Package: CellProgramMapper (via r-universe)

May 26, 2026

Type Package

Title Map Single Cells to Reference Gene Expression Programs

Version 1.0.0

Date 2026-01-26

Description Maps single-cell RNA sequencing data to reference gene expression programs (GEPs) using non-negative matrix factorization. Enables cell type annotation and state characterization by projecting query cells onto pre-built or custom reference programs. Includes tools for building consensus references from multiple datasets. Features C++ accelerated NNLS solvers and built-in machine learning models for cell type prediction.

License MIT + file LICENSE

URL https://github.com/Zaoqu-Liu/CellProgramMapper, https://zaoqu-liu.github.io/CellProgramMapper/

BugReports https://github.com/Zaoqu-Liu/CellProgramMapper/issues

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.0.0)

Imports Rcpp (>= 1.0.0), Matrix, methods, stats, utils, curl, yaml, future, future.apply, data.table, rappdirs, tools, jsonlite

Suggests Seurat (>= 4.0.0), SeuratObject (>= 4.0.0), testthat (>= 3.0.0), knitr, rmarkdown, ggplot2, pheatmap, reshape2, anndata, hdf5r, reticulate

LinkingTo Rcpp, RcppArmadillo

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation yes

Config/pak/sysreqs libssl-dev

Repository https://zaoqu-liu.r-universe.dev

Date/Publication 2026-01-26 05:56:10 UTC

RemoteUrl https://github.com/Zaoqu-Liu/CellProgramMapper

RemoteRef main

RemoteSha aa7f0ba8d09dcdb7958f65f4fd9c35cbdd4fcd44

Contents

`add_results_to_seurat` *Add Results to Seurat Object*

Description

Add CellProgramMapper results to Seurat object metadata

Usage

```
add_results_to_seurat(object, result, prefix = "")
```

Arguments

object	A Seurat object
result	A CellProgramMapperResult object
prefix	Prefix for column names (default: "")

Value

Seurat object with added metadata

---

available_references *List Available Pre-built References*

Description

Returns a data frame of available pre-built references in the CellProgramMapper database

Usage

```
available_references()
```

Value

A data frame with reference information including Name, Version, Cell_Type, Tissue, Species, etc.

Examples

```
## Not run:
refs <- available_references()
print(refs)

## End(Not run)
```

BuildConsensusReference *Build Consensus Reference from Multiple cNMF Results*

Description

Build consensus gene expression programs (cGEPs) by clustering GEPs from multiple cNMF results based on their pairwise correlations.

Usage

```
BuildConsensusReference(
  cnmf_paths,
  ks = NULL,
  density_thresholds = NULL,
  tpm_fns = NULL,
  score_fns = NULL,
  output_dir = ".",
  prefix = "",
  order_thresh = NULL,
  corr_thresh = 0.5,
  pct_thresh = 0.666,
  verbose = TRUE
)
```

Arguments

cnmf_paths	Character vector of paths to cNMF project directories. Each path should include the cNMF project name at the end, e.g., "cnmf_output_dir/cnmf_name"
ks	Integer vector of K values used for each cNMF result
density_thresholds	Numeric vector of density thresholds used for each cNMF result
tpm_fns	Optional: direct paths to TPM spectra files (alternative to ks/density_thresholds)
score_fns	Optional: direct paths to score spectra files
output_dir	Output directory for results (default: ".")
prefix	Prefix for output filenames (default: "")
order_thresh	Maximum rank for programs to be considered for clustering (default: number of datasets)
corr_thresh	Minimum correlation for programs to cluster (default: 0.5)
pct_thresh	Minimum fraction of connected programs to merge clusters (default: 0.666)
verbose	Print progress messages (default: TRUE)

Value

A list containing:

- cluster_df: Data frame showing which GEPs clustered together
- spectra_tpm: Consensus spectra in TPM units
- spectra_score: Consensus spectra scores
- hvgs_union: Union of highly variable genes
- top_genes: Top genes for each cGEP

Examples

```
## Not run:
result <- BuildConsensusReference(
  cnmf_paths = c("path/to/cnmf1", "path/to/cnmf2"),
  ks = c(10, 15),
  density_thresholds = c(0.1, 0.1),
  output_dir = "./consensus_output"
)

## End(Not run)
```

`CellProgramMapper` *Map Single Cells to Reference Gene Expression Programs*

Description

Projects single-cell gene expression data onto a reference set of gene expression programs (GEPs) using non-negative matrix factorization. This enables cell type annotation and state characterization based on established program definitions.

Usage

```
CellProgramMapper(
  query,
  reference = "TCAT.V1",
  assay = NULL,
  layer = "counts",
  return_unnormalized = FALSE,
  method = c("cd", "active_set"),
  max_iter = 1000L,
  tol = 1e-08,
  n_workers = 1L,
  cache_dir = NULL,
  verbose = TRUE
)
```

Arguments

`query` Query data. Accepts multiple input formats:

- Seurat object (V4 or V5)
- Matrix or dgCMatrix (cells as rows, genes as columns)
- data.frame (cells as rows, genes as columns)
- File path (.h5ad, .mtx.gz, .tsv, .txt)

`reference` Reference spectra. Can be:

- Name of a pre-built reference (e.g., "TCAT.V1")

- Path to a reference file (.tsv, .txt)

`assay` For Seurat objects, which assay to use (default: active assay)

`layer` For Seurat objects, which layer/slot to extract (default: "counts")

`return_unnormalized`
Logical. If TRUE, also return raw usage values. Default is FALSE, returning only normalized usage.

