

# Package: SC.MEB (via r-universe)

September 19, 2024

**Type** Package

**Title** Spatial Clustering with Hidden Markov Random Field using Empirical Bayes

**Version** 1.1

**Date** 2021-09-30

**Description** Spatial clustering with hidden markov random field fitted via EM algorithm, details of which can be found in Yi Yang (2021) <[doi:10.1101/2021.06.05.447181](https://doi.org/10.1101/2021.06.05.447181)>. It is not only computationally efficient and scalable to the sample size increment, but also is capable of choosing the smoothness parameter and the number of clusters as well.

**License** GPL-3

**Depends** mclust,parallel,ggplot2, Matrix, R (>= 3.5)

**Imports** Rcpp (>= 1.0.6), SingleCellExperiment, purrr,BiocSingular, SummarizedExperiment, scater, scran, S4Vectors

**LinkingTo** Rcpp, RcppArmadillo

**RoxygenNote** 7.1.2

**VignetteBuilder** knitr

**Suggests** knitr, rmarkdown

**Encoding** UTF-8

**NeedsCompilation** yes

**Author** Yi Yang [aut, cre], Xingjie Shi [aut], Jin Liu [aut]

**Maintainer** Yi Yang <[yygaosansiban@sina.com](mailto:yygaosansiban@sina.com)>

**Repository** CRAN

**Date/Publication** 2021-10-08 08:40:21 UTC

## Contents

ClusterPlot . . . . .	2
find_neighbors2 . . . . .	3

getneighborhood_fast . . . . .	3
ICMEM . . . . .	4
parafun . . . . .	5
PC . . . . .	7
SC.MEB . . . . .	7
sce . . . . .	9
selectK . . . . .	9
selectKPlot . . . . .	10
spatialPreprocess . . . . .	11

<b>Index</b>	<b>13</b>
--------------	-----------

---

ClusterPlot	<i>ClusterPlot.</i>
-------------	---------------------

---

## Description

The function ClusterPlot is used to Visualize spatial clusters.

## Usage

```
ClusterPlot(out, pos, size = 5, shape = 15)
```

## Arguments

out	is the output of function selectK.
pos	is a n-by-2 matrix of position.
size	is a positive value for characterizing the size of point in the plot, which is the same as size in ggplot2.
shape	is a positive value for characterizing the shape of point in the plot, which is the same as shape in ggplot2.

## Details

The function ClusterPlot is used to Visualize spatial clusters.

## Value

a ggplot2 object.

## Examples

```
pos = cbind(rep(1:5, each=5), rep(1:5, 5))
out = list()
out[[1]] = ""
out[[2]] = rep(1:5, each = 5)
ClusterPlot(out, pos)
```

---

find\_neighbors2      *find\_neighbors2.*

---

**Description**

find\_neighbors2 was used to find the neighborhood of spot.

**Usage**

```
find_neighbors2(sce, platform)
```

**Arguments**

sce                    is a SingleCellExperiment object containing PCA and position informatin.

platform              is the name of spatial transcriptomic platform. Specify 'Visium' for hex lattice geometry or 'ST' for square lattice geometry. Specifying this parameter is optional as this information is included in their metadata.

**Details**

find\_neighbors2 was used to find the neighborhood of spot.

**Value**

a sparse matrix recording the information of neighborhood.

**Examples**

```
data(sce)
platform = "ST"
Adj <- find_neighbors2(sce, platform)
```

---

getneighborhood\_fast      *getneighborhood\_fast*

---

**Description**

an efficient function to find the neighborhood based on the matrix of position and a pre-defined cutoff

**Usage**

```
getneighborhood_fast(x, cutoff)
```

**Arguments**

`x` is a n-by-2 matrix of position.

`cutoff` is a threshold of Euclidean distance to decide whether a spot is an neighborhood of another spot. For example, if the Euclidean distance between spot A and B is less than cutoff, then A is taken as the neighbourhood of B.

**Value**

A sparse matrix containing the neighbourhood

**Examples**

```
pos = cbind(rep(1:5, each=5), rep(1:5, 5))
Adj = getneighborhood_fast(pos, 2)
```

---

 ICMEM

---

*ICMEM.*


---

**Description**

The function ICMEM was used to conduct spatial clustering with hidden Markov random field for a sequence of beta and fixed number of clusters

**Usage**

```
ICMEM(
  y,
  x_int,
  Adj,
  mu_int,
  sigma_int,
  alpha,
  beta_grid,
  PX,
  maxIter_ICM,
  maxIter
)
```

**Arguments**

`y` is a matrix of PCs containing gene expression.

