of the sequences. Default value is NULL, when the positions are labeled from 1 to the length of the sequences.
`xt_freq`	Specify the frequency of the x-axis ticks.
`method`	Specify either 'bits' for information content or 'prob' for probability.
`title`	The title for the plot. Deafult is NULL.
`bits_yax`	Specify 'full' if the information content y-axis limits should be 0-2 or 'auto' for a suitable limit. The 'auto' setting adjusts the y-axis limits according to the maximum information content of the sequence logo. Default is 'full'.
`fixed_coord`	Specify TRUE if the aspect ratio of the plot should be fixed, FALSE otherwise. Default is TRUE. When 'method' argument is set to 'bits', ratio is 4, when 'prob', ratio is 6.

Value

A sequence logo plot of the given DNA sequences.

See Also

`plot_arch_for_clusters` for obtaining multiple sequence logo plots as a list.

Examples

```
res <- readRDS(system.file("extdata", "example_seqArchRresult.rds",
         package = "seqArchR", mustWork = TRUE))

# Default, using information content on y-axis
pl <- plot_ggseqlogo_of_seqs(seqs = seqs_str(res, iter=1, cl=3),
                             pos_lab = seq_len(100), title = NULL,
                             fixed_coord = TRUE)
pl

# Using probability instead of information content
pl <- plot_ggseqlogo_of_seqs(seqs = seqs_str(res, iter=1, cl=3),
                             pos_lab = seq_len(100), title = "",
                             method = "prob", fixed_coord = TRUE)
pl
```

`prepare_data_from_FASTA` *Generate one-hot encoding of sequences given as FASTA file*

Description

Given a set of sequences in a FASTA file this function returns a sparse matrix with one-hot encoded sequences. In this matrix, the sequence features are along rows, and sequences along columns. Currently, mono- and dinucleotide features for DNA sequences are supported. Therefore, the length of the feature vector is 4 and 16 times the length of the sequences (since the DNA alphabet is four characters) for mono- and dinucleotide features respectively.

Usage

```
prepare_data_from_FASTA(fasta_fname, raw_seq = FALSE, sinuc_or_dinuc = "sinuc")
```

Arguments

`fasta_fname`	Provide the name (with complete path) of the input FASTA file.
`raw_seq`	TRUE or FALSE, set this to TRUE if you want the raw sequences.
`sinuc_or_dinuc`	character string, 'sinuc' or 'dinuc' to select for mono- or dinucleotide profiles.

Value

A sparse matrix of sequences represented with one-hot-encoding.

See Also

`get_one_hot_encoded_seqs` for directly using a DNAStringSet object

Other input functions: `get_one_hot_encoded_seqs()`

Examples

```
fname <- system.file("extdata", "example_data.fa.gz",
                        package = "seqArchR", mustWork = TRUE)

# mononucleotides feature matrix
rawSeqs <- prepare_data_from_FASTA(fasta_fname = fname,
                        sinuc_or_dinuc = "sinuc")

# dinucleotides feature matrix
rawSeqs <- prepare_data_from_FASTA(fasta_fname = fname,
                        sinuc_or_dinuc = "dinuc")

# FASTA sequences as a Biostrings::DNAStringSet object
rawSeqs <- prepare_data_from_FASTA(fasta_fname = fname,
                        raw_seq = TRUE)
```

`seqArchR` *seqArchR: A package for de novo discovery of different sequence architectures*

Description

Given a set of DNA sequences, `seqArchR` enables unsupervised discovery of _de novo_ clusters with characteristic sequence architectures characterized by position-specific motifs or composition of stretches of nucleotides, e.g., CG-richness, etc.

Call this function to process a data set using seqArchR.

Usage

```
seqArchR(
  config,
  seqs_ohe_mat,
  seqs_raw,
  seqs_pos = NULL,
  total_itr = NULL,
  set_ocollation = NULL,
  fresh = TRUE,
  use_oc = NULL,
  o_dir = NULL
)
```

Arguments

`config`	seqArchR configuration object as returned by `set_config`. This is a required argument.
`seqs_ohe_mat`	A matrix of one-hot encoded sequences with sequences along columns. This is a required argument.
`seqs_raw`	A `DNAStringSet` object. The FASTA sequences as a DNAStringSet object. This argument required argument.
`seqs_pos`	Vector. Specify the tick labels for sequence positions. Default is NULL.

