Package: HiSpaR (via r-universe)

October 20, 2026

Type Package

Title Hierarchical Inference of Spatial Positions from Hi-C Data

Version 1.1.0

Date 2026-01-15

Description Provides R bindings for HiSpa, a hierarchical Bayesian model for inferring three-dimensional chromatin structures from Hi-C contact matrices using Markov Chain Monte Carlo (MCMC) sampling. The package implements a cluster-based hierarchical approach that efficiently handles large-scale Hi-C datasets. It uses Rcpp and RcppArmadillo for efficient C++ integration with the original HiSpa C++ implementation, enabling fast computation of chromatin structure inference through parallel MCMC sampling.

License MIT + file LICENSE

Depends R (>= 4.5.0)

Imports Rcpp (>= 1.0.0), utils, stats, Matrix, HiCExperiment

biocViews Software, Epigenetics, HiC, StructuralPrediction, Bayesian, Spatial

LinkingTo Rcpp, RcppArmadillo

SystemRequirements C++17, GNU make, Armadillo (>= 9.0), OpenMP

URL https://github.com/masterStormtrooper/HiSpaR

BugReports https://github.com/masterStormtrooper/HiSpaR/issues

Encoding UTF-8

LazyData false

RoxygenNote 7.3.3

Suggests testthat (>= 3.0.0), knitr, rmarkdown, BiocStyle, rgl, HiContactsData, HiContacts, plotly, callr

VignetteBuilder knitr

Config/pak/sysreqs make libssl-dev zlib1g-dev

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

Date/Publication 2026-04-28 13:06:36 UTC

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

RemoteRef HEAD

RemoteSha 14a3af9fa4829689bde7578f18fe97212c71bce6

Contents

hispa_analyze 2
su1_contact_mat 5

Index **7**

`hispa_analyze` *Run HiSpa MCMC Analysis*

Description

Performs hierarchical Bayesian inference of 3D chromatin structure from Hi-C contact matrix using MCMC sampling. This function follows the exact workflow from the HiSpa C++ implementation.

Usage

```
hispa_analyze(
  hic_experiment,
  output_dir,
  mcmc_iterations = 6000L,
  num_clusters = 0L,
  mcmc_burn_in = 0L,
  mcmc_initial_sd = 0.1,
  mcmc_sd_floor = 1e-04,
  mcmc_sd_ceil = 0.3,
  use_cluster_init = FALSE,
  cluster_init_iterations = 1000L,
  cluster_initial_sd = 0.1,
  save_samples = FALSE,
  sample_interval = 50L,
  verbose = TRUE,
  filter_quantile = -1,
  safe_mode = TRUE
)
```

Arguments

`hic_experiment`	A Bioconductor 'HiCExperiment' object or a numeric matrix containing Hi-C contact data. If a 'HiCExperiment' object, the function extracts the contact matrix using 'gi2cm()' from the interactions and converts it to a numeric matrix for analysis. If a matrix, it is used directly.

`output_dir` Character string specifying the output directory path.

`mcmc_iterations`
Integer, number of MCMC iterations (default: 6000).

`num_clusters` Integer, number of clusters for hierarchical analysis. If 0 (default), automatically determined as sqrt(n).

`mcmc_burn_in` Integer, number of burn-in iterations to discard (default: 0).

`mcmc_initial_sd`
Numeric, initial standard deviation for MCMC proposals (default: 0.1).

`mcmc_sd_floor` Numeric, minimum allowed standard deviation (default: 0.0001).

`mcmc_sd_ceil` Numeric, maximum allowed standard deviation (default: 0.3).

`use_cluster_init`
Logical, use cluster-based initialization instead of random initialization (default: FALSE).

`cluster_init_iterations`
Integer, number of iterations for cluster initialization MCMC (default: 1000).

`cluster_initial_sd`
Numeric, initial standard deviation for cluster initialization MCMC (default: 0.1).

`save_samples` Logical, whether to save MCMC trace samples (default: FALSE).

`sample_interval`
Integer, save samples every n iterations (default: 50).

