

# Package: twomodeclusteringGA (via r-universe)

May 14, 2026

**Title** Genetic Algorithm Based Two-Mode Clustering

**Version** 1.0.0

**Description** Implements two-mode clustering (biclustering) using genetic algorithms. The method was first introduced in Hageman et al. (2008) <[doi:10.1007/s11306-008-0105-7](https://doi.org/10.1007/s11306-008-0105-7)>. The package provides tools for fitting, visualization, and validation of two-mode cluster structures in data matrices.

**License** GPL-3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Depends** R (>= 3.6)

**Imports** GA, stats, utils, ggplot2

**URL** <https://github.com/joshageman/twomodeclusteringGA>

**BugReports** <https://github.com/joshageman/twomodeclusteringGA/issues>

**Repository** <https://joshageman.r-universe.dev>

**Date/Publication** 2025-09-10 19:05:00 UTC

**RemoteUrl** <https://github.com/joshageman/twomodeclusteringga>

**RemoteRef** HEAD

**RemoteSha** 9a89183606cd6015fdcee700a5b81115de5ef763

## Contents

as.data.frame.twomodeClustering . . . . .	2
gaintegerMutation . . . . .	3
gaintegerOnePointCrossover . . . . .	3
gaintegerPopulation . . . . .	4
gaintegerTwoPointCrossover . . . . .	4

monitorFactory . . . . .	5
plotTwomodeClustering . . . . .	5
print.summary.twomodeClustering . . . . .	7
print.twomodeClustering . . . . .	8
summary.twomodeClustering . . . . .	9
twomodeClusteringGA . . . . .	10
twomodeFitnessFactory . . . . .	12
twomodeToy . . . . .	13
validateTwomodePartition . . . . .	13

## Index 16

---

as.data.frame.twomodeClustering  
*Convert a twomodeClustering object to a data.frame*

---

### Description

This function creates a data.frame representation of a twomodeClustering object, listing the cluster assignments for both rows and columns.

### Usage

```
## S3 method for class 'twomodeClustering'
as.data.frame(x, row.names = NULL, optional = FALSE, myMatrix = NULL, ...)
```

### Arguments

x	An object of class 'twomodeClustering'.
row.names	Optional vector of row names for the resulting data.frame.
optional	Logical. If TRUE, allows optional parameters for data.frame.
myMatrix	Optional matrix to provide row and column names.
...	Additional arguments (currently ignored).

### Value

A data.frame with columns: name, type (row/col), and cluster assignment.

---

gaintegerMutation	<i>Integer mutation for genetic algorithm</i>
-------------------	---

---

**Description**

Performs mutation on a genetic algorithm individual by randomly changing cluster assignments with a specified probability.

**Usage**

```
gaintegerMutation(object, parent, ...)
```

**Arguments**

object	GA object containing algorithm parameters.
parent	Integer index of the parent individual to mutate.
...	Additional arguments (not used).

**Value**

Numeric vector representing the mutated individual.

---

gaintegerOnePointCrossover	<i>One-point crossover for genetic algorithm with integer encoding</i>
----------------------------	--

---

**Description**

Performs one-point crossover between two parent individuals in the genetic algorithm, exchanging genetic material at a single randomly selected point.

**Usage**

```
gaintegerOnePointCrossover(object, parents, ...)
```

**Arguments**

object	GA object containing algorithm parameters.
parents	Integer vector of length 2 containing indices of parent individuals.
...	Additional arguments (not used).

**Value**

List containing:

**children** Matrix with two rows representing the offspring  
**fitness** Vector of NA values (fitness will be calculated later)

---

`gaintegerPopulation`    *Integer population initialization for genetic algorithm*

---

### Description

Generates an initial population for the genetic algorithm where each individual represents a clustering solution with integer cluster assignments.

### Usage

```
gaintegerPopulation(object, ...)
```

### Arguments

<code>object</code>	GA object containing algorithm parameters.
<code>...</code>	Additional arguments (not used).

### Value

Matrix where each row represents an individual in the population and each column represents a cluster assignment.

