# Package: clusternomics (via r-universe)

October 31, 2024

**Type** Package

**Title** Integrative Clustering for Heterogeneous Biomedical Datasets

**Version** 0.1.1

**Author** Evelina Gabasova

**Maintainer** Evelina Gabasova <egabasova@gmail.com>

**Description** Integrative context-dependent clustering for heterogeneous biomedical datasets. Identifies local clustering structures in related datasets, and a global clusters that exist across the datasets.

**License** MIT + file LICENSE

**LazyData** TRUE

**Imports** magrittr, plyr, MASS

**Suggests** knitr, rmarkdown, testthat, mclust, gplots, ggplot2

**URL** https://github.com/evelinag/clusternomics

**BugReports** https://github.com/evelinag/clusternomics/issues

**RoxygenNote** 5.0.1

**VignetteBuilder** knitr

**Repository** https://evelinag.r-universe.dev

**RemoteUrl** https://github.com/evelinag/clusternomics

**RemoteRef** HEAD

**RemoteSha** ebccead47a895d513a8e4641acb49f7638f17151

## Contents

---

`clusterSizes` *Estimate sizes of clusters from global cluster assignments.*

---

## Description

Estimate sizes of clusters from global cluster assignments.

## Usage

```
clusterSizes(assignments)
```

## Arguments

| | |
|---|---|
| `assignments` | Matrix of cluster assignments, where each row corresponds to cluster assignments sampled in one MCMC iteration |

## Value

Sizes of individual clusters in each MCMC iteration.

## Examples

```
# Generate simple test dataset
groupCounts <- c(50, 10, 40, 60)
means <- c(-1.5,1.5)
testData <- generateTestData_2D(groupCounts, means)
datasets <- testData$data

# Fit the model
# 1. specify number of clusters
clusterCounts <- list(global=10, context=c(3,3))
# 2. Run inference
# Number of iterations is just for demonstration purposes, use
# a larger number of iterations in practice!
results <- contextCluster(datasets, clusterCounts,
     maxIter = 10, burnin = 5, lag = 1,
     dataDistributions = 'diagNormal',
     verbose = TRUE)

# Extract only the sampled global assignments
samples <- results$samples
clusters <- plyr::laply(1:length(samples), function(i) samples[[i]]$Global)
clusterSizes(clusters)
```

`coclusteringMatrix` *Compute the posterior co-clustering matrix from global cluster assignments.*

## Description

Compute the posterior co-clustering matrix from global cluster assignments.

## Usage

```
coclusteringMatrix(assignments)
```

## Arguments

| | |
|---|---|
| `assignments` | Matrix of cluster assignments, where each row corresponds to cluster assignments sampled in one MCMC iteration |

## Value

Posterior co-clustering matrix, where element `[i,j]` represents the posterior probability that data points `i` and `j` belong to the same cluster.

## Examples

```
# Generate simple test dataset
groupCounts <- c(50, 10, 40, 60)
means <- c(-1.5,1.5)
testData <- generateTestData_2D(groupCounts, means)
datasets <- testData$data

# Fit the model
# 1. specify number of clusters
clusterCounts <- list(global=10, context=c(3,3))
# 2. Run inference
# Number of iterations is just for demonstration purposes, use
# a larger number of iterations in practice!
results <- contextCluster(datasets, clusterCounts,
     maxIter = 10, burnin = 5, lag = 1,
     dataDistributions = 'diagNormal',
     verbose = TRUE)

# Extract only the sampled global assignments
samples <- results$samples
clusters <- plyr::laply(1:length(samples), function(i) samples[[i]]$Global)
coclusteringMatrix(clusters)
```

`contextCluster` *Clusternomics: Context-dependent clustering*

## Description

This function fits the context-dependent clustering model to the data using Gibbs sampling. It allows the user to specify a different number of clusters on the global level, as well as on the local level.