`x_int` is a vector of initial cluster label.

`Adj` is a matrix containing neighborhood information generated by `find_neighbors2`.

`mu_int` is a initial mean vector. we often generated it by Gaussian mixture model.

`sigma_int` is a initial co-variance matrix. we often generated it by Gaussian mixture model.

alpha	is a intercept.
beta_grid	is a sequence of smoothing parameter that can be specified by user.
PX	is a logical value specifying the parameter expansion in EM algorithm.
maxIter_ICM	is the maximum iteration of ICM algorithm.
maxIter	is the maximum iteration of EM algorithm.

### Details

The function ICMEM was used to conduct spatial clustering with hidden Markov random field for fixed beta and fixed number of clusters

### Value

a list.

The item 'x' is the clustering result.

The item 'gam' is the posterior probability matrix.

The item 'ell' is the opposite log-likelihood.

The item 'mu' is the mean of each component.

The item 'sigma' is the variance of each component.

### Examples

```

y = matrix(rnorm(50, 0, 1), 25,2)
pos = cbind(rep(1:5, each=5), rep(1:5, 5))
Adj = getneighborhood_fast(pos, 1.2)
beta_grid = c(0.5,1)
G = 2
fit_int = Mclust(y, G = G)
x_gmm <- fit_int$classification
mu_int <- unname(fit_int$parameter$mean)
sigma_int <- unname(fit_int$parameter$variance$sigma)
alpha <- -log(fit_int$parameter$pro)*0
reslist <- ICMEM(y = y, x_int = x_gmm, Adj = Adj, mu_int = mu_int, sigma_int = sigma_int,
alpha = alpha, beta_grid = beta_grid,
PX = TRUE, maxIter_ICM = 10, maxIter = 50)

```

---

parafun

*parafun.*

---

### Description

The function parafun implements the model SC-MEB for fixed number of clusters and a sequence of beta with initial value from Gaussian mixture model

**Usage**

```
parafun(
  y,
  Adj,
  G,
  beta_grid = seq(0, 4, 0.2),
  PX = TRUE,
  maxIter_ICM = 10,
  maxIter = 50
)
```

**Arguments**

<code>y</code>	is n-by-d PCs.
<code>Adj</code>	is a sparse matrix of neighborhood.
<code>G</code>	is an integer specifying the numbers of clusters.
<code>beta_grid</code>	is a numeric vector specifying the smoothness parameter of Random Markov Field. The default is seq(0,4,0.2).
<code>PX</code>	is a logical value specifying the parameter expansion in EM algorithm.
<code>maxIter_ICM</code>	is the maximum iteration of ICM algorithm. The default is 10.
<code>maxIter</code>	is the maximum iteration of EM algorithm. The default is 50.

**Details**

The function `parafun` implements the model SC-MEB for fixed number of clusters and a sequence of beta with initial value from Gaussian mixture model

**Value**

a list, We briefly explain the output of the SC.MEB.

The item `'x'` storing clustering results.

The item `'gam'` is the posterior probability matrix.

The item `'ell'` is the opposite log-likelihood.

The item `'mu'` is the mean of each component.

The item `'sigma'` is the variance of each component.

**Examples**

```
y = matrix(rnorm(50, 0, 1), 25,2)
pos = cbind(rep(1:5, each=5), rep(1:5, 5))
Adj_sp = getneighborhood_fast(pos, 1.2)
beta_grid = c(0.5,1)
G = 2
out = parafun(y, Adj_sp, G, beta_grid)
```

---

PC	<i>simulated PCs</i>
----	----------------------

---

**Description**

A dataset containing PCs

**Usage**

```
data(PC)
```

**Format**

It is a matrix containing 5 PCs  
the variables are listed as following

**PC1** The 1th PC

**PC2** The 2th PC ...

**PC5** The 5th PC

**Examples**

```
## run the PC with the Gaussian mixture model
data(PC)
out1 = mclust::Mclust(PC,G = 2)
```

---

SC.MEB	<i>SC.MEB.</i>
--------	----------------

---

**Description**

SC.MEB implements the model SC-MEB, spatial clustering with hidden Markov random field using empirical Bayes.

