Package 'FIRM'

May 7, 2026

Type Package

Title Flexible Integration of Single-Cell RNA-Seq Data

Version 0.1.2

Maintainer Jingsi Ming <jsming@fem.ecnu.edu.cn>

Description Provides functions for the flexible integration of heterogeneous scRNA-seq datasets across multiple tissue types, platforms, and experimental batches. Implements the method described in Ming (2022) <doi:10.1093/bib/bbac167>. The package incorporates modified 'C++' source code from the 'flashpca' library (Abraham, 2014-2016 <https://github.com/gabraham/flashpca>) for efficient principal component analysis, and the 'Spectra' library (Qiu, 2016-2025) for large-scale eigenvalue and singular value decomposition; see 'inst/COPYRIGHTS' for details on third-party code.

URL https://github.com/mingjingsi/FIRM

BugReports https://github.com/mingjingsi/FIRM/issues

License GPL-3

Encoding UTF-8

Copyright See file 'inst/COPYRIGHTS' for details.

LazyData true

LazyDataCompression bzip2

Imports Seurat (>= 5.0.0), RANN

Suggests ggplot2

LinkingTo Rcpp, RcppArmadillo, RcppEigen

RoxygenNote 7.3.3

NeedsCompilation yes

Depends R (>= 3.5.0)

Author Jingsi Ming [aut, cre, cph] (R package development and method implementation, ORCID: 0000-0001-7059-4156),
Shuzhen Ding [ctb] (R package development and maintenance),
Gad Abraham [ctb, cph] (Original 'flashpca' 'C++' code for fast randomized PCA),
Yixuan Qiu [ctb, cph] ('Spectra' library for large-scale eigenvalue and

SVD computation),
Anna Araslanova [ctb, cph] (Author of 'LOBPCGSolver.h' in 'Spectra'),
Felipe Zapata [ctb, cph] (Author of 'DavidsonSymEigsSolver.h' and 'Orthogonalization.h' in 'Spectra' (Netherlands eScience Center)),
Nicolas Renaud [ctb, cph] (Author of 'RitzPairs.h' and 'SearchSpace.h' in 'Spectra' (Netherlands eScience Center)),
Jens Wehner [ctb, cph] (Author of 'JDSymEigsBase.h' in 'Spectra' (Netherlands eScience Center)),
Gael Guennebaud [ctb, cph] ('Eigen' library code for tridiagonal eigenvalue decomposition),
Jitse Niesen [ctb, cph] ('Eigen' library code for eigenvalue computation)

Repository CRAN

Date/Publication 2026-02-19 19:40:02 UTC

Contents

`ExampleData` *Example single-cell datasets*

Description

A list of Seurat objects and corresponding metadata from two 10x Genomics peripheral blood samples (SS2 and tenx).

Usage

```
ExampleData
```

Format

A list of 4 elements:

SS2 Seurat object from Smart-seq2 platform

tenx Seurat object from 10x Chromium platform

meta_SS2 data.frame, cell-level metadata for SS2

meta_tenx data.frame, cell-level metadata for tenx

Examples

```
data("ExampleData")
names(ExampleData)
head(ExampleData$meta_SS2)
```

`FIRM` *Flexible Integration of Single-Cell RNA-Seq Data*

Description

Performs unsupervised integration of two single-cell RNA-seq datasets by searching for the optimal clustering resolution pair that maximises mutual-nearest-neighbour (MNN) mixing in the combined PCA space. The final integrated expression matrix is returned after batch-effect correction.

Usage

```
FIRM(
  SS2,
  tenx,
  hvg1,
  hvg2,
  dims,
  all_genes = FALSE,
  res_seq_SS2 = seq(0.1, 2, 0.1),
  res_seq_tenx = seq(0.1, 2, 0.1),
  coreNum = 1,
  verbose = FALSE,
  seed = NULL
)
```

Arguments

`SS2`	`Seurat` object for reference dataset (e.g. Smart-seq2).
`tenx`	`Seurat` object for query dataset (e.g. 10x Genomics).
`hvg1, hvg2`	Character vectors giving the high-variable gene names selected in SS2 and tenx, respectively.
`dims`	Integer scalar, number of principal components to use during integration.

`all_genes` Logical scalar. If `TRUE` the integration is carried out on the union of all genes; otherwise only on the intersected HVGs (default).

`res_seq_SS2`, `res_seq_tenx` Numeric vectors of clustering resolutions to be screened for the reference and query dataset, respectively. Defaults to `seq(0.1, 2, 0.1)`.

`coreNum` Integer scalar, number of CPU cores used for parallel screening.

`verbose` Logical scalar. If `TRUE` a list containing the integrated matrix and quality metrics is returned; otherwise only the integrated expression matrix is returned.