`verbose` Logical, enable verbose output (default: TRUE).

`filter_quantile`
Numeric, quantile threshold for filtering loci by contact counts (default: -1, no filtering). If >= 0, loci with row sums below this quantile are removed. For example, 0.1 removes loci in the bottom 10% of contact counts.

`safe_mode` Logical, whether to run analysis in a safe subprocess using callr (default: TRUE). If TRUE and callr is available, analysis runs in an isolated subprocess to prevent R crashes from affecting the parent session. Set to FALSE to run inline.

Details

This function implements the complete HiSpa workflow:

1. Filter loci by contact count (optional)
2. Load contact matrix from file
3. Assign loci to clusters (k-means)
4. Build cluster relationships
5. Initialize structure (random or cluster-based)
6. Assemble global structure
7. Run main MCMC algorithm
8. Save results to output directory

Locus Filtering: By default, no filtering is applied (filter_quantile = -1). If filter_quantile >= 0, loci with contact counts below the specified quantile are removed before analysis. For example, filter_quantile = 0.1 removes loci in the bottom 10% of contact counts. This improves computational efficiency and focuses analysis on loci with sufficient data.

All results are automatically saved to the output directory:

- **final_positions.txt** - Final inferred 3D coordinates (n x 3 matrix)
- **initial_positions.txt** - Initial positions before MCMC (n x 3 matrix)

Read results using standard R functions: `final_pos <- read.table("output_dir/final_positions.txt")`

Value

A list containing:

- **positions** - A numeric matrix of dimensions n x 3 containing the final inferred 3D coordinates of loci. Rows correspond to loci, columns are (x, y, z).
- **beta0** - The final intercept parameter of the log-distance relationship.
- **beta1** - The final slope parameter of the log-distance relationship.

If filtering was applied, the positions matrix has an attribute 'filtered_locus_indices' containing the original indices of the retained loci (before filtering). Additionally, all analysis results are saved as text files in the output directory.

Examples

```
# Load example contact matrix
data(su1_contact_mat)

# Check dimensions
dim(su1_contact_mat)


# Example 1: Run analysis with matrix input
su_res <- hispa_analyze(
  hic_experiment = su1_contact_mat,
  output_dir = tempdir(),
  mcmc_iterations = 100,
  mcmc_burn_in = 10,
  use_cluster_init = FALSE,
  verbose = TRUE
)

# check results
dim(su_res$positions)  # Should be n x 3
```

su1_contact_mat *Example Hi-C Contact Matrix*

Description

Hi-C contact matrix of Human Chromosome 21, collected from https://zenodo.org/records/3928890. This dataset is provided as an example for testing and demonstrating the HiSpaR package functionality.

Usage

```
su1_contact_mat
```

Format

A symmetric numeric matrix with 649 rows and 649 columns, where entry (i,j) represents the normalized contact frequency between genomic loci i and j. The matrix is:

Dimensions 649 x 649

Type Numeric matrix

Symmetry Symmetric (contact_matrix[i,j] = contact_matrix[j,i])

Diagonal Zero or near-zero (self-contacts)

Range Non-negative contact frequencies

Details

This contact matrix represents chromatin interaction frequencies derived from Hi-C experiments on Human Chromosome 21. Higher values indicate more frequent spatial proximity between genomic loci in the 3D nuclear space.

The data can be used directly with `hispa_analyze`

Source

Human Hi-C data, chromosome 21, collected from https://zenodo.org/records/3928890

See Also

`hispa_analyze` for running the analysis

Examples

```
# Load the example data
data(su1_contact_mat)

# Check dimensions
dim(su1_contact_mat)
```

```
# Check if matrix is symmetric
isSymmetric(su1_contact_mat)

# Summary statistics
summary(as.vector(su1_contact_mat))


# Visualize contact matrix
image(su1_contact_mat,
      main = "Hi-C Contact Matrix (SU1)",
      xlab = "Genomic Locus",
      ylab = "Genomic Locus")
```

Index

∗ **datasets**
su1_contact_mat, 5

hispa_analyze, 2, *5*

su1_contact_mat, 5