Package 'ASSIGN'

October 7, 2014

Type Package

Title Adaptive Signature Selection and InteGratioN (ASSIGN)

Version 1.0.0

Date 2013-09-29

Author Ying Shen, Andrea H. Bild, and W. Evan Johnson

Maintainer Ying Shen <yshen3@bu.edu>

Depends Rlab, msm, gplots

Description ASSIGN is a computational tool to evaluate the pathway deregulation/activation status in individual patient samples. ASSIGN employs a flexible Bayesian factor analysis approach that adapts predetermined pathway signatures derived either from knowledge-based literatures or from perturbation experiments to the cell-/tissue-specific pathway signatures. The deregulation/activation level of each context-specific pathway is quantified to a score, which represents the extent to which a patient sample encompasses the pathway deregulation/activation signature.

License MIT

Imports graphics, grDevices, stats, utils

biocViews Software, GeneExpression, Pathways, Bayesian

R topics documented:

`assign.convergence` *Check the convergence of the MCMC chain*

Description

The assign.convergence checks the convergence of the MCMC chain of the model parameters generated by the Gibbs sampling algorithm.

Usage

```
assign.convergence(test, burn_in=0, iter=2000, parameter = c("B", "S", "Delta",
"beta", "kappa", "gamma", "sigma"), whichGene, whichSample, whichPath)
```

Arguments

`test`	The list object returned from the assign.mcmc function. The list components are the MCMC chains of the B, S, Delta, beta, gamma, and sigma.
`burn_in`	The number of burn-in iterations. These iterations are discarded when computing the posterior means of the model parameters. The default is 0.
`iter`	The number of total iterations. The default is 2000.
`parameter`	A charater string indicating which model parameter is to be checked for convergency. This must be one of "B", "S", "Delta", "beta", "kappa", "gamma", and "sigma".
`whichGene`	A numerical value indicating which gene is to be checked for convergence. The value has to be in the range between 1 and G.
`whichSample`	A numerical value indicating which test sample is to be checked for convergence. The value has to be in the range between 1 and N.
`whichPath`	A numerical value indicating which pathway is to be checked for convergence. The value has to be in the range between 1 and K.

Details

To compute the convergency of the gth gene in B, set whichGene=g, whichSample=NA, whichPath=NA.

To compute the convergency of the gth gene in the kth pathway within the signature matrix (S), set whichGene=g, whichSample=NA, whichPath=NA.

To compute the convergency of the kth pathway in the jth test sample within the pathway activation matrix (A), set whichGene=NA, whichSample=n, whichPath=k.

Value

The assign.convergence function returns the a vector of the estimated values from each Gibbs sampling iteration of the model parameter to be checked, and a trace plot of this parameter.

Author(s)

Ying Shen

Examples

```
## Not run:
# check the 10th gene in the 1st pathway for the convergency
trace.plot <- assign.convergence(test=mcmc.chain, burn_in=0, iter=2000, parameter="S",
whichGene=10, whichSample=NA, whichPath=1)

## End(Not run)
```

---

assign.cv.output *Cross validation output*

Description

The assign.cv.output function outputs the summary results and plots for the cross validation done on the training dataset.

Usage

```
assign.cv.output(processed.data, mcmc.pos.mean.trainingData, trainingData,
trainingLabel, adaptive_B=FALSE, adaptive_S=FALSE, mixture_beta=TRUE, outputDir)
```

Arguments

`processed.data`	The list object returned from the assign.preprocess function.
`mcmc.pos.mean.trainingData`	The list object returned from the assign.mcmc function. Notice that for cross validation, the Y argument in the assign.mcmc function should be set as the training dataset.
`trainingData`	The genomic measure matrix of training samples (i.g., gene expression matrix). The dimension of this matrix is probe number x sample number. The default is NULL.
`trainingLabel`	The list linking the index of each training sample to a specific group it belongs to.
`adaptive_B`	Logicals. If TRUE, the model adapts the baseline/background (B) of genomic measures for the test samples. The default is FALSE.
`adaptive_S`	Logicals. If TRUE, the model adapts the signatures (S) of genomic measures for the test samples. The default is FALSE.

mixture_beta	Logicals. If TRUE, elements of the pathway activation matrix are modeled by a spike-and-slab mixuture distribution. The default is TRUE.
outputDir	The path to the directory to save the output files. The path needs to be quoted in double quotation marks.