`seed` Integer, random seed for reproducibility (optional). If NULL, uses current random state.

Details

The algorithm performs the following steps:

1. PCA on each dataset using the intersected HVGs.
2. SNN graph construction (via `Seurat::FindNeighbors`).
3. Screening of clustering resolution pairs (`res_seq_SS2` × `res_seq_tenx`) to maximise mutual-nearest-neighbour mixing in the joint PCA space.
4. Batch-effect correction with `FIRM_res*` functions.
5. Final integrated expression matrix is scaled and returned.

Quality control: the integrated embedding is compared with the naive PCA; if correction does not improve mixing the latter is returned.

Value

By default (`verbose = FALSE`) a single `matrix` of batch-corrected, scaled expression values with genes as rows and combined cells as columns.

If `verbose = TRUE` a named `list` is returned:

integrated As above, the corrected expression matrix.

Metric_PCA Mean MNN mixing score of the naive PCA (no correction).

Metric_FIRM Mean MNN mixing score of the FIRM-corrected embedding (matrix when multiple resolution pairs were screened).

References

Ming, J., Lin, Z., Zhao, J., Wan, X., Ezran, C., Liu, S., ... & TTM Consortium. (2022). FIRM: Flexible integration of single-cell RNA-sequencing data for large-scale multi-tissue cell atlas datasets. *Briefings in bioinformatics*, 23(5).

See Also

`prep_data` for Data preprocessing.

Examples

```
set.seed(42)

library(Seurat)
library(FIRM)

data("ExampleData")
prep_SS2  <- prep_data(ExampleData$SS2, hvg_genes = 1000)
Dataset1  <- prep_SS2$Dataset
hvg1      <- prep_SS2$hvg

prep_tenx <- prep_data(ExampleData$tenx, hvg_genes = 1000)
Dataset2  <- prep_tenx$Dataset
hvg2      <- prep_tenx$hvg

res <- FIRM(Dataset1, Dataset2, hvg1, hvg2,
           dims = 15, all_genes = FALSE, seed = 42)

dim(res)
```

---

label_trans *Label transfer via k-NN (hard assignment)*

Description

For every query cell, find its k nearest reference cells in the integrated PCA space and assign the most frequent label among them.

Usage

```
label_trans(
  integrated,
  ref = "10X",
  query = "SS2",
  label_key = "annotation",
  emb_key = "pca",
  batch_key = "dataset",
  k = 10,
  seed = NULL
)
```

Arguments

integrated	Seurat object with integrated PCA and meta-data columns dataset and annotation.
ref	Character string identifying the reference dataset (default "10X").
query	Character string identifying the query dataset (default "SS2").

label_key	Name of the meta-data column that indicates celltype
emb_key	Name of the reduction to use (default "pca").
batch_key	Name of the meta-data column that indicates dataset/batch
k	Number of nearest neighbours to vote (default 10).
seed	Integer, random seed for reproducibility (optional). If NULL, uses current random state.

Value

Named character vector: predicted label for each query cell (names are cell barcodes).

---

label_trans_prob *Label transfer via k-NN (soft assignment / probabilities)*

Description

Identical to label_trans but returns a probability matrix instead of a hard label.

Usage

```
label_trans_prob(
  integrated,
  ref = "10X",
  query = "SS2",
  label_key = "annotation",
  emb_key = "pca",
  batch_key = "dataset",
  k = 10,
  seed = NULL
)
```

Arguments

integrated	Seurat object containing the joint embedding and meta-data.
ref	Name of the reference dataset/batch (default "10X").
query	Name of the query dataset/batch (default "SS2").
label_key	Name of the meta-data column that stores cell-type / label information (default "annotation").
emb_key	Name of the reduction to use (default "pca").
batch_key	Name of the meta-data column that indicates dataset/batch origin (default "dataset").
k	Number of nearest neighbours to consider (default 10).
seed	Integer, random seed for reproducibility (optional). If NULL, uses current random state.

Value

Numeric matrix (query-cells × unique-labels). Row sums equal 1. Row names are query cell barcodes, column names are reference labels.

`Local_Struct` *Local structure preservation metric*

Description

Compute the neighbourhood overlap between the original PCA space of each dataset and the integrated PCA space, averaged over the top `neighbors` nearest neighbours. High overlap indicates better local structure preservation after integration.

Usage

```
Local_Struct(
  SS2,
  tenx,
  integrated,
  dims = 30,
  emb_key = "pca",
  batch_key = "dataset",
  neighbors = 20
)
```

Arguments

`SS2`, `tenx`	Seurat objects containing the **unintegrated** PCA reduction (slot `reductions$pca`).
`integrated`	Seurat object that contains the **joint** PCA reduction and a meta-data column `dataset`.
`dims`	Number of principal components to use (default 30).
`emb_key`	Name of the reduction to use (default `"pca"`).
`batch_key`	Name of the meta-data column that indicates dataset/batch origin (default `"dataset"`).
`neighbors`	Size of the neighbourhood to evaluate (default 20).