---

`gaintegerTwoPointCrossover`  
*Two-point crossover for genetic algorithm with integer encoding*

---

### Description

Performs two-point crossover between two parent individuals in the genetic algorithm, exchanging genetic material between two randomly selected points.

### Usage

```
gaintegerTwoPointCrossover(object, parents, ...)
```

### Arguments

<code>object</code>	GA object containing algorithm parameters.
<code>parents</code>	Integer vector of length 2 containing indices of parent individuals.
<code>...</code>	Additional arguments (not used).

### Value

List containing:

**children** Matrix with two rows representing the offspring  
**fitness** Vector of NA values (fitness will be calculated later)

---

monitorFactory	<i>Two mode clustering monitoring function factory for GA progress</i>
----------------	--

---

### Description

Creates a monitoring function that prints the current generation and the best fitness score to the console at specified intervals. Intended for use as a monitor function in GA runs.

### Usage

```
monitorFactory(interval = 100)
```

### Arguments

interval	An integer specifying the interval for printing progress updates. Default is 100 (prints every 100 generations).
----------	--

### Value

A monitoring function that can be used with GA. The returned function takes a GA object and prints progress information at the specified interval.

### Examples

```
# Create monitor that prints every 100 generations (default)
monitor <- monitorFactory()
# ga(..., monitor = monitor)

# Create monitor that prints every 50 generations
monitor <- monitorFactory(50)
# ga(..., monitor = monitor)
```

---

plotTwomodeClustering	<i>Plot two-mode clustering results (validation-aware, compact labels)</i>
-----------------------	--

---

### Description

Heatmap of the clustered matrix with clear cluster boundaries. If `result$validation` is present, each block shows one label with the chosen value plus significance stars.

**Usage**

```

plotTwomodeClustering(
  myMatrix,
  result,
  title = "",
  xlabel = "",
  ylabel = "",
  varOrder = 0,
  objOrder = 0,
  palette = c("diverging", "viridis", "grey"),
  showBoundaries = TRUE,
  boundaryColor = "white",
  boundarySize = 1,
  showMeans = TRUE,
  fixAspect = TRUE,
  showValidation = TRUE,
  value = c("mean", "standardized", "effectSS"),
  digits = 2,
  sigLevels = c(0.001, 0.01, 0.05, 0.1),
  showMarginal = TRUE,
  labelColor = "white",
  showGlobal = TRUE
)

```

**Arguments**

<code>myMatrix</code>	Numeric matrix or coercible data.frame with the data.
<code>result</code>	Result from <code>twomodeClusteringGA()</code> , with <code>rowClusters</code> , <code>colClusters</code> , and optionally validation.
<code>title</code>	Text for title.
<code>xlabel</code>	Text for x-axis label.
<code>ylabel</code>	Text for y-axis label.
<code>varOrder</code>	Order of column clusters (0 = automatic).
<code>objOrder</code>	Order of row clusters (0 = automatic).
<code>palette</code>	Color scale: "diverging", "viridis", or "grey".
<code>showBoundaries</code>	Logical; show cluster boundaries.
<code>boundaryColor</code>	Color of the boundaries.
<code>boundarySize</code>	Width of the boundaries.
<code>showMeans</code>	Logical; show block labels (value + stars if validation).
<code>fixAspect</code>	Logical; square cells.
<code>showValidation</code>	Logical; use validation information if available.
<code>value</code>	Which block statistic to label: "mean", "standardized", or "effectSS". For "standardized", $\text{sign}(\text{mean}) * \sqrt{\text{chi}^2_1}$ is shown if validation is available.
<code>digits</code>	Number of decimals in the label.

sigLevels	Thresholds for stars: c(0.001, 0.01, 0.05, 0.1).
showMarginal	Logical; show "." for $p < 0.1$ .
labelColor	Color of the block labels.
showGlobal	Logical; add global validation (R2, F, p, p_MC) to subtitle.

**Value**

A ggplot object.

