Package ‘INSPIRE’

December 9, 2016

Type Package

Title Inferring Shared Modules from Multiple Gene Expression Datasets with Partially Overlapping Gene Sets

Version 1.5

Date 2016-12-08

Author Safiye Celik

Maintainer Safiye Celik <safiye@cs.washington.edu>

Description A method to infer modules of co-expressed genes and the dependencies among the modules from multiple expression datasets that may contain different sets of genes. Please refer to: Extracting a low-dimensional description of multiple gene expression datasets reveals a potential driver for tumor-associated stroma in ovarian cancer, Safiye Celik, Benjamin A. Logsdon, Stephanie Battle, Charles W. Drescher, Mara Rendi, R. David Hawkins and Su-In Lee (2016) <DOI:10.1186/s13073-016-0319-7>.

License GPL (>= 2)

URL inspire.cs.washington.edu

Imports missMDA

RoxygenNote 5.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-12-09 22:52:45

R topics documented:

`exmp_dataset1` *Example Gene Expression Dataset-1*

Description

This example ovarian cancer dataset contains expression of random half of the genes on the 28 samples from the GSE19829.GPL570 accession in Gene Expression Omnibus. Contains 28 samples (as rows) and 9056 genes (as columns). 4117 of the genes are overlapping with the genes in exmp_dataset2.

`exmp_dataset2` *Example Gene Expression Dataset-2*

Description

This example ovarian cancer dataset contains expression of random half of the genes on the 42 samples from the GSE19829.GPL8300 accession in Gene Expression Omnibus. Contains 42 samples (as rows) and 4165 genes (as columns). 4117 of the genes are overlapping with the genes in exmp_dataset1.

`INSPIRE` *Inferring Shared Modules from Multiple Gene Expression Datasets with Partially Overlapping Gene Sets*

Description

Takes a list of data matrices, with potentially different number of genes, number of modules, and a penalty parameter, and returns the final assignment of the data points in each dataset to the modules, the values of the module latent variables, and the conditional dependency network among the module latent variables.

Usage

```
INSPIRE(datasetlist, mcnt, lambda, printoutput = 0, maxinitKMiter = 100,
  maxiter = 100, threshold = 0.01, initseed = 123)
```

Arguments

`datasetlist` A list of gene expression matrices of size n_i x p_i where rows represent samples and columns represent genes for each dataset i. This can be created by using the list() command, e.g., list(dataset1, dataset2, dataset3)

`mcnt` A positive integer representing the number of modules to learn from the data

`lambda` A penalty parameter that regularizes the estimated precision matrix representing the conditional dependencies among the modules

`printoutput` 0 or 1 representing whether the progress of the algorithm should be displayed (0 means no display which is the default)

`maxinitKMiter` Maximum number of K-means iterations performed to initialize the parameters (the default is 100 iterations)

`maxiter` Maximum number of INSPIRE iterations performed to update the parameters (the default is 100 iterations)

`threshold` Convergence threshold measured as the relative change in the sum of the elements of the estimated precision matrices in two consecutive iterations (the default is 10^-2)

`initseed` The random seed set right before the K-means call which is performed to initialize the parameters

Value

`L` A matrix of size (sum_n_i) x mcnt representing the inferred latent variables (the low-dimensional representation - or LDR - of the data)

`Z` A list of vectors of size p_i representing the learned assignment of each of the genes in each dataset i to one of mcnt modules

`theta` Estimated precision matrix of size mcnt x mcnt representing the conditional dependencies among the modules

Examples

```
## Not run:
library(INSPIRE)
mcnt = 90 #module size
lambda = .1 #penalty parameter to induce sparsity
# download two real gene expression datasets, where the rows are genes and columns are samples
data('two_example_datasets')
# log-normalize, and standardize each dataset
res = INSPIRE(list(scale(log(exmp_dataset1)), scale(log(exmp_dataset2))), mcnt, lambda)

## End(Not run)
```

Index