Details

The assign.cv.output function is suggested to run after the assign.preprocess, assign.mcmc and assign.summary function. For the cross validation, The Y argument in the assign.mcmc function is the output value "trainingData_sub" from the assign.preprocess function.

Value

The assign.cv.output returns one .csv file containing one/multiple pathway activity for each individual training samples, scatter plots of pathway activity for each individual pathway in all the training samples, and heatmap plots for the gene expression signatures for each individual pathways.

Author(s)

Ying Shen

Examples

```
assign.cv.output(processed.data=processed.data,
mcmc.pos.mean.trainingData=mcmc.pos.mean, trainingData=trainingData1,
trainingLabel=trainingLabel1,
adaptive_B=FALSE, adaptive_S=FALSE, mixture_beta=TRUE, outputDir=tempdir)
```

---

assign.mcmc *The Gibbs sampling algorithm to approximate the joint distribution of the model parameters*

Description

The assign.mcmc function uses a Bayesian sparse factor analysis model to estimate the adaptive baseline/background, adaptive pathway signature, and pathway activation status of individual test (disease) samples.

Usage

```
assign.mcmc(Y, Bg, X, Delta_prior_p, iter=2000, adaptive_B=TRUE, adaptive_S=FALSE,
mixture_beta=TRUE, sigma_sZero = 0.01, sigma_sNonZero = 1, p_beta = 0.01,
sigma_bZero = 0.01, sigma_bNonZero = 1, alpha_tau = 1, beta_tau = 0.01,
Bg_zeroPrior=TRUE, S_zeroPrior=TRUE, ECM = FALSE)
```

Arguments

Y	The G x J matrix of genomic measures (i.e., gene expession) of test samples. Y is the testData_sub variable returned from the data.process function. Genes/probes present in at least one pathway signature are retained.
Bg	The G x 1 vector of genomic measures of the baseline/background (B). Bg is the B_vector variable returned from the data.process function. Bg is the starting value of baseline/background level in the MCMC chain.
X	The G x K matrix of genomic measures of the signature. X is the S_matrix variable returned from the data.process function. X is the starting value of pathway signatures in the MCMC chain.
Delta_prior_p	The G x K matrix of prior probability of a gene being "significant" in its associated pathway. Delta_prior_p is the Pi_matrix variable returned from the data.process function.
adaptive_B	Logicals. If TRUE, the model adapts the baseline/background (B) of genomic measures for the test samples. The default is TRUE.
adaptive_S	Logicals. If TRUE, the model adapts the signatures (S) of genomic measures for the test samples. The default is FALSE.
iter	The number of iterations in the MCMC. The default is 2000.
sigma_sZero	Each element of the signature matrix (S) is modeled by a spike-and-slab mixuture distribution. Sigma_sZero is the variance of the spike normal distribution. The default is 0.01.
sigma_sNonZero	Each element of the signature matrix (S) is modeled by a spike-and-slab mixuture distribution. Sigma_sNonZero is the variance of the slab normal distribution. The default is 1.
p_beta	p_beta is the prior probability of a pathway being activated in individual test samples. The default is 0.01.
sigma_bZero	Each element of the pathway activation matrix (A) is modeled by a spike-and-slab mixuture distribution. sigma_bZero is the variance of the spike normal distribution. The default is 0.01.
sigma_bNonZero	Each element of the pathway activation matrix (A) is modeled by a spike-and-slab mixuture distribution. sigma_bNonZero is the variance of the slab normal distribution. The default is 1.
alpha_tau	The shape parameter of the precision (inverse of the variance) of a gene. The default is 1.
beta_tau	The rate parameter of the precision (inverse of the variance) of a gene. The default is 0.01.
Bg_zeroPrior	Logicals. If TRUE, the prior distritribution of baseline/background level follows a normal distribution with mean zero. The default is TRUE.
mixture_beta	Logicals. If TRUE, elements of the pathway activation matrix are modeled by a spike-and-slab mixuture distribution. The default is TRUE.
S_zeroPrior	Logicals. If TRUE, the prior distritribution of signature follows a normal distribution with mean zero. The default is TRUE.
ECM	Logicals. If TRUE, ECM algorithm, rather than Gibbs sampling, is applied to approximate the model parameters. The default is FALSE.