`total_itr` Numeric. Specify the number of iterations to perform. This should be greater than zero. Default is NULL.

`set_ocollation` Logical vector. A logical vector of length 'total_itr' specifying for every iteration of seqArchR if collation of clusters from outer chunks should be performed. TRUE denotes clusters are collated, FALSE otherwise.

`fresh` Logical. Specify if this is (not) a fresh run. Because seqArchR enables checkpointing, it is possible to perform additional iterations upon clusters from an existing seqArchR result (or a checkpoint) object. See 'use_oc' argument. For example, when processing a set of FASTA sequences, if an earlier call to seqArchR performed two iterations, and now you wish to perform a third, the arguments 'fresh' and 'use_oc' can be used. Simply set 'fresh' to FALSE and assign the sequence clusters from iteration two from the earlier result to 'use_oc'. As of v0.1.3, with this setting, seqArchR returns a new result object as if the additional iteration performed is the only iteration.

`use_oc` List. Clusters to be further processed with seqArchR. These can be from a previous seqArchR result (in which case use `get_seqs_clust_list` function), or simply clusters from any other method. Warning: This has not been rigorously tested yet (v0.1.3).

`o_dir` Character. Specify the output directory with its path. seqArchR will create this directory. If a directory with the given name exists at the given location, seqArchR will add a suffix to the directory name. This change is reported to the user. Default is NULL. When NULL, just the result is returned, and no plots or checkpoints or result is written to disk.

Details

The seqArchR package provides three categories of important functions: related to data preparation and manipulation, performing non-negative matrix factorization, performing clustering, and visualization-related functions.

Value

A nested list of elements as follows:

seqsClustLabels A list with cluster labels for all sequences per iteration of seqArchR. The cluster labels as stored as characters.

clustBasisVectors A list with information on NMF basis vectors per iteration of seqArchR. Per iteration, there are two variables 'nBasisVectors' storing the number of basis vectors after model selection, and 'basisVectors', a matrix storing the basis vectors themselves. Dimensions of the 'basisVectors' matrix are 4*L x nBasisVectors (mononucleotide case) or 16*L x nBasisVectors (dinucleotide case).

clustSol The clustering solution obtained upon processing the raw clusters from the last iteration of seqArchR's result. This is handled internally by the function `collate_seqArchR_result` using the default setting of Euclidean distance and ward.D linkage hierarchical clustering.

rawSeqs The input sequences as a DNAStringSet object.

timeInfo Stores the time taken (in minutes) for processing each iteration. This element is added only if 'time' flag is set to TRUE in config.

config The configuration used for processing.

call The function call itself.

Functions for data preparation and manipulation

- prepare_data_from_FASTA
- get_one_hot_encoded_seqs

Functions for visualizations

- plot_arch_for_clusters
- plot_ggseqlogo_of_seqs
- viz_bas_vec
- viz_seqs_acgt_mat
- viz_pwm

Examples

```
# Here,we re-use the example input sequences and one-hot encoded matrix
# shipped with seqArchR. Please see examples in the corresponding man pages
# for generating a one-hot encoded input matrix from raw FASTA sequences
# in `prepare_data_from_FASTA`
#
inputSeqsMat <- readRDS(system.file("extdata", "tssSinuc.rds",
                            package = "seqArchR", mustWork = TRUE))

inputSeqsRaw <- readRDS(system.file("extdata", "tssSeqsRaw.rds",
                            package = "seqArchR", mustWork = TRUE))

# Set seqArchR configuration
seqArchRconfig <- seqArchR::set_config(
    parallelize = TRUE,
    n_cores = 2,
    n_runs = 100,
    k_min = 1,
    k_max = 20,
    mod_sel_type = "stability",
    bound = 10^-8,
    chunk_size = 100,
    flags = list(debug = FALSE, time = TRUE, verbose = TRUE,
        plot = FALSE)
)

# Run seqArchR
seqArchRresult <- seqArchR::seqArchR(config = seqArchRconfig,
                          seqs_ohe_mat = inputSeqsMat,
                          seqs_raw = inputSeqsRaw,
                          seqs_pos = seq(1,100,by=1),
                          total_itr = 2,
                          set_ocollation = c(TRUE, FALSE))
```

