

# Package

September 1, 2025

**Version** 1.1.0

**Date** 2025-09-01

**Title** A Fast Tool for Single-Cell Spatially Variable Genes Identifications on Large-Scale Data

**Description** Identifying spatially variable genes is critical in linking molecular cell functions with tissue phenotypes. This package utilizes a granularity-based dimension-agnostic tool, single-cell big-small patch (scBSP), implementing sparse matrix operation and KD tree methods for distance calculation, for the identification of spatially variable genes on large-scale data. The detailed description of this method is available at Wang, J. and Li, J. et al. 2023 (Wang, J. and Li, J. (2023), <[doi:10.1038/s41467-023-43256-5](https://doi.org/10.1038/s41467-023-43256-5)>).

**License** GPL (>= 2)

**Encoding** UTF-8

**Imports** Matrix,  
sparseMatrixStats,  
fitdistrplus,  
RANN,  
spam

**Suggests** knitr,  
rmarkdown

**RoxygenNote** 7.3.2

## Contents

CombinePvalues	1
LoadSpatial	2
scBSP	3
SpFilter	4
<b>Index</b>	<b>5</b>

---

CombinePvalues	<i>Combine p-values across multiple samples from scBSP</i>
----------------	--

---

## Description

Given the results from multiple samples with gene names and p-values, this function merges them by gene and computes a combined p-value for each gene. Fisher's method or Stouffer's method can be used.

**Usage**

```
CombinePvalues(list_of_pvalues, method = c("fisher", "stouffer"))
```

**Arguments**

`list_of_pvalues` A list of data.frames, each with columns: GeneNames, P\_values.

`method` Combination method. One of "fisher" (default) or "stouffer".

**Value**

A data.frame with columns:

- GeneNames
- Number\_Samples: number of datasets contributing to this gene
- Calibrated\_P\_values: the combined p-value

**Examples**

```
df1 <- data.frame(GeneNames = c("A", "B", "C"),
                  P_values = c(0.01, 0.20, 0.03))
df2 <- data.frame(GeneNames = c("A", "C", "D"),
                  P_values = c(0.04, 0.10, 0.50))
df3 <- data.frame(GeneNames = c("B", "C", "E"),
                  P_values = c(0.05, 0.02, 0.80))

CombinePvalues(list(df1, df2, df3), method = "fisher")
```

---

LoadSpatial

*Loading data from a Seurat object or a data frame.*


---

**Description**

A function to load and filter data from a Seurat object or a data frame.

**Usage**

```
LoadSpatial(InputData, Dimension = 2)
```

**Arguments**

`InputData` A Seurat spatial object or a M x (D + N) data matrix representing the D-dimensional coordinates and expressions of N genes on M spots. The coordinates should be placed at the first D columns

`Dimension` The dimension of coordinates

**Value**

A list of two data frame:

`Coords` A M x D matrix representing D-dimensional coordinates for M spots

`ExpMatrix` A sparse, N x M expression matrix in dgCMatrx class with N genes and M spots

**Description**

This function is designed to identify spatially variable genes through a granularity-based approach.

**Usage**

```
scBSP(Coords, ExpMat_Sp, D_1 = 1.0, D_2 = 3.0,
      Exp_Norm = TRUE, Coords_Norm_Method = c("Sliced", "Overall", "None"),
      K_NN = 100, treetype = "kd")
```

**Arguments**

Coords	A M x D matrix representing D-dimensional coordinates for M spots
ExpMat_Sp	A sparse, N x M expression matrix in dgCMatrx class with N genes and M spots
D_1	Size of the small patch
D_2	Size of the big patch
Exp_Norm	A Boolean value indicating whether the expression matrix should be normalized
Coords_Norm_Method	Normalization method for the coordinates matrix, which can be "None", "Sliced", or "Overall".
K_NN	The maximum number of nearest neighbours to compute.
treetype	Character vector specifying the standard 'kd' tree or a 'bd' (box-decomposition, AMNSW98) tree which may perform better for larger point sets.

**Details**

This function utilizes a MxD matrix (Coords) representing D-dimensional coordinates with M spots and a sparse, NxM expression matrix (ExpMat\_Sp) with N genes and M spots.

**Value**

A data frame with the name of genes and corresponding p-values.

**Examples**

```
Coords <- expand.grid(1:100, 1:100, 1:3)
RandFunc <- function(n) floor(10 * stats::rbeta(n, 1, 5))
Raw_Exp <- Matrix::rsparsematrix(nrow = 10^4, ncol = 3*10^4, density = 0.0001, rand.x = RandFunc)
Filtered_ExpMat <- SpFilter(Raw_Exp)
rownames(Filtered_ExpMat) <- paste0("Gene_", 1:nrow(Filtered_ExpMat))
P_values <- scBSP(Coords, Filtered_ExpMat)
```

---

**SpFilter***A function for filtering low expressed genes*

---

**Description**

A function for filtering low expressed genes

**Usage**

```
SpFilter(ExpMat_Sp, Threshold = 5)
```

**Arguments**

ExpMat_Sp	A sparse, N x M expression matrix in dgCMatrix class with N genes and M spots
Threshold	A threshold set to filter out genes with a total read count below this specified value

**Value**

A sparse expression matrix in dgCMatrix class

**Examples**

```
# create a sparse expression matrix
Raw_ExpMat <- Matrix::rsparsematrix(nrow = 10000, ncol = 2000,
density = 0.01, rand.x = function(n) rpois(n, 15))
Filtered_ExpMat <- SpFilter(Raw_ExpMat)
```

# Index

CombinePvalues, [1](#)

LoadSpatial, [2](#)

scBSP, [3](#)

SpFilter, [4](#)