Value

A named list with two numeric vectors:

`SS2`	Per-cell overlap scores for the SS2 dataset.
`tenx`	Per-cell overlap scores for the 10X dataset.

`match_score` *Match score between query and reference neighbourhoods*

Description

Compute, for each query cell, the ratio of the average within-query distance to the average query-to-reference distance in the integrated PCA space. Smaller values imply better alignment.

Usage

```
match_score(
  integrated,
  ref = "10X",
  query = "SS2",
  label_key = "annotation",
  emb_key = "pca",
  batch_key = "dataset",
  k = 10,
  seed = NULL
)
```

Arguments

`integrated`	Seurat object containing the joint embedding and meta-data.
`ref`	Name of the reference dataset/batch (default `"10X"`).
`query`	Name of the query dataset/batch (default `"SS2"`).
`label_key`	Name of the meta-data column that stores cell-type / label information (default `"annotation"`).
`emb_key`	Name of the reduction to use (default `"pca"`).
`batch_key`	Name of the meta-data column that indicates dataset/batch origin (default `"dataset"`).
`k`	Number of nearest neighbours to consider (default `10`).
`seed`	Integer, random seed for reproducibility (optional). If NULL, uses current random state.

Value

Named numeric vector of length equal to the number of query cells.

Mixing_Metric *k-NN Mixing Metric*

Description

Evaluate the degree of mixing between two or more datasets in a low-dimensional embedding.

Usage

```
Mixing_Metric(embedding, dataset_list, k = 5, max.k = 300, seed = NULL)
```

Arguments

embedding	Numeric matrix (n × d) where each row is the coordinates of one sample in the low-dimensional space (PCA, UMAP, t-SNE, ...).
dataset_list	Factor or character vector of length n indicating which dataset each sample comes from.
k	Positive integer. The rank of the within-dataset neighbour to look for (default 5).
max.k	Positive integer. Total number of nearest neighbours to compute (default 300).
seed	Integer, random seed for reproducibility (optional). If NULL, uses current random state.

Value

Named numeric vector with one entry per unique level of 'dataset_list'. The entry is the median position (among 1 ... max.k) of the k-th within-dataset neighbour across all samples that belong to that dataset. Values are clamped to max.k when fewer than k within-dataset neighbours exist.

prep_data *Data preprocessing for FIRM integration*

Description

Performs a standard Seurat workflow: normalization, scaling and selection of the top 4 000 highly-variable genes (HVGs).

Usage

```
prep_data(counts, hvg_genes = 4000)
```

Arguments

counts	Raw count matrix (genes × cells) or dgCMatrix.
hvg_genes	Target number of genes to return (default 4 000).

Value

A named list with elements

Dataset Seurat object after normalization, scaling and feature selection.

hvg Character vector of the 4 000 most variable genes.

Author(s)

Jingsi Ming

See Also

FIRM for the integration step.

Examples

```
set.seed(42)
library(Seurat)
library(FIRM)
data("ExampleData")
prep_SS2  <- prep_data(ExampleData$SS2, hvg_genes = 1000)
Dataset1  <- prep_SS2$Dataset
hvg1      <- prep_SS2$hvg

prep_tenx <- prep_data(ExampleData$tenx, hvg_genes = 1000)
Dataset2  <- prep_tenx$Dataset
hvg2      <- prep_tenx$hvg
```

SelectGene *Select consensus genes across multiple HVG lists*

Description

Rank-based selection of genes that appear in multiple HVG lists, prioritising those present in all datasets.

Usage

```
SelectGene(hvg_list, gene_all = NULL, num = 4000)
```

Arguments

hvg_list	List of character vectors, each containing HVGs from one dataset.
gene_all	Optional character vector of all candidate genes (intersection filter).
num	Target number of genes to return (default 4 000).

Value

Character vector of selected genes.

`Select_hvg` *Combine or select top HVGs from saved prep_data outputs*

Description

Load HVG lists saved by `prep_data` and either take their union or rank-based top consensus.

Usage

```
Select_hvg(file_names, file_path, method, hvg_genes = 4000)
```

Arguments

`file_names`	Character vector of basenames (without "_hvg.RData").
`file_path`	Directory containing the `*_hvg.RData` files.
`method`	"all" for union, "top" for ranked consensus (see `SelectGene`).
`hvg_genes`	Target number of genes to return (default 4 000).

Value

Character vector of selected genes.

See Also

`prep_data`

Index