Package 'RBPEqBind'

September 22, 2026

Title RNA-Binding Protein Competitive Binding Simulation

Version 0.99.4

Description Simulates competitive binding of N RBPs to RNA sequences using an equilibrium binding model. Supports single sequences, concentration grids, and transcriptome-wide analysis. Designed for Bioconductor.

License GPL-3

Encoding UTF-8

biocViews MathematicalBiology, SystemsBiology, FunctionalGenomics, Transcriptomics, GeneRegulation, Visualization

BugReports https://github.com/S00NYI/RBPEqBind/issues

URL https://github.com/S00NYI/RBPEqBind

Depends R (>= 4.6.0)

Imports Biostrings, data.table, future.apply, GenomicRanges, ggplot2, IRanges, jsonlite, rtracklayer, S4Vectors, stats, SummarizedExperiment, utils

Suggests testthat (>= 3.0.0), knitr, rmarkdown, progressr, BiocStyle, future, vdiffr

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

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

git_branch devel

git_last_commit e1f3433

git_last_commit_date 2026-08-24

Repository Bioconductor 3.24

Date/Publication 2026-09-21

Author Soon Yi [aut, cre] (ORCID: <https://orcid.org/0000-0002-4535-6532>), National Institutes of Health [fnd] (T32 GM152319)

Maintainer Soon Yi <cu.soonyi@gmail.com>

Contents

`callPeaks` *Call Peaks from Binding Simulation*

Description

Identifies peak regions based on binding occupancy. If the transcript headers contain genomic coordinates (e.g. "chr1:1000-2000(+)"), it appends translated genomic coordinate columns (chrom, genomic_start, genomic_end, strand).

Usage

```
callPeaks(
  results,
  rbp_primary,
  rbp_reference = NULL,
  method = c("competitive", "local", "global"),
  threshold = 2,
  min_width = 3,
  window_size = 50
)
```

Arguments

`results`	data.table of simulation results.
`rbp_primary`	Name of the primary RBP.
`rbp_reference`	Name of the reference RBP (for competitive advantage) (default NULL).
`method`	Method: "competitive" (default), "local", or "global".
`threshold`	Threshold (ratio for competitive, occupancy for others) (default 2.0).

min_width	Minimum width of peak to report (default 3).
window_size	Window size for local background (default 50).

Value

A data.table of peaks with start, end, scores, and optionally genomic coordinates.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateBinding("ACGUACGUACGUUUUUACGUACGU", rbp_models, c(HH = 100, HL = 100))
results$transcript <- "test_transcript"
peaks <- callPeaks(results, rbp_primary = "HH", method = "global", threshold = 1.1)
head(peaks)
```

---

exportBed *Export BED format*

Description

Simple BED export for ranges exceeding threshold. If the transcript headers contain genomic coordinates (e.g. "chr1:1000-2000(+)"), it translates transcript-relative peak coordinates into genomic coordinates.

Usage

```
exportBed(results, output_file, rbp, threshold = 0)
```

Arguments

results	data.table.
output_file	File path.
rbp	RBP name to export.
threshold	Threshold for score/occupancy.

Value

Invisibly returns NULL. Called for side effect of writing BED file.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateBinding("ACGUACGU", rbp_models, c(HH = 100, HL = 100))
results$transcript <- "test_transcript"
tmp_bed <- tempfile(fileext = ".bed")
unlink(tmp_bed)
```

`exportResults` *Export Results*

Description

Exports simulation results to JSON or CSV.

Usage

```
exportResults(
  results,
  output_file,
  format = c("json", "csv"),
  include_sequence = TRUE
)
```

Arguments

`results`	data.table of results.
`output_file`	Output file path.
`format`	"json" or "csv".
`include_sequence`	Logical, include sequence in output (default TRUE).

Value

Invisibly returns NULL. Called for side effect of writing file.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateBinding("ACGUACGU", rbp_models, c(HH = 100, HL = 100))

# Export JSON
tmp_json <- tempfile(fileext = ".json")
exportResults(results, tmp_json, format = "json")
unlink(tmp_json)

# Export CSV
tmp_csv <- tempfile(fileext = ".csv")
exportResults(results, tmp_csv, format = "csv")
unlink(tmp_csv)
```

generateRNA *Generate Random RNA Sequence*

Description

Generates a random RNA sequence with specified nucleotide frequencies.

Usage

```
generateRNA(L, A = 0.25, G = 0.25, C = 0.25, U = 0.25)
```

Arguments

L	Length of the sequence.
A	Frequency of Adenine.
G	Frequency of Guanine.
C	Frequency of Cytosine.
U	Frequency of Uracil.

Value

A character string representing the RNA sequence.

Examples