## Usage

```
contextCluster(datasets, clusterCounts, dataDistributions = "diagNormal",
  prior = NULL, maxIter = 1000, burnin = NULL, lag = 3,
  verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `datasets` | List of data matrices where each matrix represents a context-specific dataset. Each data matrix has the size $N$ times $M$, where $N$ is the number of data points and $M$ is the dimensionality of the data. The full list of matrices has length $C$. The number of data points $N$ must be the same for all data matrices. |
| `clusterCounts` | Number of cluster on the global level and in each context. List with the following structure: `clusterCounts = list(global=global, context=context)` where `global` is the number of global clusters, and `context` is the list of numbers of clusters in the individual contexts (datasets) of length $C$ where `context[c]` is the number of clusters in dataset $c$. |
| `dataDistributions` | Distribution of data in each dataset. Can be either a list of length $C$ where `dataDistributions[c]` is the distribution of dataset $c$, or a single string when all datasets have the same distribution. Currently implemented distribution is the `'diagNormal'` option for multivariate Normal distribution with diagonal covariance matrix. |
| `prior` | Prior distribution. If `NULL` then the prior is estimated using the datasets. The `'diagNormal'` distribution uses the Normal-Gamma distribution as a prior for each dimension. |
| `maxIter` | Number of iterations of the Gibbs sampling algorithm. |
| `burnin` | Number of burn-in iterations that will be discarded. If not specified, the algorithm discards the first half of the `maxIter` samples. |
| `lag` | Used for thinning the samples. |
| `verbose` | Print progress, by default `FALSE`. |

## Value

Returns list containing the sequence of MCMC states and the log likelihoods of the individual states.

| | |
|---|---|
| samples | List of assignments sampled from the posterior, each state samples[[i]] is a data frame with local and global assignments for each data point. |
| logliks | Log likelihoods during MCMC iterations. |
| DIC | Deviance information criterion to help select the number of clusters. Lower values of DIC correspond to better-fitting models. |

## Examples

```
# Example with simulated data (see vignette for details)
# Number of elements in each cluster
groupCounts <- c(50, 10, 40, 60)
# Centers of clusters
means <- c(-1.5,1.5)
testData <- generateTestData_2D(groupCounts, means)
datasets <- testData$data

# Fit the model
# 1. specify number of clusters
clusterCounts <- list(global=10, context=c(3,3))
# 2. Run inference
# Number of iterations is just for demonstration purposes, use
# a larger number of iterations in practice!
results <- contextCluster(datasets, clusterCounts,
     maxIter = 10, burnin = 5, lag = 1,
     dataDistributions = 'diagNormal',
     verbose = TRUE)

# Extract results from the samples
# Final state:
state <- results$samples[[length(results$samples)]]
# 1) assignment to global clusters
globalAssgn <- state$Global
# 2) context-specific assignmnets- assignment in specific dataset (context)
contextAssgn <- state[,"Context 1"]
# Assess the fit of the model with DIC
results$DIC
```

---

# empiricalBayesPrior *Fit an empirical Bayes prior to the data*

---

## Description

Fit an empirical Bayes prior to the data

## Usage

```
empiricalBayesPrior(datasets, distributions = "diagNormal",
  globalConcentration = 0.1, localConcentration = 0.1, type = "fitRate")
```

## Arguments

| | |
|---|---|
| `datasets` | List of data matrices where each matrix represents a context-specific dataset. Each data matrix has the size $N$ times $M$, where $N$ is the number of data points and $M$ is the dimensionality of the data. The full list of matrices has length $C$. The number of data points $N$ must be the same for all data matrices. |
| `distributions` | Distribution of data in each dataset. Can be either a list of length $C$ where `dataDistributions[c]` is the distribution of dataset $c$, or a single string when all datasets have the same distribution. Currently implemented distribution is the `'diagNormal'` option for multivariate Normal distribution with diagonal covariance matrix. |
| `globalConcentration` | Prior concentration parameter for the global clusters. Small values of this parameter give larger prior probability to smaller number of clusters. |
| `localConcentration` | Prior concentration parameter for the local context-specific clusters. Small values of this parameter give larger prior probability to smaller number of clusters. |
| `type` | Type of prior that is fitted to the data. The algorithm can fit either rate of the prior covariance matrix, or fit the full covariance matrix to the data. |

## Value

Returns the prior object that can be used as an input for the `contextCluster` function.

## Examples