**Usage**

```
SC.MEB(  
  y,  
  Adj_sp,  
  beta_grid = seq(0, 4, 0.2),  
  K_set = 2:10,  
  parallel = TRUE,  
  num_core = 5,  
  PX = TRUE,  
  maxIter_ICM = 10,  
  maxIter = 50  
)
```

**Arguments**

<code>y</code>	is n-by-d PCs.
<code>Adj_sp</code>	is a sparse matrix of neighborhood. It is often generated from function <code>find_neighbors2</code> or <code>getneighborhood_fast</code> .
<code>beta_grid</code>	is a numeric vector specifying the smoothness parameter of Random Markov Field. The default is <code>seq(0,4,0.2)</code> .
<code>K_set</code>	is an integer vector specifying the numbers of mixture components (clusters) for which the BIC is to be calculated. The default is <code>K = 2:10</code> .
<code>parallel</code>	is a logical value to decide whether the function SC.MEB run in parallel. The default is <code>TRUE</code> .
<code>num_core</code>	is an integer value to decide how many cores are used to run SC.MEB in parallel.
<code>PX</code>	is a logical value to decide whether to use parameter expansion in EM algorithm
<code>maxIter_ICM</code>	is the maximum iteration of ICM algorithm. The default is 10.
<code>maxIter</code>	is the maximum iteration of EM algorithm. The default is 50.

**Details**

SC.MEB can implements the model SC-MEB in parallel which can improve the speed of the computation.

**Value**

a list, We briefly explain the output of the SC.MEB.

The item `'x'` contains clustering results.

The item `'gam'` is the posterior probability matrix.

The item `'ell'` is the opposite log-likelihood.

The item `'mu'` is the mean of each component.

The item `'sigma'` is the variance of each component.

**References**

Yang Y, Shi X, Zhou Q, et al. SC-MEB: spatial clustering with hidden Markov random field using empirical Bayes[J]. bioRxiv, 2021.

**Examples**

```
y = matrix(rnorm(50, 0, 1), 25,2)
pos = cbind(rep(1:5, each=5), rep(1:5, 5))
Adj_sp = getneighborhood_fast(pos, 1.2)
beta_grid = c(0.5,1)
K_set = 2:3
out = SC.MEB(y, Adj_sp, beta_grid, K_set, TRUE, 2)
```



---

sce	<i>A simulated SingleCellExperiment</i>
-----	---

---

**Description**

A dataset of SingleCellExperiment

**Usage**

```
data(sce)
```

**Format**

It is a SingleCellExperiment object with gene expression and meta information

**References**

Amezquita R A, Lun A T L, Becht E, et al. Orchestrating single-cell analysis with Bioconductor[J]. Nature methods, 2020, 17(2): 137-145.

**Examples**

```
## find the neighborhood of spots in SingleCellExperiment
data(sce)
out = find_neighbors2(sce, "ST")
```

---

selectK	<i>selectK.</i>
---------	-----------------

---

**Description**

The function selectK is used to select the best K according to BIC or Modified BIC criterion.

**Usage**

```
selectK(SCobject, K_set = 2:10, criterion = "BIC", c = 1)
```

**Arguments**

SCobject	is an object generated from SC.MEB function.
K_set	is a integer vector used in SC.MEB. The default is 2:10
criterion	is a character specifying the criterion for selecting K. The default value is BIC. The alternative value MBIC can also be used.

- `c` is a positive value in the modified BIC. The default is 1. Here we briefly explain how to choose the parameter `c` in the modified BIC. In general, For the ST or Visium dataset, it often ranges from 0.4 to 1 while for the MERFISH dataset with large number of cells, it often becomes larger, for example 10,20. Most importantly, SC-MEB is fast, scaling well in terms of sample size, which allow the user to tune the `c` based on their prior knowledge about the tissues or cells.

### Details

The function `selectK` is used to select the best `K` according to BIC or Modified BIC criterion.

### Value

a list contains two items. one is for the best `K` and the other is the clustering labels of `n` spots.

