

Package: metabinR (via r-universe)

September 27, 2026

Type Package

Title Abundance and Compositional Based Binning of Metagenomes

Version 2.1.1

biocViews Classification, Clustering, Microbiome, Sequencing, Software

Description Provide functions for performing abundance and compositional based binning on metagenomic samples, directly from FASTA or FASTQ files. Functions are implemented in Java and called via rJava. Parallel implementation that operates directly on input FASTA/FASTQ files for fast execution. Inputs may be file paths or Biostrings/ShortRead sequence objects; results are returned as a MetabinResult S4 object wrapping cluster assignments, algorithm parameters, and input metadata.

License GPL-3

Encoding UTF-8

Language en-US

LazyData false

Depends R (>= 4.4), methods

Imports BiocGenerics, BiocParallel, Biostrings, checkmate, cli, rJava, S4Vectors, ShortRead, utils

SystemRequirements Java (>= 11)

RoxygenNote 7.3.3

URL <https://github.com/gkanogiannis/metabinR>

BugReports <https://github.com/gkanogiannis/metabinR/issues>

Suggests BiocStyle, covr, cvms, data.table, dplyr, ggplot2, gridExtra, knitr, R.utils, rmarkdown, sabre, spelling, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/pak/sysreqs

make default-jdk libbz2-dev libjpeg-dev liblzma-dev libpng-dev xz-utils zlib1g-dev

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

Date/Publication 2026-09-27 18:29:26 UTC
RemoteUrl <https://github.com/bioc/metabinR>
RemoteRef HEAD
RemoteSha fa449d025c1422b1912668e85f063cb1f719802e

Contents

abundance_based_binning	2
ambiguous_reads	4
as.data.frame, MetabinResult-method	4
bin_summary	5
composition_based_binning	6
evaluate_bins	7
hierarchical_binning	8
metabinR_jvm_options	10
MetabinResult-accessors	10
MetabinResult-class	11

Index	13
--------------	-----------

abundance_based_binning	
<i>Abundance based binning on metagenomic samples</i>	

Description

This function performs abundance based binning on metagenomic samples, directly from FASTA or FASTQ files, by long kmer analysis (k>8). See [doi:10.1186/s1285901611863](https://doi.org/10.1186/s1285901611863) for more details.

Usage

```
abundance_based_binning(  
  ...,  
  eMin = 1,  
  eMax = 0,  
  kMerSizeAB = 10,  
  numOfClustersAB = 3,  
  outputAB = "AB.cluster",  
  keepQuality = FALSE,  
  dryRun = FALSE,  
  gzip = FALSE,  
  numThreads = BiocParallel::bpworkers()  
)
```

Arguments

...	Input sequences. Either character paths to FASTA/FASTQ files (uncompressed or gzip compressed), a DNAStringSet / QualityScaledDNAStringSet , or a ShortReadQ object. Non-file inputs are staged to a temporary FASTA/FASTQ file for the Java backend.
eMin	Exclude kmers of less or equal count.
eMax	Exclude kmers of more or equal count.
kMerSizeAB	kmer length for Abundance based Binning.
numOfClustersAB	Number of Clusters for Abundance based Binning.
outputAB	Output Abundance based Binning Clusters files location and prefix.
keepQuality	Keep fastq qualities on the output files. (will produce .fastq)
dryRun	Don't write any output files.
gzip	Gzip output files.
numOfThreads	Number of threads to use. Defaults to bpworkers() .

Value

A [MetabinResult](#) object. Its assignments slot is a [DataFrame](#) with numOfClustersAB + 2 columns:

- read_id : read identifier from fasta header
- AB : read was assigned to this AB cluster index
- AB.n : read to cluster AB.n distance

For backwards-compatible data.frame output use `as.data.frame(result)`.

Author(s)

Anestis Gkanogiannis, <anestis@gkanogiannis.com>

References

<https://github.com/gkanogiannis/metabinR>

Examples

```
res <- abundance_based_binning(
  system.file("extdata", "reads.metagenome.fasta.gz", package = "metabinR"),
  dryRun = TRUE, kMerSizeAB = 8
)
res
head(as.data.frame(res))
```

ambiguous_reads	<i>Find reads with close competing bins</i>
-----------------	---

Description

Returns reads for which the second-smallest finite distance is within margin of the smallest distance. In hierarchical results, distances to bins outside the read's parent abundance bin are NA and are ignored. Reads with fewer than two finite distances are omitted. The margin is an absolute difference on the algorithm's distance scale, not a probability.