`method` NNLS solver algorithm:

- "cd" - Coordinate descent (default, generally faster)
- "active_set" - Lawson-Hanson active set method

`max_iter` Maximum iterations for NNLS solver (default: 1000)

`tol` Convergence tolerance for NNLS solver (default: 1e-8)

`n_workers` Number of parallel workers for large datasets (default: 1)

`cache_dir` Directory for caching downloaded references

`verbose` Logical. Print progress messages (default: TRUE)

Details

The algorithm projects each cell's expression profile onto the reference gene expression programs by solving a non-negative least squares problem:

$$\min_{w_i \geq 0} ||x_i - H^T w_i||_2^2$$

where x_i is the scaled expression vector for cell i, H is the reference spectra matrix, and w_i is the usage vector to be estimated.

Input data is preprocessed by:

1. Subsetting to genes present in both query and reference
2. Scaling each gene by its standard deviation (without centering)

Value

A CellProgramMapperResult object containing:

usage Raw usage matrix (cells x programs)

usage_norm Normalized usage matrix (rows sum to 1)

scores Computed add-on scores (if defined in reference)

overlap_genes Genes used for mapping

ref_name Reference name

n_cells Number of cells processed

n_programs Number of programs in reference

See Also

`available_references` for listing pre-built references `add_results_to_seurat` for Seurat integration `BuildConsensusReference` for building custom references

Examples

```
## Not run:
# With Seurat object
result <- CellProgramMapper(seurat_obj, reference = "TCAT.V1")

# With matrix
result <- CellProgramMapper(counts_matrix, reference = "path/to/ref.tsv")

# Access results
head(result$usage_norm)
head(result$scores)

# Add to Seurat object
seurat_obj <- add_results_to_seurat(seurat_obj, result)

## End(Not run)
```

`compute_scores` *Compute Add-on Scores*

Description

Compute pre-defined scores from usage matrix based on score definitions

Usage

```
compute_scores(
  usage,
  usage_norm,
  score_data,
  score_path = NULL,
  ref_name = NULL,
  verbose = TRUE
)
```

Arguments

`usage`	Usage matrix (cells x programs)
`usage_norm`	Normalized usage matrix (rows sum to 1)
`score_data`	Score definitions loaded from YAML file
`score_path`	Path to score definitions directory
`ref_name`	Reference name (for built-in model lookup)
`verbose`	Print progress messages

Value

Data frame of computed scores

Examples

```
## Not run:
scores <- compute_scores(usage, usage_norm, score_data)

## End(Not run)
```

`consensus` *Build Consensus Reference from Multiple cNMF Results*

Description

Functions for building consensus gene expression programs from multiple datasets

---

`create_score_data` *Create Score Data Structure*

Description

Create a score_data structure from individual score definitions

Usage

```
create_score_data(...)
```

Arguments

`...`	Score definitions created by create_score_definition

Value

A score_data list compatible with compute_scores

Examples

```
## Not run:
score_data <- create_score_data(
  create_score_definition("Score1", c("GEP1", "GEP2")),
  create_score_definition("Score2", c("GEP3"), type = "discrete", threshold = 0.5)
)

## End(Not run)
```

create_score_definition *Create Custom Score Definition*

Description

Create a custom score definition for use with compute_scores

Usage

```
create_score_definition(
  name,
  columns,
  weights = NULL,
  type = c("continuous", "discrete"),
  threshold = NULL,
  normalization = c("normalized", "raw")
)
```

Arguments

name	Score name
columns	Vector of program names to include in score
weights	Optional vector of weights (default: all 1s)
type	Score type: "continuous" or "discrete"
threshold	Threshold for discrete scores
normalization	Use "normalized" or "raw" usage

Value

A score definition list

Examples

```
## Not run:
my_score <- create_score_definition(
  name = "MyScore",
  columns = c("GEP1", "GEP2", "GEP3"),
  weights = c(1, 2, 1),
  type = "continuous"
)

## End(Not run)
```

download_reference *Download Reference to Cache*

Description

Download a pre-built reference from the database to local cache

Usage

```
download_reference(reference, cache_dir = NULL, verbose = TRUE)
```

Arguments

reference	Name of the reference to download
cache_dir	Directory to cache the reference (default: package cache)
verbose	Print progress messages

Value

Path to the downloaded reference directory

Examples

```
## Not run:
ref_path <- download_reference("TCAT.V1")

## End(Not run)
```

get_lineage_probabilities *Get Lineage Probabilities*

Description

Get probability for each lineage class instead of just the predicted label.