**Examples**

```
data("twomodeToy")
myMatrix_s <- scale(twomodeToy)

#Run the GA-based two-mode clustering
result <- twomodeClusteringGA(
  myMatrix = myMatrix_s,
  nRowClusters = 2,
  nColClusters = 3,
  seeds = 1,
  maxiter = 200,
  popSize = 30,
  elitism = 1,
  validate = TRUE,
  verbose = TRUE
)

#Inspect the result
print(result)
summary(result)
myTwomodeResult <- as.data.frame(result)
head(myTwomodeResult)

#Plot the clustered heatmap
plotTwomodeClustering(
  myMatrix = myMatrix_s,
  result = result,
  title = "Two-mode clustering Toy example",
  fixAspect = FALSE
)
```

---

```
print.summary.twomodeClustering
```

*Print method for summary.twomodeClustering objects*

---

**Description**

Prints key information about a two-mode clustering result, including matrix dimensions, cluster sizes, fitness, and (if available) validation highlights.

**Usage**

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

**Arguments**

x	An object of class 'summary.twomodeClustering'.
...	Additional arguments (currently ignored).

**Value**

Invisibly returns x.

---

```
print.twomodeClustering
```

*Print method for twomodeClustering objects*

---

**Description**

Prints a concise summary of a twomodeClustering object, including matrix dimensions, cluster counts, fitness, and (if available) validation results.

**Usage**

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

**Arguments**

x	An object of class 'twomodeClustering'.
...	Additional arguments (currently ignored).

**Value**

Invisibly returns x.

---

summary.twomodeClustering

*Summary method for twomodeClustering objects*


---

## Description

Creates a summary of a twomodeClustering object, including matrix dimensions, cluster sizes, fitness, optional bicluster summaries (if matrix available), and optional validation highlights (if validation is present).

## Usage

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

## Arguments

**object**            An object of class 'twomodeClustering'.  
**...**             Additional arguments (currently ignored).

## Value

An object of class summary.twomodeClustering with components:

**matrixDim** Named integer vector: rows, cols

**nRowClusters** Number of row clusters

**nColClusters** Number of column clusters

**rowClusterSizes** Table of row cluster sizes

**colClusterSizes** Table of column cluster sizes

**biclusters** Data frame with bicluster summaries (if myMatrix present), possibly merged with validation per-block stats

**fitness** Best fitness value if available, else NA

**validationGlobal** List with r2, fStat, pValue, dfModel, dfResid, pMonteCarlo (if present), or NULL

**nSigBlocks** Number of BH-significant blocks at 0.05 if available, else NULL

**rowContribution** Data frame with total effectSS per row cluster (if available), else NULL

**colContribution** Data frame with total effectSS per column cluster (if available), else NULL

---

twomodeClusteringGA    *Two-mode clustering using genetic algorithm (with optional validation)*

---

### Description

Performs two-mode clustering on a numeric matrix using a genetic algorithm. The algorithm simultaneously clusters rows and columns to minimize within-cluster sum of squared errors (SSE). Optionally, a validation step is executed that tests the statistical significance of the found partition using `validateTwomodePartition()`.

### Usage

```
twomodeClusteringGA(
  myMatrix,
  nColClusters,
  nRowClusters,
  seeds = 1:5,
  verbose = FALSE,
  maxiter = 2000,
  popSize = 300,
  pmutation = 0.05,
  pcrossover = 0.5,
  elitism = 100,
  interval = 100,
  parallel = FALSE,
  run = NULL,
  validate = FALSE,
  validateCenter = TRUE,
  validatePerBlock = TRUE,
  validateMonteCarlo = 0L,
  validateFixBlockSizes = TRUE,
  validateStoreNull = FALSE,
  validateSeed = NULL
)
```

### Arguments

<code>myMatrix</code>	Numeric matrix or <code>data.frame</code> to be clustered. Must be coercible to numeric.
<code>nColClusters</code>	Integer. Number of column clusters to form.
<code>nRowClusters</code>	Integer. Number of row clusters to form.
<code>seeds</code>	Integer vector. Random seeds for multiple GA runs. Default is 1:5.
<code>verbose</code>	Logical. If TRUE, prints progress information. Default is FALSE.
<code>maxiter</code>	Integer. Maximum number of GA iterations. Default is 2000.
<code>popSize</code>	Integer. Population size for the GA. Default is 300.