Details

The assign.mcmc function can be set as following major modes. The combination of logical values of adaptive_B, adaptive_S and mixture_beta can form different modes.

Mode A: adaptive_B = FALSE, adaptive_S = FALSE, mixture_beta = FALSE. This is a regression mode without adaptation of baseline/background, signature, and no shrinkage of the pathway activation level.

Mode B: adaptive_B = TRUE, adaptive_S = FALSE, mixture_beta = FALSE. This is a regression mode with adaptation of baseline/background, but without signature, and with no shrinkage of the pathway activation level.

Mode C: adaptive_B = TRUE, adaptive_S = FALSE, mixture_beta = TRUE. This is a regression mode with adaptation of baseline/background, but without signature, and with shrinkage of the pathway activation level when it is not significantly activated.

Mode D: adaptive_B = TRUE, adaptive_S = TRUE, mixture_beta = TRUE. This is a Bayesian factor analysis mode with adaptation of baseline/background, adaptation signature, and with shrinkage of the pathway activation level.

Value

`beta_mcmc`	The iter x K x J array of the pathway activation level estimated in every iteration of MCMC.
`tau2_mcmc`	The iter x G matrix of the precision of genes estimated in every iteration of MCMC
`gamma_mcmc`	The iter x K x J array of probability of pathway being activated estimated in every iteration of MCMC.
`kappa_mcmc`	The iter x K x J array of pathway activation level (adjusted beta scaling between 0 and 1) estimated in every iteration of MCMC.)
`S_mcmc`	The iter x G x K array of signature estimated in every iteration of MCMC.
`Delta_mcmc`	The iter x G x K array of binary indicator of a gene being significant estimated in every iteration of MCMC.

Author(s)

Ying Shen

Examples

```
mcmc.chain <- assign.mcmc(Y=processed.data$testData_sub, Bg = processed.data$B_vector,
X=processed.data$S_matrix, Delta_prior_p = processed.data$Pi_matrix, iter = 20,
adaptive_B=TRUE, adaptive_S=FALSE, mixture_beta=TRUE)
```

assign.output *Prediction/validation output for test data*

Description

The assign.output function outputs the summary results and plots for prediction/validation for the test dataset.

Usage

```
assign.output(processed.data, mcmc.pos.mean.testData, trainingData, testData,
trainingLabel, testLabel, geneList, adaptive_B=TRUE, adaptive_S=FALSE, mixture_beta=TRUE,
outputDir)
```

Arguments

processed.data	The list object returned from the assign.preprocess function.
mcmc.pos.mean.testData	The list object returned from the assign.mcmc function. Notice that for prediction/validation in the test dataset, the Y argument in the assign.mcmc function should be set as the test dataset.
trainingData	The genomic measure matrix of training samples (i.g., gene expression matrix). The dimension of this matrix is probe number x sample number.
testData	The genomic measure matrix of test samples (i.g., gene expression matrix). The dimension of this matrix is probe number x sample number.
trainingLabel	The list linking the index of each training sample to a specific group it belongs to.
testLabel	The vector of the phenotypes/labels of the test samples.
geneList	The list that collects the signature genes of one/multiple pathways. Every component of this list contains the signature genes associated with one pathway.
adaptive_B	Logicals. If TRUE, the model adapts the baseline/background (B) of genomic measures for the test samples. The default is TRUE.
adaptive_S	Logicals. If TRUE, the model adapts the signatures (S) of genomic measures for the test samples. The default is FALSE.
mixture_beta	Logicals. If TRUE, elements of the pathway activation matrix are modeled by a spike-and-slab mixuture distribution. The default is TRUE.
outputDir	The path to the directory to save the output files. The path needs to be quoted in double quotation marks.