```
set.seed(42)
generateRNA(100)
```

importResults *Import Results*

Description

Reads previously exported simulation results from a JSON or CSV file.

Usage

```
importResults(input_file)
```

Arguments

input_file	Path to JSON or CSV file (exported by exportResults).

Value

A data.table with the imported results.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateBinding("ACGUACGU", rbp_models, c(HH = 100, HL = 100))

# JSON Example
tmp_json <- tempfile(fileext = ".json")
exportResults(results, tmp_json, format = "json")
results_json <- importResults(tmp_json)
unlink(tmp_json)

# CSV Example
tmp_csv <- tempfile(fileext = ".csv")
exportResults(results, tmp_csv, format = "csv")
results_csv <- importResults(tmp_csv)
unlink(tmp_csv)
```

`loadModel` *Load RBP Model from CSV*

Description

Reads a CSV file containing RBP binding affinities/scores. The file should have a 'Motif' column and one or more RBP score columns.

Usage

```
loadModel(file, k = NULL, rbp = NULL)
```

Arguments

`file`	Path to the CSV file.
`k`	Optional integer. If provided, filters for k-mers of this length.
`rbp`	Optional character vector of RBP names to load. If NULL, loads all.

Value

A named list of data.tables, each with 'motif' and 'score' columns.

Examples

```
# Load sample model from package
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
raw_models <- loadModel(model_file)
names(raw_models)

# Load specific RBPs
raw_models <- loadModel(model_file, rbp = c("HH", "HL"))
```

makeSE *Create SummarizedExperiment from Results*

Description

Converts simulation results to a SummarizedExperiment object.

Usage

```
makeSE(results, rbp_models = NULL)
```

Arguments

results	data.table from simulateBinding or simulateBindingFasta.
rbp_models	Optional named list of RBP models (for colData metadata).

Value

A SummarizedExperiment with assays and row/col metadata.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateBinding("ACGUACGU", rbp_models, c(HH = 100, HL = 100))

# Create density columns for the mock results to enable makeSE
results[, HH_density := HH / sum(HH)]
results[, HL_density := HL / sum(HL)]
results[, HH_density_fc := HH_density * 8]
results[, HL_density_fc := HL_density * 8]
results[, HH_occupancy_fc := HH / mean(HH)]
results[, HL_occupancy_fc := HL / mean(HL)]

se <- makeSE(results, rbp_models)
se
```

parseGenomicHeader *Parse Genomic Coordinates from FASTA Header*

Description

Extracts genomic coordinates from a UCSC-style header string, e.g. "chr1:1000-2000" or "chr1:1000-2000(-)".

Usage

```
parseGenomicHeader(h)
```

Arguments

h	Character string.

Value

A list with chrom, start, end, and strand, or NULL.

`plotBinding` *Plot Binding Profile*

Description

Plots the binding profile for one or more RBPs.

Usage

```
plotBinding(
  results,
  rbp,
  transcript = NULL,
  metric = c("occupancy", "density", "density_fc", "occupancy_fc"),
  ylim = NULL,
  xlim = NULL,
  xaxis_type = c("nucleotide", "position", "both"),
  rna_conc = NULL,
  protein_conc = NULL,
  window = NULL
)
```

Arguments

`results`	data.table of simulation results.
`rbp`	Character vector of RBP names to plot.
`transcript`	Optional transcript name to filter (default NULL).
`metric`	Type of metric: "occupancy" (default), "density", "density_fc", or "occupancy_fc".
`ylim`	Optional y-axis limits (default NULL).
`xlim`	Optional x-axis limits (default NULL).
`xaxis_type`	Type of x-axis labels: "nucleotide" (default), "position", or "both".
`rna_conc`	Optional numeric. Filter by RNA concentration (default NULL).
`protein_conc`	Optional named numeric vector. Filter by protein concentration (default NULL).
`window`	Optional integer. Applies K-mer sliding window average (default NULL).

Value

A ggplot object.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateBinding("ACGUACGUACGU", rbp_models, c(HH = 100, HL = 100))
results$transcript <- "test_transcript"
plotBinding(results, rbp = c("HH", "HL"))
```

plotGrid *Plot Competition Grid*

Description

Visualizes binding across a concentration grid.