<code>pmutation</code>	Numeric. Probability of mutation (0-1). Default is 0.05.
<code>pcrossover</code>	Numeric. Probability of crossover (0-1). Default is 0.5.
<code>elitism</code>	Integer. Number of best individuals to preserve. Default is 100. If NULL, uses 5% of popSize.
<code>interval</code>	Integer. Interval for progress monitoring when verbose=TRUE. Default is 100.
<code>parallel</code>	Logical. Whether to use parallel processing. Default is FALSE.
<code>run</code>	Integer. Number of consecutive generations without improvement before stopping. If NULL, runs for full maxiter iterations.
<code>validate</code>	Logical. If TRUE, run validation on the best partition and attach results under \$validation. Default FALSE.
<code>validateCenter</code>	Logical. Passed to validateTwomodePartition(center=...). Default TRUE.
<code>validatePerBlock</code>	Logical. Passed to validateTwomodePartition(perBlock=...). Default TRUE.
<code>validateMonteCarlo</code>	Integer. Number of random partitions for MC p-value. Passed to validateTwomodePartition(monteCarlo=...). Default 0 (disabled).
<code>validateFixBlockSize</code>	Logical. Keep observed cluster sizes in MC. Default TRUE.
<code>validateStoreNull</code>	Logical. Store full null vector from MC. Default FALSE.
<code>validateSeed</code>	Optional integer seed for the validation step. Default NULL.

## Details

The function runs multiple GA instances with different random seeds and returns the best solution. The fitness function minimizes the sum of squared errors within clusters. Row and column clusters are optimized simultaneously.

## Value

A list of class "twomodeClustering" containing:

- bestGa** The best GA object from all runs
- bestFitness** Best fitness value achieved (negative SSE)
- bestSeed** Seed that produced the best result
- rowClusters** Integer vector of row cluster assignments
- colClusters** Integer vector of column cluster assignments
- control** List of control parameters used
- validation** List returned by validateTwomodePartition() if validate=TRUE; otherwise NULL

## References

Hageman, J. A., van den Berg, R. A., Westerhuis, J. A., van der Werf, M. J., & Smilde, A. K. (2008). Genetic algorithm based two-mode clustering of metabolomics data. *Metabolomics*, 4, 141–149. doi:10.1007/s1130600801057

**See Also**

[ga](#) for the underlying genetic algorithm implementation

**Examples**

```
data("twomodeToy")
myMatrix_s <- scale(twomodeToy)

#Run the GA-based two-mode clustering
result <- twomodeClusteringGA(
  myMatrix = myMatrix_s,
  nRowClusters = 2,
  nColClusters = 3,
  seeds = 1,
  maxiter = 200,
  popSize = 30,
  elitism = 1,
  validate = TRUE,
  verbose = TRUE
)

#Inspect the result
print(result)
summary(result)
myTwomodeResult <- as.data.frame(result)
head(myTwomodeResult)

#Plot the clustered heatmap
plotTwomodeClustering(
  myMatrix = myMatrix_s,
  result = result,
  title = "Two-mode clustering Toy example",
  fixAspect = FALSE
)
```

---

twomodeFitnessFactory *Two-mode clustering genetic algorithm evaluation function (fast, robust)*

---

**Description**

Fast evaluation of a two-mode clustering solution.

**Usage**

```
twomodeFitnessFactory(myMatrix)
```

**Arguments**

myMatrix            Numeric matrix or coercible data.frame.

**Value**

Function(string, ...) -> numeric fitness value = negative SSE (higher is better).

---

twomodeToy	<i>Toy matrix with one multiplicative and one additive bicluster</i>
------------	--

---

**Description**

A small 12×9 matrix with a 2 x 3 two-mode cluster structure to demonstrate twomodeclusteringGA in a controlled setting.