seqs_str *Get sequences from the seqArchR result object*

Description

Wrapper to fetch sequences from the seqArchR result object as character

Usage

```
seqs_str(res, iter = NULL, cl = NULL, ord = FALSE)
```

Arguments

res	seqArchR result object
iter	Specify the iteration of seqArchR result. If set to NULL (the default), the original set of sequences (‘seqArchRresult$rawSeqs‘) is returned.
cl	Specify the cluster number. Sequences belonging to this cluster in iteration ‘iter‘ of seqArchR result are returned as character. When ‘iter‘ is NULL, this is treated as denoting the cluster number in seqArchR’s final clustering solution (‘seqArchRresult$clustSol$clusters‘).
ord	Specify TRUE if sequences are ordered by clusters. The original ordering of the sequences can be fetched by setting ‘iter‘ to NULL and ‘ord‘ to FALSE.

Details

Setting iter to NULL will fetch sequences as per the final clustering solution of seqArchR (‘clustSol$clusters‘). When ‘iter‘ is not NULL, use ‘cl‘ to further choose a particular cluster. When ‘cl‘ is NULL, the set of sequences returned can be ordered by clusters with ‘ord = TRUE‘. Using ‘ord = FALSE‘ fetches the sequences by their original order.

Value

The selected DNA sequences from the DNAStringSet object as a character vector.

Examples

```
res <- system.file("extdata", "example_seqArchRresult.rds",
         package = "seqArchR", mustWork = TRUE)

# Fetch sequences from 2nd cluster of seqArchR's final solution
ans <- seqArchR::seqs_str(readRDS(res), iter=NULL, cl=2)

# Fetch all sequences ordered by the final clustering
ans <- seqArchR::seqs_str(readRDS(res), iter=NULL, cl=NULL, ord=TRUE)

# Fetch sequences belonging to first cluster in seqArchR's first iteration
ans <- seqArchR::seqs_str(readRDS(res), iter=1, cl=1)
```

set_config *Set seqArchR run configuration*

Description

This function sets the configuration for 'seqArchR'.

Usage

```
set_config(
  chunk_size = 500,
  k_min = 1,
  k_max = 50,
  mod_sel_type = "stability",
  bound = 10^-6,
  cv_folds = 5,
  parallelize = FALSE,
  n_cores = NA,
  n_runs = 100,
  alpha_base = 0,
  alpha_pow = 1,
  min_size = 25,
  result_aggl = "complete",
  result_dist = "euclid",
  checkpointing = TRUE,
  flags = list(debug = FALSE, time = FALSE, verbose = TRUE, plot = FALSE)
)
```

Arguments

chunk_size	Numeric. Specify the size of the inner chunks of sequences.
k_min	Numeric. Specify the minimum of the range of values to be tested for number of NMF basis vectors. Default is 1.
k_max	Numeric. Specify the maximum of the range of values to be tested for number of NMF basis vectors. Default is 50.
mod_sel_type	Character. Specify the model selection strategy to be used. Default is 'stability'. Another option is 'cv', short for cross-validation. Warning: The cross-validation approach can be time consuming and computationally expensive than the stability-based approach.
bound	Numeric. Specify the lower bound value as criterion for choosing the most appropriate number of NMF factors. Default is 1e-08.
cv_folds	Numeric. Specify the number of cross-validation folds used for model selection. Only used when mod_sel_type is set to 'cv'. Default value is 5.
parallelize	Logical. Specify whether to parallelize the procedure. Note that running seqArchR serially can be time consuming, especially when using cross-validation for model selection. See 'n_cores'. Consider parallelizing with at least 2 or 4 cores.
n_cores	The number of cores to be used when 'parallelize' is set to TRUE. If 'parallelize' is FALSE, nCores is ignored.

`n_runs` Numeric. Specify the number of bootstrapped runs to be performed with NMF. Default value is 100. When using cross-validation more than 100 iterations may be needed (upto 500).

`alpha_base, alpha_pow` Specify the base and the power for computing 'alpha' in performing model selection for NMF. alpha = alpha_base^alpha_pow. Alpha specifies the regularization for NMF. Default: 0 and 1 respectively. _Warning_: Currently, not used (for future).

`min_size` Numeric. Specify the minimum number of sequences, such that any cluster/chunk of size less than or equal to it will not be further processed. Default is 25.