Usage

```
plotGrid(
  results,
  rbp1,
  rbp2,
  roi_range,
  metric = c("occupancy", "density", "density_fc", "occupancy_fc"),
  rbp1_concs = NULL
)
```

Arguments

results	data.table from simulateGrid.
rbp1	Name of first RBP (subdivides each cell).
rbp2	Name of second RBP (X-axis).
roi_range	Numeric vector (start, end) for region of interest.
metric	"occupancy" (default), "density", "density_fc", or "occupancy_fc".
rbp1_concs	Optional vector of RBP1 concentrations to include (default NULL).

Value

A ggplot object.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateGrid(
  sequence = "ACGUACGU",
  rbp_models = rbp_models,
  protein_conc_grid = list(HH = c(10, 100), HL = c(10, 100)),
  rna_conc_grid = 10
)
plotGrid(results, rbp1 = "HH", rbp2 = "HL", roi_range = c(1, 8))
```

plotHeatmap *Plot Binding Heatmap*

Description

Plots a heatmap of binding for multiple RBPs.

Usage

```
plotHeatmap(
  results,
  transcript,
  rbps = NULL,
  xlim = NULL,
  xaxis_type = c("nucleotide", "position", "both"),
  metric = c("occupancy", "density", "density_fc", "occupancy_fc"),
  window = NULL,
  zlim = NULL
)
```

Arguments

results	data.table of simulation results.
transcript	Transcript name to filter (required).
rbps	Optional vector of RBP names to include (default NULL, infers all).
xlim	Optional x-axis limits (default NULL).
xaxis_type	Type of x-axis labels: "nucleotide" (default), "position", or "both".
metric	Type of metric: "occupancy" (default), "density", "density_fc", or "occupancy_fc".
window	Optional integer for K-mer sliding window average (default NULL).
zlim	Optional numeric vector for color scale limits (default NULL).

Value

A ggplot object.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateBinding("ACGUACGUACGU", rbp_models, c(HH = 100, HL = 100))
results$transcript <- "test_transcript"
plotHeatmap(results, transcript = "test_transcript")
```

RBPEqBind *RBPEqBind: Simulate Competitive RBP Binding to RNA*

Description

RBPEqBind simulates competitive binding of multiple RNA-binding proteins (RBPs) to RNA sequences using an equilibrium binding model. The package provides tools for loading RBP affinity models, running binding simulations, and visualizing competition patterns.

Core Functions

Model Loading `loadModel`, `setModel`, `viewModel`

Simulation `simulateBinding`, `simulateGrid`, `simulateBindingFasta`, `simulateGridFasta`

Visualization `plotBinding`, `plotHeatmap`, `plotGrid`

Export `exportResults`, `makeSE`

Getting Started

See the package vignette for a comprehensive introduction: `vignette("RBPEqBind-intro", package = "RBPEqBind")`

Acknowledgments

This package was developed in part with Google Antigravity. For the original Python implementation of RNA-RBP interaction simulations, see https://github.com/S00NYI/Specificity_BITs.

Author(s)

Maintainer: Soon Yi <cu.soonyi@gmail.com> (ORCID)

Authors:

- Soon Yi <cu.soonyi@gmail.com> (ORCID)

Other contributors:

- National Institutes of Health (T32 GM152319) [funder]

References

Yi S, Singh SS, Ye X, Krishna R, Kothwela V, Jankowsky E, Luna JM. (2025). Inherent Specificity and Mutational Sensitivity as Quantitative Metrics for RBP Binding. bioRxiv. https://www.biorxiv.org/content/10.1101/2025.03.28.646018v4

See Also

Useful links:

- https://github.com/S00NYI/RBPEqBind
- Report bugs at https://github.com/S00NYI/RBPEqBind/issues

setModel *Set Model Affinity (Score to Kd Conversion)*

Description

Converts a model's normalized scores to Kd values using affinity scaling. Score is assumed to be proportional to affinity (Ka), so Kd = 1/Ka.

Usage

```
setModel(models, max_affinity = 100, min_affinity = 1e-05)
```

Arguments

models	Named list of models (from loadModel), each with 'motif' and 'score'.
max_affinity	Named numeric vector of max Ka per RBP, or single default (100).
min_affinity	Named numeric vector of min Ka per RBP, or single default (0.00001).

Value

A named list of data.tables, each with 'motif' and 'Kd' columns.

Examples

```
# Load and process model
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
raw_models <- loadModel(model_file)
rbp_models <- setModel(raw_models, max_affinity = 100)

# Check Kd values
head(rbp_models[[1]])
```

simulateBinding *Simulate Competitive Binding*

Description

Simulates competitive binding of multiple RBPs to a single RNA sequence.