Usage

```
ambiguous_reads(x, margin = 0.05)
```

Arguments

x	A MetabinResult object.
margin	Nonnegative maximum difference between the two smallest distances. Defaults to 0.05.

Value

A [S4Vectors::DataFrame](#) with read_id, bin, best_distance, second_distance, and distance_margin, in input order.

Examples

```
res <- composition_based_binning(
  system.file("extdata", "reads.metagenome.fasta.gz", package = "metabinR"),
  dryRun = TRUE, kMerSizeCB = 2, numOfClustersCB = 2
)
ambiguous_reads(res, margin = 0.05)
```

as.data.frame, MetabinResult-method	<i>Convert a MetabinResult to a data.frame</i>
-------------------------------------	--

Description

Preserves backwards compatibility with the data.frame return of metabinR <= 1.x.

Usage

```
## S4 method for signature 'MetabinResult'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x` A [MetabinResult](#).
`row.names, optional, ...`
 Unused; retained for S3 signature compatibility.

Value

A base `data.frame` of the assignments.

bin_summary	<i>Summarize the bins in a metabinR result</i>
-------------	--

Description

Counts reads in each observed bin and summarizes their distance to the assigned bin. Distances retain the scale of the binning algorithm; they are not probabilities or comparable across algorithms.

Usage

```
bin_summary(x)
```

Arguments

`x` A [MetabinResult](#) object.

Value

A [S4Vectors::DataFrame](#) with `bin`, `n_reads`, `proportion`, `mean_distance`, and `median_distance`. Only observed bins are included.

Examples

```
res <- composition_based_binning(
  system.file("extdata", "reads.metagenome.fasta.gz", package = "metabinR"),
  dryRun = TRUE, kMerSizeCB = 2, numOfClustersCB = 2
)
bin_summary(res)
```

composition_based_binning

Composition based binning on metagenomic samples

Description

This function performs composition based binning on metagenomic samples, directly from FASTA or FASTQ files, by short kmer analysis ($k < 8$). See [doi:10.1186/s1285901611863](https://doi.org/10.1186/s1285901611863) for more details.

Usage

```
composition_based_binning(
  ...,
  kMerSizeCB = 4,
  numOfClustersCB = 5,
  outputCB = "CB.cluster",
  keepQuality = FALSE,
  dryRun = FALSE,
  gzip = FALSE,
  numOfWorkers = BiocParallel::bpworkers()
)
```

Arguments

...	Input sequences. Either character paths to FASTA/FASTQ files (uncompressed or gzip compressed), a DNAStringSet / QualityScaledDNAStringSet , or a ShortReadQ object. Non-file inputs are staged to a temporary FASTA/FASTQ file for the Java backend.
kMerSizeCB	kmer length for Composition based Binning.
numOfClustersCB	Number of Clusters for Composition based Binning.
outputCB	Output Composition based Binning Clusters files location and prefix.
keepQuality	Keep fastq qualities on the output files. (will produce .fastq)
dryRun	Don't write any output files.
gzip	Gzip output files.
numOfWorkers	Number of threads to use. Defaults to bpworkers() .

Value

A [MetabinResult](#) object. Its assignments slot is a [DataFrame](#) with `numOfClustersCB + 2` columns:

- `read_id`: read identifier from fasta header
- `CB`: read was assigned to this CB cluster index
- `CB.n`: read to cluster CB.n distance

For backwards-compatible data.frame output use `as.data.frame(result)`.

Author(s)

Anestis Gkanogiannis, <anestis@gkanogiannis.com>

References

<https://github.com/gkanogiannis/metabinR>

Examples

```
res <- composition_based_binning(
  system.file("extdata", "reads.metagenome.fasta.gz", package = "metabinR"),
  dryRun = TRUE, kMerSizeCB = 2
)
res
```

evaluate_bins

Evaluate read-level bin assignments against known origins

Description

Joins assignments to a truth table by read identifier, then calculates per-bin purity, per-origin best-bin recovery, and the adjusted Rand index (ARI). These are read-count metrics. They do not measure genome or metagenome-assembled genome completeness, and they are not CAMI/AMBER base-pair-weighted scores. Bin labels are treated as arbitrary identifiers.

Usage