`result_aggl` Character. Specify the agglomeration method to be used for final result collation with hierarchical clustering. Default is 'complete' linkage. Possible values are those allowed with `hclust`. Also see details below.

`result_dist` Character. Specify the distance method to be used for final result collation with hierarchical clustering. Default is 'cor' for correlation. Possible values are those allowed with `hclust`. Also see details below.

`checkpointing` Logical. Specify whether to write intermediate checkpoints to disk as RDS files. Checkpoints and the final result are saved to disk provided the 'o_dir' argument is set in `seqArchR`. When 'o_dir' argument is not provided or NULL, this is ignored. Default is TRUE.

`flags` List with four logical elements as detailed.

debug Whether debug information for the run is printed

verbose Whether verbose information for the run is printed

plot Whether verbose plotting is performed for the run

time Whether timing information is printed for the run

Details

Setting suitable values for the following parameters is dependent on the data: 'inner_chunk_size', 'k_min', 'k_max', 'mod_sel_type', 'min_size', 'result_aggl', 'result_dist'.

Value

A list with all params for seqArchR set

Examples

```
# Set seqArchR configuration
seqArchRconfig <- seqArchR::set_config(
    chunk_size = 100,
    parallelize = TRUE,
    n_cores = 2,
    n_runs = 100,
    k_min = 1,
    k_max = 20,
    mod_sel_type = "stability",
    bound = 10^-8,
    flags = list(debug = FALSE, time = TRUE, verbose = TRUE,
        plot = FALSE)
)
```

viz_bas_vec *Visualize the NMF basis vectors*

Description

The given features matrix is visualized as a paired heatmap and sequence logo where the positions are aligned for better visualization., or as a single heatmap or as a single sequence logo.

Usage

```
viz_bas_vec(
  feat_mat,
  ptype = c("heatmap", "seqlogo"),
  method = "bits",
  pos_lab = NULL,
  pdf_name = NULL,
  add_pseudo_counts = FALSE,
  sinuc_or_dinuc = "sinuc",
  fixed_coord = FALSE
)
```

Arguments

feat_mat	The features matrix (basis vectors matrix) from seqArchR.
ptype	Character vector of length one or two. Specify just one of "heatmap" or "seqlogo" to visualize the basis vectors as such, or specify a vector of length two for plotting both, heatmap and seqlogo. These are then arranged one below the other, the first on top and the second under it.
method	Specify either of "custom", "bits", or "probability" for plotting sequence logo. Default is "bits".
pos_lab	Labels for sequence positions, should be of same length as that of the sequences. Default value is NULL, when the positions are labeled from 1 to the length of the sequences.
pdf_name	Filename to save the plot, also provide the extension.
add_pseudo_counts	Logical, taking values TRUE or FALSE, default set to FALSE. Setting it to TRUE will enable adding pseudo-counts to the features matrix.
sinuc_or_dinuc	"sinuc" or "dinuc" for choosing between mono- and dinucleotide profiles respectively.
fixed_coord	Set this to TRUE to use a fixed aspect ratio for the plot irrestive of the width and height of the PDF. Default is FALSE.

Value

nothing

See Also

Other visualization functions: viz_pwm(), viz_seqs_acgt_mat()

Examples

```
res <- readRDS(system.file("extdata", "example_seqArchRresult.rds",
         package = "seqArchR", mustWork = TRUE))

# Visualize basis vectors at iteration 1 of seqArchR result as heatmap and
# sequence logo
viz_bas_vec(feat_mat = get_clBasVec_m(res,iter=1), sinuc_or_dinuc = "dinuc",
                ptype = c("heatmap", "seqlogo"))


# Visualize basis vectors at iteration 1 of seqArchR result as sequence logos
viz_bas_vec(feat_mat = get_clBasVec_m(res,iter=1), ptype = "seqlogo",
                 sinuc_or_dinuc = "dinuc")


# Visualizing basis vector for a single cluster as a heatmap
viz_bas_vec(feat_mat = as.matrix(get_clBasVec_m(res,iter=1)[,3]),
                 ptype = "heatmap", sinuc_or_dinuc = "dinuc")
```

`viz_pwm` *Visualize a position weight matrix as a heatmap or sequence logo*

Description

The given position weight matrix is plotted as a heatmap or sequence logo

Usage

```
viz_pwm(
  pwm_mat,
  method = "heatmap",
  pos_lab = NULL,
  pdf_name = NULL,
  fixed_coord = FALSE,
  bits_yax = "full"
)
```