Usage

```
get_lineage_probabilities(usage_norm, model_name = "TCAT.V1")
```

Arguments

usage_norm	Normalized usage matrix
model_name	Model name (default: "TCAT.V1")

Value

Matrix of probabilities (cells x classes)

Examples

```
## Not run:
probs <- get_lineage_probabilities(usage_norm, "TCAT.V1")

## End(Not run)
```

---

get_scores *Extract Scores from Result*

Description

Convenience function to extract computed scores from results.

Usage

```
get_scores(result)
```

Arguments

result	A CellProgramMapperResult object

Value

Scores data frame (or NULL if no scores computed)

Examples

```
## Not run:
result <- CellProgramMapper(seurat_obj, reference = "TCAT.V1")
scores <- get_scores(result)

## End(Not run)
```

get_usage *Extract Usage Matrix from Result*

Description

Convenience function to extract usage matrix from results.

Usage

```
get_usage(result, normalized = TRUE)
```

Arguments

result	A CellProgramMapperResult object
normalized	Logical. Return normalized usage (TRUE, default) or raw values (FALSE)

Value

Usage matrix as data frame

Examples

```
## Not run:
result <- CellProgramMapper(seurat_obj, reference = "TCAT.V1")
usage <- get_usage(result)

## End(Not run)
```

io *Data Input/Output Functions*

Description

Functions for reading and writing data

list_builtin_models *List Available Built-in Models*

Description

List Available Built-in Models

Usage

```
list_builtin_models()
```

Value

Character vector of available model names

load_counts *Load Counts Matrix from Various Formats*

Description

Load a counts matrix from various file formats including 10X outputs, tab-delimited text files, or h5ad files.

Usage

```
load_counts(counts_file)
```

Arguments

counts_file	Path to counts file. Supported formats: • .h5ad - AnnData format (requires anndata package) • .mtx.gz or .mtx - 10X sparse matrix format • .tsv, .txt, .csv - Tab or comma delimited text files

Value

A list containing:

- counts: Sparse matrix (cells x genes)
- cell_names: Character vector of cell barcodes
- gene_names: Character vector of gene names

Examples

```
## Not run:
data <- load_counts("path/to/matrix.mtx.gz")
dim(data$counts)

## End(Not run)
```

`load_reference` *Load Reference Spectra*

Description

Load reference spectra from file or cache

Usage

```
load_reference(reference, cache_dir = NULL, verbose = TRUE)
```

Arguments

`reference`	Either a reference name (e.g., "TCAT.V1") or path to a reference file (.tsv/.txt)
`cache_dir`	Directory to cache references
`verbose`	Print progress messages

Value

A list containing:

- spectra: Reference spectra matrix (programs x genes)
- score_data: Score definitions (if available)
- score_path: Path to score file (if available)
- ref_name: Reference name

Examples

```
## Not run:
ref <- load_reference("TCAT.V1")
dim(ref$spectra)

## End(Not run)
```

mapper *CellProgramMapper - Main Function*

Description

Map single cells to reference gene expression programs

print.CellProgramMapperResult *Print Method for CellProgramMapperResult*

Description

Print Method for CellProgramMapperResult

Usage

```
## S3 method for class 'CellProgramMapperResult'
print(x, ...)
```

Arguments

x	A CellProgramMapperResult object
...	Additional arguments (ignored)

Value

Invisible x

read_df_from_npz *Read DataFrame from NPZ File*

Description

Read a pandas DataFrame saved as NPZ file (format used by cNMF)

Usage

```
read_df_from_npz(file)
```

Arguments

file	Path to .npz file saved with np.savez() from pandas DataFrame

Value

data.frame with proper row and column names

Examples

```
## Not run:
df <- read_df_from_npz("path/to/data.df.npz")
head(df)

## End(Not run)
```

---

read_npz *Read NumPy .npz File*

Description

Read a NumPy .npz archive file (compressed collection of .npy files). For files containing object arrays (strings), uses reticulate/numpy if available.

Usage

```
read_npz(file)
```

Arguments

file	Path to .npz file

Value

Named list of arrays

Examples

```
## Not run:
data <- read_npz("path/to/file.npz")
names(data)

## End(Not run)
```

---

reference *Reference Management Functions*

Description

Functions for loading, downloading, and managing references

save_results *Save Results to Files*

Description

Save CellProgramMapper results to tab-delimited files

Usage

```
save_results(
  result,
  output_dir = ".",
  prefix = "CellProgramMapper",
  verbose = TRUE
)
```

Arguments

result	A CellProgramMapperResult object
output_dir	Output directory
prefix	Prefix for output files
verbose	Print progress messages

Value

Invisible NULL

Examples

```
## Not run:
save_results(result, output_dir = "./output", prefix = "my_analysis")

## End(Not run)
```

scoring *Scoring Functions*

Description

Functions for computing add-on scores from usage matrix

summary.CellProgramMapperResult *Summary Method for CellProgramMapperResult*

Description

Summary Method for CellProgramMapperResult

Usage

```
## S3 method for class 'CellProgramMapperResult'
summary(object, ...)
```

Arguments

object	A CellProgramMapperResult object
...	Additional arguments (ignored)

Value

Invisible summary data frame

Index