Usage

```
simulateBinding(sequence, rbp_models, protein_concs, rna_conc = 10, k = 5)
```

Arguments

sequence	RNA sequence string.
rbp_models	Named list of data.frames/data.tables with 'motif' and 'Kd'.
protein_concs	Named numeric vector of total protein concentrations.
rna_conc	Total RNA concentration (default 10).
k	K-mer size (default 5).

Value

A data.table containing per-position binding probabilities/occupancies.

Examples

```
# Load and process model
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
raw_models <- loadModel(model_file, rbp = c("HH", "HL"))
rbp_models <- setModel(raw_models, max_affinity = 100)

# Simulate binding on a short sequence
seq <- "ACGUACGUACGUACGUACGU"
results <- simulateBinding(seq, rbp_models, c(HH = 100, HL = 100), rna_conc = 10)
head(results)
```

---

simulateBindingFasta *Simulate Binding from FASTA File(s)*

Description

Simulates competitive binding for all sequences in one or more FASTA files. Concurrency is controlled by the user's active `plan` (default is sequential).

Usage

```
simulateBindingFasta(
  fasta_file,
  rbp_models,
  protein_concs,
  rna_conc = 10,
  k = 5
)
```

Arguments

`fasta_file`	Character vector of path(s) to FASTA files.
`rbp_models`	Named list of RBP models.
`protein_concs`	Named numeric vector of protein concentrations.
`rna_conc`	RNA concentration (default 10).
`k`	K-mer size (default 5).

Value

A data.table with combined results, including a transcript column.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
fasta_file <- system.file("extdata", "test_transcripts.fa", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateBindingFasta(fasta_file, rbp_models, c(HH = 100, HL = 100))
head(results)
```

simulateGrid *Simulate Concentration Grid*

Description

Runs simulations across a grid of protein/RNA concentrations. Concurrency is controlled by the user's active `plan` (default is sequential).

Usage

```
simulateGrid(sequence, rbp_models, protein_conc_grid, rna_conc_grid, k = 5)
```

Arguments

`sequence`	RNA sequence string.
`rbp_models`	Named list of RBP models.
`protein_conc_grid`	List of numeric vectors for each RBP concentration.
`rna_conc_grid`	Numeric vector of RNA concentrations to sweep.
`k`	K-mer size (default 5).

Value

A data.table with results for all grid combinations.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
grid <- simulateGrid("ACGUACGU", rbp_models,
  protein_conc_grid = list(HH = c(10, 100), HL = c(10, 100)),
  rna_conc_grid = 10)
head(grid)
```

simulateGridFasta *Simulate Grid from FASTA File(s)*

Description

Simulates competitive binding for all sequences in one or more FASTA files across a concentration grid. Concurrency is controlled by the user's active `plan` (default is sequential).

Usage

```
simulateGridFasta(
  fasta_file,
  rbp_models,
  protein_conc_grid,
  rna_conc_grid,
  k = 5
)
```

Arguments

fasta_file	Character vector of path(s) to FASTA files.
rbp_models	Named list of RBP models.
protein_conc_grid	List of numeric vectors for each RBP concentration.
rna_conc_grid	Numeric vector of RNA concentrations to sweep.
k	K-mer size (default 5).

Value

A data.table with combined results, including a transcript column.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
fasta_file <- system.file("extdata", "test_transcripts.fa", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file, rbp = c("HH", "HL")))
results <- simulateGridFasta(
  fasta_file = fasta_file,
  rbp_models = rbp_models,
  protein_conc_grid = list(HH = c(10, 100), HL = c(10, 100)),
  rna_conc_grid = 10,
  k = 5
)
head(results)
```

---

viewModel *View Model Affinity Distributions*

Description

Creates a histogram of Kd or Ka distributions for RBP models.

Usage

```
viewModel(
  rbp_models,
  rbp = NULL,
  metric = c("Kd", "Ka"),
  bins = 50,
  colors = NULL,
  alpha = 0.5,
  log_scale = FALSE
)
```

Arguments

rbp_models	Named list of RBP model data.tables (from setModel).
rbp	Character vector of RBP names to plot (default NULL, plots all).
metric	Display metric: "Kd" (default) or "Ka" (affinity = 1/Kd).
bins	Number of histogram bins (default 50).
colors	Named vector of colors per RBP (default NULL, uses Okabe-Ito palette).
alpha	Transparency of histogram bars (default 0.5).
log_scale	Logical, use log10 scale for x-axis (default FALSE).

Value

A ggplot object.

Examples

```
model_file <- system.file("extdata", "model_RBP.csv", package = "RBPEqBind")
rbp_models <- setModel(loadModel(model_file))
viewModel(rbp_models, metric = "Kd")
```

Index