### Examples

```
y = matrix(rnorm(50, 0, 1), 25,2)
pos = cbind(rep(1:5, each=5), rep(1:5, 5))
Adj_sp = getneighborhood_fast(pos, 1.2)
beta_grid = c(0.5,1)
K_set = 2:3
out = SC.MEB(y, Adj_sp, beta_grid, K_set, TRUE, 2)
selectK(out, K_set)
```

---

`selectKPlot`

*selectKPlot.*

---

### Description

The function `selectKPlot` is used to demonstrate the scatter plot of BIC or Modified BIC vs `K` for selecting the best `K`.

### Usage

```
selectKPlot(SCobject, K_set = 2:10, criterion = "BIC", c = 1)
```

### Arguments

- `SCobject` is a object generated from `SC.MEB` function.
- `K_set` is the corresponding `K_set` used in your previous function `SC.MEB`.
- `criterion` is a character specifying the criterion for selecting `K`. The default is `BIC`, the alternative criterion `MBIC` can also be used.
- `c` is a positive value in modified BIC. The default is 1. Here we briefly explain how to choose the parameter `c` in the modified BIC. In general, For the ST or Visium dataset, it often ranges from 0.4 to 1 while for the MERFISH dataset with large number of cells, it often becomes larger, for example 10,20. Most importantly, SC-MEB is fast, scaling well in terms of sample size, which allow the user to tune the `c` based on their prior knowledge about the tissues or cells.

**Details**

The function `selectKPlot` is used to demonstrate the scatter plot of BIC or Modified BIC vs K for selecting the best K.

**Value**

a `ggplot2` object.

**Examples**

```
y = matrix(rnorm(50, 0, 1), 25,2)
pos = cbind(rep(1:5, each=5), rep(1:5, 5))
Adj_sp = getneighborhood_fast(pos, 1.2)
beta_grid = c(0.5,1)
K_set = 2:3
out = SC.MEB(y, Adj_sp, beta_grid, K_set, TRUE, 2)
selectKPlot(out, K_set)
```

---

spatialPreprocess      *Preprocess a spatial dataset for SC-MEB*

---

**Description**

Adds metadata required for downstream analyses, and (optionally) performs PCA on log-normalized expression of top HVGs.

**Usage**

```
spatialPreprocess(
  sce,
  platform = c("Visium", "ST"),
  n.PCs = 15,
  n.HVGs = 2000,
  skip.PCA = FALSE,
  log.normalize = TRUE,
  assay.type = "logcounts",
  BSPARAM = BiocSingular::ExactParam()
)
```

**Arguments**

<code>sce</code>	SingleCellExperiment to preprocess
<code>platform</code>	Spatial sequencing platform. Used to determine spot layout and neighborhood structure (Visium = hex, ST = square).
<code>n.PCs</code>	Number of principal components to compute. We suggest using the top 15 PCs in most cases.
<code>n.HVGs</code>	Number of highly variable genes to run PCA upon.

skip.PCA	Skip PCA (if dimensionality reduction was previously computed.)
log.normalize	Whether to log-normalize the input data with scater. May be omitted if log-normalization previously computed.
assay.type	Name of assay in sce containing normalized counts. Leave as "logcounts" unless you explicitly pre-computed a different normalization and added it to sce under another assay. Note that we do not recommend running BayesSpace on PCs computed from raw counts.
BSPARAM	A <a href="#">BiocSingularParam</a> object specifying which algorithm should be used to perform the PCA. By default, an exact PCA is performed, as current spatial datasets are generally small (<10,000 spots). To perform a faster approximate PCA, please specify <code>FastAutoParam()</code> and set a random seed to ensure reproducibility.

**Value**

SingleCellExperiment with PCA and SC.MEB metadata

**Examples**

```
## read the simulated data
data(sce)
platform = "ST"
out = find_neighbors2(sce, platform)
```

# Index

## \* datasets

PC, [7](#)

sce, [9](#)

BiocSingularParam, [12](#)

ClusterPlot, [2](#)

find\_neighbors2, [3](#)

getneighborhood\_fast, [3](#)

ICMEM, [4](#)

parafun, [5](#)

PC, [7](#)

SC.MEB, [7](#)

sce, [9](#)

selectK, [9](#)

selectKPlot, [10](#)

spatialPreprocess, [11](#)