Arguments

`pwm_mat`	Matrix (usually a PWM, but can be any non-normalized matrix) to be visualized. Rownames must be letters.
`method`	Character. Set this to 'heatmap' when plotting a heatmap, else you can set it to either of 'custom', 'bits', or 'probability' when you wish to visualize it as a sequence logo. Default is 'heatmap'.
`pos_lab`	Labels for sequence positions, should be of same length as that of the sequences. Default value is NULL, when the positions are labeled from 1 to the length of the sequences.
`pdf_name`	Name of the file which will be saved as PDF.
`fixed_coord`	Set this to TRUE to use a fixed aspect ratio for the plot. Default is FALSE.
`bits_yax`	Specify 'full' if the information content y-axis limits should be 0-2 or 'auto' for a suitable limit. The 'auto' setting adjusts the y-axis limits according to the maximum information content of the sequence logo. Default is 'full'.

Value

A ggplot object so you can simply call `print` or `save` on it later. If `pdf_name` is given, it is also saved and the `ggplot2` object returned.

See Also

`plot_ggseqlogo_of_seqs` for visualizing a collection of sequences by their sequence logo.

Other visualization functions: `viz_bas_vec()`, `viz_seqs_acgt_mat()`

Examples

```
res <- readRDS(system.file("extdata", "example_seqArchRresult.rds",
         package = "seqArchR", mustWork = TRUE))

pwm <- seqArchR::make_PWMs(get_clBasVec_m(res,iter=1)[,1],
                        add_pseudo_counts = FALSE, sinuc = FALSE)

viz_pwm(pwm_mat = pwm, method = "heatmap", fixed_coord = TRUE)

viz_pwm(pwm_mat = pwm, method = "bits", fixed_coord = TRUE)
```

---

viz_seqs_acgt_mat *Visualize raw DNA sequences as an image*

Description

This function plots the collection of sequences as an image matrix.

Usage

```
viz_seqs_acgt_mat(
  seqs,
  pos_lab = NULL,
  xt_freq = min(length(pos_lab), 5),
  yt_freq = min(length(seqs), 100),
  use_col = c("darkgreen", "blue", "orange", "red"),
  add_legend = TRUE,
  use_legend = Biostrings::DNA_BASES,
  save_fname = NULL,
  file_type = "PNG",
  f_width = 450,
  f_height = 900,
  f_units = "px"
)
```

Arguments

`seqs`	The sequences as a DNAStringSet object.
`pos_lab`	The labels to be used for the sequence positions. Default: Sequence positions are labeled from 1 to the length of the sequences.

xt_freq	The x-axis tick frequency. Expects a positive integer less than the length of the sequences. Default is 5.
yt_freq	The y-axis tick frequency. Expects a positive integer less than number of sequences. Default is 100.
use_col	A vector of four colors used for the DNA bases A, C, G, and T (in that order).
add_legend	Logical. Whether legend should be added to the plot. Default is TRUE.
use_legend	A character vector of letters in the input sequences. Default is DNA_BASES, used for DNA sequences.
save_fname	Specify the filename (with extension) for saving the plot to disk.
file_type	Specify the file type, namely PNG, JPEG, TIFF.
f_width	Specify the width for the plot. This depends on the length of sequences.
f_height	Specify the height for the plot. This depends on the number of sequences.
f_units	Specify the units in which the height and width are given.

Value

Nothing returned to the R interpreter.

See Also

Other visualization functions: viz_bas_vec(), viz_pwm()

Examples

```
res <- readRDS(system.file("extdata", "example_seqArchRresult.rds",
         package = "seqArchR", mustWork = TRUE))

# Image matrix of sequences in the input order
viz_seqs_acgt_mat(seqs = seqs_str(res))

# Image matrix of sequences ordered by the clustering from seqArchR
use_seqs <- seqs_str(res, iter = NULL, cl = NULL, ord = TRUE)
viz_seqs_acgt_mat(seqs = use_seqs)

# Image matrix of sequences belonging to a single cluster
use_seqs <- seqs_str(res, iter = 2, cl = 2)
viz_seqs_acgt_mat(seqs = use_seqs)
```

Index