Details

The assign.output function is suggested to run after the assign.preprocess, assign.mcmc and assign.summary functions. For the prediction/validation in the test dataset, The Y argument in the assign.mcmc function is the output value "testData_sub" from the assign.preprocess function.

Value

The assign.output returns one .csv file containing one/multiple pathway activity for each individual test samples, scatter plots of pathway activity for each individual pathway in all the test samples, and heatmap plots for the gene expression of the prior signature and posterior signtures (if adaptive_S equals TRUE) of each individual pathway in the test samples.

Author(s)

Ying Shen

Examples

```
assign.output(processed.data=processed.data,
mcmc.pos.mean.testData=mcmc.pos.mean, trainingData=trainingData1,
testData=testData1, trainingLabel=trainingLabel1, testLabel=testLabel1,
geneList=NULL, adaptive_B=TRUE, adaptive_S=FALSE, mixture_beta=TRUE, outputDir=tempdir)
```

---

assign.preprocess *Input data preprocessing*

Description

The assign.preprocess function is used to perform quality control on the user-provided input data and generate starting values and/or prior values for the model parameters. The assign.preprocess function is optional. For users who already have the correct format for the input of the assign function, they can skip this step and go directly to the assign.mcmc function.

Usage

```
assign.preprocess(trainingData=NULL, testData, trainingLabel, geneList=NULL,
n_sigGene=NA, theta0=0.05, theta1=0.9)
```

Arguments

`trainingData`	The genomic measure matrix of training samples (i.g., gene expression matrix). The dimension of this matrix is probe number x sample number. The default is NULL.
`testData`	The genomic measure matrix of test samples (i.g., gene expression matrix). The dimension of this matrix is probe number x sample number.
`trainingLabel`	The list linking the index of each training sample to a specific group it belongs to. See details and examples for more information.
`geneList`	The list that collects the signature genes of one/multiple pathways. Every component of this list contains the signature genes associated with one pathway. The default is NULL.

`n_sigGene` The vector of the signature genes to be identified for one pathway. n_sigGene needs to be specified when geneList is set NULL. The default is NA. See examples for more information.

`theta0` The prior probability for a gene to be significant, given that the gene is NOT defined as "significant" in the signature gene lists provided by the user. The default is 0.05.

`theta1` The prior probability for a gene to be significant, given that the gene is defined as "significant" in the signature gene lists provided by the user. The default is 0.9.

Details

The assign.preprocess is applied to perform quality control on the user-provided genomic data and meta data, re-format the data in a way that can be used in the following analysis, and generate starting/prior values for the pathway signature matrix. The output values of the assign.preprocess function will be used as input values for the assign.mcmc function.

For training data with 1 control group and 3 experimental groups (10 samples/group; all 3 experimental groups share 1 control group), the trainingLabel can be specified as: trainingLabel <- list(control = list(expr1=1:10, expr2=1:10, expr3=1:10), expr1 = 11:20, expr2 = 21:30, expr3 = 31:40)

For training data with 3 control groups and 3 experimental groups (10 samples/group; Each experimental group has its corresponding control group), the trainingLabel can be specified as: trainingLabel <- list(control = list(expr1=1:10, expr2=21:30, expr3=41:50), expr1 = 11:20, expr2 = 31:40, expr3 = 51:60)

It is highly recommended that the user use the same expriment name when specifying control indice and exprimental indice.

Value

`trainingData_sub`
The G x N matrix of G genomic measures (i.e., gene expession) of N training samples. Genes/probes present in at least one pathway signature are retained. Only returned when the training dataset is available.

`testData_sub` The G x N matrix of G genomic measures (i.e., gene expession) of N test samples. Genes/probes present in at least one pathway signature are retained.

`B_vector` The G x 1 vector of genomic measures of the baseline/background. Each element of the B_vector is calculated as the mean of the genomic measures of the control samples in training data.