**Usage**

```
data(twomodeToy)
```

**Format**

A numeric matrix of dimension  $12 \times 9$  with a 2 x 3 two-mode cluster structure

**Examples**

```
data("twomodeToy")
str(twomodeToy)
image(t(twomodeToy))
```

---

```
validateTwomodePartition
```

*Validate a two-mode clustering partition by global and per-block significance*

---

**Description**

Given a numeric matrix and a full two-mode partition (exclusive row and column clusters), this function tests whether the fitted block-means model explains more structure than expected under a no-structure null. The global test uses an F-statistic based on `SS_fit` and `SSE` derived from your fitness definition. Optionally, it also reports per-block chi-square tests and a fast Monte Carlo p-value using random partitions (no GA reruns).

**Usage**

```
validateTwomodePartition(
  myMatrix,
  rowClusters,
  colClusters,
  center = TRUE,
  perBlock = TRUE,
  monteCarlo = 0,
  fixBlockSizes = TRUE,
  storeNull = FALSE,
  seed = NULL
)
```

**Arguments**

myMatrix	Numeric matrix or coercible data.frame.
rowClusters	Integer vector of length nrow(myMatrix) with cluster labels (1..kR, arbitrary labels allowed).
colClusters	Integer vector of length ncol(myMatrix) with cluster labels (1..kC, arbitrary labels allowed).
center	Logical, center the matrix by its global mean before testing (default TRUE). Centering aligns the null with zero-mean noise and generally stabilizes inference.
perBlock	Logical, compute per-block tests (default TRUE).
monteCarlo	Integer, number of random partitions to draw for a MC p-value (default 0 disables).
fixBlockSizes	Logical, if TRUE keep row and column cluster sizes equal to the observed sizes when generating random partitions (default TRUE). If FALSE, only kR and kC are fixed.
storeNull	Logical, store the vector of null F statistics from random partitions (default FALSE). If FALSE, only quantiles are stored.
seed	Optional integer seed for reproducibility (default NULL).

**Value**

A list of class "twomodeValidation" with elements:

- nR, nC, kR, kC
- dfModel, dfResid
- ssTot, ssFit, sse, sigma2Hat, r2
- fStat, pValue (global F test)
- perBlock (data.frame with per-block stats) if perBlock=TRUE
- mc (list with nSim, pMonteCarlo, fNull or fNullQuantiles) if monteCarlo>0

**Examples**

```
data("twomodeToy")
myMatrix_s <- scale(twomodeToy)

#Run the GA-based two-mode clustering
result <- twomodeClusteringGA(
  myMatrix = myMatrix_s,
  nRowClusters = 2,
  nColClusters = 3,
  seeds = 1,
  maxiter = 200,
  popSize = 30,
  elitism = 1,
  validate = FALSE,
  verbose = TRUE
)

result$validation <- validateTwomodePartition(myMatrix_s,
                                             rowClusters=result$rowClusters,
                                             colClusters=result$colClusters)

#Inspect the result
print(result)
summary(result)
myTwomodeResult <- as.data.frame(result)
head(myTwomodeResult)

#Plot the clustered heatmap
plotTwomodeClustering(
  myMatrix = myMatrix_s,
  result = result,
  title = "Two-mode clustering Toy example",
  fixAspect = FALSE
)
```

# Index

## \* datasets

twomodeToy, [13](#)

as.data.frame.twomodeClustering, [2](#)

ga, [12](#)

gaintegerMutation, [3](#)

gaintegerOnePointCrossover, [3](#)

gaintegerPopulation, [4](#)

gaintegerTwoPointCrossover, [4](#)

monitorFactory, [5](#)

plotTwomodeClustering, [5](#)

print.summary.twomodeClustering, [7](#)

print.twomodeClustering, [8](#)

summary.twomodeClustering, [9](#)

twomodeClusteringGA, [10](#)

twomodeFitnessFactory, [12](#)

twomodeToy, [13](#)

validateTwomodePartition, [13](#)