```
# Example with simulated data (see vignette for details)
nContexts <- 2
# Number of elements in each cluster
groupCounts <- c(50, 10, 40, 60)
# Centers of clusters
means <- c(-1.5,1.5)
testData <- generateTestData_2D(groupCounts, means)
datasets <- testData$data

# Generate the prior
fullDataDistributions <- rep('diagNormal', nContexts)
prior <- empiricalBayesPrior(datasets, fullDataDistributions, 0.01, 0.1, 'fitRate')

# Fit the model
# 1. specify number of clusters
clusterCounts <- list(global=10, context=c(3,3))
# 2. Run inference
# Number of iterations is just for demonstration purposes, use
# a larger number of iterations in practice!
results <- contextCluster(datasets, clusterCounts,
     maxIter = 10, burnin = 5, lag = 1,
     dataDistributions = 'diagNormal', prior = prior,
     verbose = TRUE)
```

generatePrior *Generate a basic prior distribution for the datasets.*

## Description

Creates a basic prior distribution for the clustering model, assuming a unit prior covariance matrix for clusters in each dataset.

## Usage

```
generatePrior(datasets, distributions = "diagNormal",
  globalConcentration = 0.1, localConcentration = 0.1)
```

## Arguments

| | |
|---|---|
| datasets | List of data matrices where each matrix represents a context-specific dataset. Each data matrix has the size $N$ times $M$, where $N$ is the number of data points and $M$ is the dimensionality of the data. The full list of matrices has length $C$. The number of data points $N$ must be the same for all data matrices. |
| distributions | Distribution of data in each dataset. Can be either a list of length $C$ where `dataDistributions[c]` is the distribution of dataset $c$, or a single string when all datasets have the same distribution. Currently implemented distribution is the `'diagNormal'` option for multivariate Normal distribution with diagonal covariance matrix. |
| globalConcentration | Prior concentration parameter for the global clusters. Small values of this parameter give larger prior probability to smaller number of clusters. |
| localConcentration | Prior concentration parameter for the local context-specific clusters. Small values of this parameter give larger prior probability to smaller number of clusters. |

## Value

Returns the prior object that can be used as an input for the `contextCluster` function.

## Examples

```
# Example with simulated data (see vignette for details)
nContexts <- 2
# Number of elements in each cluster
groupCounts <- c(50, 10, 40, 60)
# Centers of clusters
means <- c(-1.5,1.5)
testData <- generateTestData_2D(groupCounts, means)
datasets <- testData$data

# Generate the prior
fullDataDistributions <- rep('diagNormal', nContexts)
```

```
prior <- generatePrior(datasets, fullDataDistributions, 0.01, 0.1)

# Fit the model
# 1. specify number of clusters
clusterCounts <- list(global=10, context=c(3,3))
# 2. Run inference
# Number of iterations is just for demonstration purposes, use
# a larger number of iterations in practice!
results <- contextCluster(datasets, clusterCounts,
     maxIter = 10, burnin = 5, lag = 1,
     dataDistributions = 'diagNormal', prior = prior,
     verbose = TRUE)
```

---

`generateTestData_1D` *Generate simulated 1D dataset for testing*

---

## Description

Generate simple 1D dataset with two contexts, where the data are generated from Gaussian distributions. The generated output contains two datasets, where each dataset contains 4 global clusters, originating from two local clusters in each context.

## Usage