`S_matrix` The G x K matrix of genomic measures of the signature. Each column of the S_matrix represents a pathway. Each element of the S_matrix is calculated as the mean of genomic measures of the experimental samples minus the mean of the control samples in the training data.

`Delta_matrix` The G x K matrix of binary indicators. Each column of the Delta_matrix represents a pathway. The elements in Delta_matrix are binary (0, insignificant gene; 1, significant gene).

Pi_matrix	The G x K matrix of probability p of a Bernoulli distribution. Each column of the Pi_matrix represents a pathway. Each element in the Pi_matrix is the probability of a gene to be significant in its associated pathway.
diffGeneList	The list that collects the signature genes of one/multiple pathways generated from the training samples or from the user provided gene list. Every component of this list contains the signature genes associated with one pathway.

Author(s)

Ying Shen

Examples

```
processed.data <- assign.preprocess(trainingData=trainingData1,
testData=testData1, trainingLabel=trainingLabel1, geneList=geneList1)
```

---

assign.summary *Summary of the model parameters estimated by the Gibbs sampling algorithm*

Description

The assign.summary function computes the posterior mean of the model parameters estimated in every iteration during the Gibbs sampling.

Usage

```
assign.summary(test, burn_in=1000, iter=2000, adaptive_B = TRUE, adaptive_S = FALSE,
mixture_beta = TRUE)
```

Arguments

test	The list object returned from the assign.mcmc function. The list components are the MCMC chains of the B, S, Delta, beta, gamma, and sigma.
burn_in	The number of burn-in iterations. These iterations are discarded when computing the posterior means of the model parameters. The default is 1000.
iter	The number of total iterations. The default is 2000.
adaptive_B	Logicals. If TRUE, the model adapts the baseline/background (B) of genomic measures for the test samples. The default is TRUE.
adaptive_S	Logicals. If TRUE, the model adapts the signatures (S) of genomic measures for the test samples. The default is FALSE.
mixture_beta	Logicals. If TRUE, elements of the pathway activation matrix are modeled by a spike-and-slab mixuture distribution. The default is TRUE.

Details

The assign.summary function is suggested to run after the assign.convergence function, which is used to check the convergency of the MCMC chain. If the MCMC chain does not converge to a stationary phase, more iterations are required in the assign.mcmc function. The number of burn-in iterations is usually set to be half of the number of total iterations, meaning that the first half of the MCMC chain is discarded when computing the posterior means.

Value

`beta_pos`	The N x K matrix of the posterior mean of the pathway activation level in test samples (transposed matrix A). Columns:K pathways; rows: N test samples
`sigma_pos`	The G x 1 vector of the posterior mean of the variance of gene.
`kappa_pos`	The N x K matrix of posterior mean of pathway activation level in test samples (transposed matrix A) (adjusted beta_pos scaling between 0 and 1). Columns:K pathways; rows: N test samples
`gamma_pos`	The N x K matrix of the posterior probability of pathways being activated in test samples.
`S_pos`	The G x K matrix of the posterior mean of pathway signature genes.
`Delta_pos`	The G x K matrix of the posterior probability of genes being significant in the associated pathways.

Author(s)

Ying Shen

Examples

`assign.wrapper` *ASSIGN All-in-one function*

Description

The assign.wrapper function integrates the assign.preprocess, assign.mcmc, assign.summary, assign.output, assign.cv.output functions into one wrapper function.

Usage

```
assign.wrapper(trainingData=NULL, testData, trainingLabel, testLabel=NULL,
geneList=NULL, n_sigGene=NA, adaptive_B=TRUE, adaptive_S=FALSE, mixture_beta=TRUE,
outputDir, p_beta=0.01, theta0=0.05, theta1=0.9, iter=2000, burn_in=1000)
```