```
evaluate_bins(x, truth, id = "read_id", label = "genome_id")
```

Arguments

x	A MetabinResult object.
truth	A <code>data.frame</code> or S4Vectors::DataFrame containing read IDs and known origins or classes.
id	Name of the read-ID column in truth. The result's <code>read_id</code> column is matched to this column.
label	Name of the truth-label column in truth.

Details

Every result read must have exactly one matching truth row. Extra truth rows are ignored. Duplicate or missing identifiers are errors. If several truth groups or bins tie for the maximum count, the first encountered is reported as dominant.

Value

A list with confusion (bins as rows, truth groups as columns), per_bin (read count, dominant truth group, purity), per_origin (read count, dominant bin, best-bin recovery), and overall (read count, numbers of bins and truth groups, weighted purity, weighted recovery, and ARI). Purity is the dominant truth count divided by bin size; best-bin recovery is the largest count for an origin in any one bin divided by that origin's read count. Both weighted measures sum the relevant dominant counts and divide by the total number of evaluated reads. ARI is NA for fewer than two reads.

Examples

```
x <- composition_based_binning(
  system.file("extdata", "reads.metagenome.fasta.gz", package = "metabinR"),
  dryRun = TRUE, kMerSizeCB = 2, numOfClustersCB = 2
)
truth <- read.delim(system.file(
  "extdata", "reads_mapping.tsv.gz", package = "metabinR"
))
evaluation <- evaluate_bins(x, truth, id = "anonymous_read_id",
  label = "genome_id")
evaluation$overall
```

hierarchical_binning *Hierarchical (ABxCB) binning on metagenomic samples*

Description

This function performs hierarchical binning on metagenomic samples, directly from FASTA or FASTQ files. First it analyzes sequences by long kmer analysis ($k > 8$), as in [abundance_based_binning](#). Then for each AB bin, it guesses the number of composition bins in it and performs composition based binning by short kmer analysis ($k < 8$), as in [composition_based_binning](#). See [doi:10.1186/s1285901611863](https://doi.org/10.1186/s1285901611863) for more details.

Usage

```
hierarchical_binning(
  ...,
  eMin = 1,
  eMax = 0,
  kMerSizeAB = 10,
  kMerSizeCB = 4,
  genomeSize = 3e+06,
  numOfClustersAB = 3,
  outputC = "ABxCB.cluster",
  keepQuality = FALSE,
  dryRun = FALSE,
  gzip = FALSE,
  numOfThreads = BiocParallel::bpworkers()
)
```

Arguments

...	Input sequences. Either character paths to FASTA/FASTQ files (uncompressed or gzip compressed), a DNAStringSet / QualityScaledDNAStringSet , or a ShortReadQ object. Non-file inputs are staged to a temporary FASTA/FASTQ file for the Java backend.
eMin	Exclude kmers of less or equal count.
eMax	Exclude kmers of more or equal count.
kMerSizeAB	kmer length for Abundance based Binning.
kMerSizeCB	kmer length for Composition based Binning.
genomeSize	Average genome size of taxa in the metagenome data.
numOfClustersAB	Number of Clusters for Abundance based Binning.
outputC	Output Hierarchical Binning (ABxCB) Clusters files location and prefix.
keepQuality	Keep fastq qualities on the output files. (will produce .fastq)
dryRun	Don't write any output files.
gzip	Gzip output files.
numOfThreads	Number of threads to use. Defaults to bpworkers() .

Value

A [MetabinResult](#) object. Its assignments slot is a [DataFrame](#):

- read_id : read identifier from fasta header
- ABxCB : read was assigned to this ABxCB cluster index
- ABxCB.n : read to cluster ABxCB.n distance

For backwards-compatible data.frame output use `as.data.frame(result)`.

Author(s)

Anestis Gkanogiannis, <anestis@gkanogiannis.com>

References

<https://github.com/gkanogiannis/metabinR>

Examples