```
generateTestData_1D(groupCounts, means)
```

## Arguments

| | |
|---|---|
| `groupCounts` | Number of data samples in each global cluster. It is assumed to be a vector of four elements: `c(c11, c21, c12, c22)` where `cij` is the number of samples coming from cluster i in context 1 and cluster j in context 2. |
| `means` | Means of the simulated clusters. It is assumed to be a vector of two elements: `c(m1, m2)` where `m1` is the mean of the first cluster in both contexts, and `m2` is the mean of the second cluster in both contexts. |

## Value

Returns the simulated datasets together with true assignmets.

| | |
|---|---|
| `data` | List of datasets for each context. This can be used as an input for the `contextCluster` function. |
| `groups` | True cluster assignments that were used to generate the data. |

## Examples

```
groupCounts <- c(50, 10, 40, 60)
means <- c(-1.5,1.5)
testData <- generateTestData_1D(groupCounts, means)
# Use the dataset as an input for the contextCluster function for testing
datasets <- testData$data
```

---

`generateTestData_2D` *Generate simulated 2D dataset for testing*

---

## Description

Generate simple 2D dataset with two contexts, where the data are generated from Gaussian distributions. The generated output contains two datasets, where each dataset contains 4 global clusters, originating from two local clusters in each context.

## Usage

```
generateTestData_2D(groupCounts, means, variances = NULL)
```

## Arguments

| | |
|---|---|
| `groupCounts` | Number of data samples in each global cluster. It is assumed to be a vector of four elements: `c(c11, c21, c12, c22)` where `cij` is the number of samples coming from cluster i in context 1 and cluster j in context 2. |
| `means` | Means of the simulated clusters. It is assumed to be a vector of two elements: `c(m1, m2)` where `m1` is the mean of the first cluster in both contexts, and `m2` is the mean of the second cluster in both contexts. Because the data are two-dimensional, the mean is assumed to be the same in both dimensions. |
| `variances` | Optionally, it is possible to specify different variance for each of the clusters. The variance is assumed to be a vector of two elements: `c(v1, v2)` where `v1` is the variance of the first cluster in both contexts, and `v2` is the variance of the second cluster in both contexts. Because the data are two-dimensional, the variance is diagonal and the same in both dimensions. |

## Value

Returns the simulated datasets together with true assignmets.

| | |
|---|---|
| `data` | List of datasets for each context. This can be used as an input for the `contextCluster` function. |
| `groups` | True cluster assignments that were used to generate the data. |

## Examples

```
groupCounts <- c(50, 10, 40, 60)
means <- c(-1.5,1.5)
testData <- generateTestData_1D(groupCounts, means)
# Use the dataset as an input for the contextCluster function for testing
datasets <- testData$data
```

---

`numberOfClusters` *Estimate number of clusters from global cluster assignments.*

---

## Description

Estimate number of clusters from global cluster assignments.

## Usage

```
numberOfClusters(assignments)
```

## Arguments

| | |
|---|---|
| `assignments` | Matrix of cluster assignments, where each row corresponds to cluster assignments sampled in one MCMC iteration |

## Value

Number of unique clusters in each MCMC iteration.

## Examples

```
# Generate simple test dataset
groupCounts <- c(50, 10, 40, 60)
means <- c(-1.5,1.5)
testData <- generateTestData_2D(groupCounts, means)
datasets <- testData$data

# Fit the model
# 1. specify number of clusters
clusterCounts <- list(global=10, context=c(3,3))
# 2. Run inference
# Number of iterations is just for demonstration purposes, use
# a larger number of iterations in practice!
results <- contextCluster(datasets, clusterCounts,
     maxIter = 10, burnin = 5, lag = 1,
     dataDistributions = 'diagNormal',
     verbose = TRUE)

# Extract only the sampled global assignments
samples <- results$samples
```

```
clusters <- plyr::laply(1:length(samples), function(i) samples[[i]]$Global)
numberOfClusters(clusters)
```

# Index