Arguments

trainingData	The genomic measure matrix of training samples (i.g., gene expression matrix). The dimension of this matrix is probe number x sample number. The default is NULL.
testData	The genomic measure matrix of test samples (i.g., gene expression matrix). The dimension of this matrix is probe number x sample number.
trainingLabel	The list linking the index of each training sample to a specific group it belongs to. See examples for more information.
testLabel	The vector of the phenotypes/labels of the test samples. The default is NULL.
geneList	The list that collects the signature genes of one/multiple pathways. Every component of this list contains the signature genes associated with one pathway. The default is NULL.
n_sigGene	The vector of the signature genes to be identified for one pathway. n_sigGene needs to be specified when geneList is set NULL. The default is NA. See examples for more information.
adaptive_B	Logicals. If TRUE, the model adapts the baseline/background (B) of genomic measures for the test samples. The default is TRUE.
adaptive_S	Logicals. If TRUE, the model adapts the signatures (S) of genomic measures for the test samples. The default is FALSE.
mixture_beta	Logicals. If TRUE, elements of the pathway activation matrix are modeled by a spike-and-slab mixuture distribution. The default is TRUE.
outputDir	The path to the directory to save the output files. The path needs to be quoted in double quotation marks.
p_beta	p_beta is the prior probability of a pathway being activated in individual test samples. The default is 0.01.
theta0	The prior probability for a gene to be significant, given that the gene is NOT defined as "significant" in the signature gene lists provided by the user. The default is 0.05.
theta1	The prior probability for a gene to be significant, given that the gene is defined as "significant" in the signature gene lists provided by the user. The default is 0.9.
iter	The number of iterations in the MCMC. The default is 2000.
burn_in	The number of burn-in iterations. These iterations are discarded when computing the posterior means of the model parameters. The default is 1000.

Details

The assign.wrapper function is an all-in-one function which output the necessary results for the basic users. For the users who need more intermetiate results for model diagnosis, it is better to run the assign.preprocess, assign.mcmc, assign.convergence, assign.summary functions by order and extract the output values from the returned list objects of those functions.

Value

The assign.wrapper returns one/multiple pathway activity for each individual training samples and test samples, scatter plots of pathway activity for each individual pathway in the training and test samples, heatmap plots for gene expression signatures for each individual pathways, heatmap plots for the gene expression of the prior signature and posterior signtures (if adaptive_S equals TRUE) of each individual pathway in the test samples.

Author(s)

Ying Shen and W. Evan Johnson

Examples

```
data(trainingData1)
data(testData1)
data(geneList1)

trainingLabel1 <- list(control = list(bcat=1:10, e2f3=1:10, myc=1:10, ras=1:10,
src=1:10), bcat = 11:19, e2f3 = 20:28, myc= 29:38, ras = 39:48, src = 49:55)
testLabel1 <- rep(c("subtypeA","subtypeB"),c(53,58))

assign.wrapper(trainingData=trainingData1, testData=testData1,
trainingLabel=trainingLabel1, testLabel=testLabel1, geneList=geneList1,
adaptive_B=TRUE, adaptive_S=FALSE, mixture_beta=TRUE,
outputDir=tempdir, p_beta=0.01, theta0=0.05, theta1=0.9,
iter=20, burn_in=10)
```

---

geneList1 *Pathway signature gene sets*

Description

Signature genes for 5 oncogenic pathways.

Usage

```
data(geneList1)
```

Format

List with 5 components representing each pathway. 200 signature genes are selected for each pathway.

Source

Bild et al. (2006) Oncogenic pathway signatures in human cancers as a guide to targeted therapies. Nature, 439, 353-357.

testData1 *Gene expression profiling from cancer patients (test dataset)*

Description

Gene expression datasets for 111 cancer patient samples.

Usage

```
data(testData1)
```

Format

Data frame with 1000 genes/probes (rows) and 111 samples (columns)

Source

Bild et al. (2006) Oncogenic pathway signatures in human cancers as a guide to targeted therapies. Nature, 439, 353-357.

trainingData1 *Gene expression profiling from cell line preturbation experiments (training dataset)*

Description

Gene expression datasets for 5 pathway preturbation experiments.

Usage

```
data(trainingData1)
```

Format

Data frame with 1000 genes/probes (rows) and 55 samples (columns)

Source

Bild et al. (2006) Oncogenic pathway signatures in human cancers as a guide to targeted therapies. Nature, 439, 353-357.

Index