```
res <- hierarchical_binning(
  system.file("extdata", "reads.metagenome.fasta.gz", package = "metabinR"),
  dryRun = TRUE, kMerSizeAB = 4, kMerSizeCB = 2
)
res
```

`metabinR_jvm_options` *JVM options used when metabinR loads*

Description

Returns the vector of JVM flags passed to `.jpackage` on package load. Set `options(metabinR.jvm.flags = c(...))` before loading the package to override; set `options(java.parameters = ...)` to prepend heap-size flags (e.g. `"-Xmx4g"`) in the usual rJava way.

Usage

```
metabinR_jvm_options()
```

Value

A character vector of JVM flags.

Examples

```
metabinR_jvm_options()
```

MetabinResult-accessors
Accessors for [MetabinResult](#)

Description

Accessors for [MetabinResult](#)

Usage

```
assignments(x)

nClusters(x)

parameters(x)

algorithm(x)

## S4 method for signature 'MetabinResult'
assignments(x)

## S4 method for signature 'MetabinResult'
parameters(x)

## S4 method for signature 'MetabinResult'
```

```
algorithm(x)

## S4 method for signature 'MetabinResult'
nClusters(x)
```

Arguments

x A [MetabinResult](#) object.

Value

`assignments()` returns a [DataFrame](#) with the per-read cluster assignments and distances. `nClusters()` returns an integer scalar: the number of clusters inferred by the algorithm. `parameters()` returns the list of arguments passed to the binning function. `algorithm()` returns the algorithm tag ("AB", "CB", or "ABxCB").

Examples

```
res <- abundance_based_binning(
  system.file("extdata", "reads.metagenome.fasta.gz", package = "metabinR"),
  dryRun = TRUE, kMerSizeAB = 4, numOfClustersAB = 2
)
assignments(res)
nClusters(res)
algorithm(res)
parameters(res)$kMerSizeAB
```

MetabinResult-class	<i>MetabinResult: binning result container</i>
---------------------	--

Description

An S4 class returned by [`abundance_based_binning()`], [`composition_based_binning()`] and [`hierarchical_binning()`].

Usage

```
## S4 method for signature 'MetabinResult'
show(object)
```

Arguments

object A [MetabinResult](#).

Value

Objects of this class are returned by the binning functions. Use [assignments](#), [nClusters](#), [parameters](#), [algorithm](#), or `as.data.frame()` to access the results.

Slots

assignments A [DataFrame](#) of cluster assignments. The first column is read_id; subsequent columns are algorithm-specific (see the corresponding binning function).

parameters Named list of the parameters passed to the algorithm.

inputs Character vector of input file paths that were processed.

algorithm Character scalar: one of "AB", "CB", "ABxCB".

Examples

```
res <- abundance_based_binning(  
  system.file("extdata", "reads.metagenome.fasta.gz", package = "metabinR"),  
  dryRun = TRUE, kMerSizeAB = 4, numOfClustersAB = 2  
)  
res  
is(res, "MetabinResult")
```

Index

.jpackage, [10](#)

abundance_based_binning, [2, 8](#)

algorithm, [11](#)

algorithm (MetabinResult-accessors), [10](#)

algorithm, MetabinResult-method
(MetabinResult-accessors), [10](#)

ambiguous_reads, [4](#)

as.data.frame, MetabinResult-method, [4](#)

assignments, [11](#)

assignments (MetabinResult-accessors),
[10](#)

assignments, MetabinResult-method
(MetabinResult-accessors), [10](#)

bin_summary, [5](#)

bpworkers, [3, 6, 9](#)

composition_based_binning, [6, 8](#)

DataFrame, [3, 6, 9, 11, 12](#)

DNAStringSet, [3, 6, 9](#)

evaluate_bins, [7](#)

hierarchical_binning, [8](#)

metabinR_jvm_options, [10](#)

MetabinResult, [3–7, 9–11](#)

MetabinResult (MetabinResult-class), [11](#)

MetabinResult-accessors, [10](#)

MetabinResult-class, [11](#)

nClusters, [11](#)

nClusters (MetabinResult-accessors), [10](#)

nClusters, MetabinResult-method
(MetabinResult-accessors), [10](#)

parameters, [11](#)

parameters (MetabinResult-accessors), [10](#)

parameters, MetabinResult-method
(MetabinResult-accessors), [10](#)

QualityScaledDNAStringSet, [3, 6, 9](#)

S4Vectors::DataFrame, [4, 5, 7](#)

ShortReadQ, [3, 6, 9](#)

show, MetabinResult-method
(MetabinResult-class), [11](#)