

# Package ‘mAPKL’

March 30, 2021

**Type** Package

**Title** A Hybrid Feature Selection method for gene expression data

**Version** 1.20.0

**Date** 2019-05-07

**Author** Argiris Sakellariou

**Maintainer** Argiris Sakellariou <argisake@gmail.com>

**Depends** R (>= 3.6.0), Biobase

**Imports** multtest, clusterSim, apcluster, limma, e1071, AnnotationDbi,  
methods, parmigene, igraph, reactome.db

**Suggests** BiocStyle, knitr, mAPKLDData, hgu133plus2.db, RUnit,  
BiocGenerics

**VignetteBuilder** knitr

**Description** We propose a hybrid FS method (mAP-KL), which combines multiple hypothesis testing and affinity propagation (AP)-clustering algorithm along with the Krzanowski & Lai cluster quality index, to select a small yet informative subset of genes.

**License** GPL (>= 2)

**biocViews** FeatureExtraction, DifferentialExpression, Microarray,  
GeneExpression

**NeedsCompilation** no

**git\_url** <https://git.bioconductor.org/packages/mAPKL>

**git\_branch** RELEASE\_3\_12

**git\_last\_commit** 68b77e0

**git\_last\_commit\_date** 2020-10-27

**Date/Publication** 2021-03-29

## R topics documented:

mAPKL-package . . . . .	2
Annot-class . . . . .	2
annotate . . . . .	3
classification . . . . .	4
Classify-class . . . . .	5
DataLD-class . . . . .	6
loadFiles . . . . .	6

mAPKL . . . . .	7
mAPKLRes-class . . . . .	8
metrics . . . . .	9
NetAttr-class . . . . .	10
netwAttr . . . . .	10
preprocess . . . . .	11
probes2pathways . . . . .	12
report . . . . .	13
sampling . . . . .	14

<b>Index</b>	<b>16</b>
--------------	-----------

---

mAPKL-package	<i>A hybrid feature selection method for gene expression data</i>
---------------	---

---

### Description

A hybrid FS method (mAP-KL), which combines multiple hypothesis testing and affinity propagation (AP) clustering algorithm along with the Krzanowski & Lai cluster quality index, to select an informative subset of genes.

### Details

Package: mAPKL  
 Type: Package  
 Version: 0.99.01  
 Date: 2014-05-07  
 License: GPL (>= 2)

### Author(s)

Argiris Sakellariou Argiris Sakellariou <a.sakellariou@gonkhosp.gr>

### References

A. Sakellariou, D. Sanoudou, and G. Spyrou, "Combining multiple hypothesis testing and affinity propagation clustering leads to accurate, robust and sample size independent classification on gene expression data, " BMC Bioinformatics, vol. 13, p. 270, 2012.

---

Annot-class	<i>Class "Annot"</i>
-------------	----------------------

---

### Description

S4 class for storing Annot analysis results

**Slots**

The following slots are defined for [Annot](#) objects:

- results: The accumulated annotation results
- probe: The probe id
- symbol: The official gene symbol
- entrezId: The Entrez gene Identifier
- ensemblId: The ensembl ID as indicated by ensembl
- map: The cytoband locations of the gene

**Author(s)**

Argiris Sakellariou <a.sakellariou@gonkhosp.gr>

---

annotate	<i>Genome annotation of the "exemplars".</i>
----------	--

---

**Description**

The user may extract several genome specific information for the "exemplars" as described in the microarray annotation file.

**Usage**

```
annotate(exemplars, chip)
```

**Arguments**

- exemplars      The "exemplars" of the mAPKL class.
- chip            The platform's name of the microarray chip used (e.g. "hgu133plus2.db")

**Details**

This function uses as key the probe id and returns the matching information as described in the gene chip annotation file. The returned information are usually multiple to the number of probe ids (one to many relationship) and are stored in the results. In the rest items only the first of the returned values are stored.

**Value**

- results      The accumulated annotation results.
- probe        The probe id.
- symbol      The official gene symbol.
- entrezId    The Entrez gene Identifier.
- ensemblId   The ensembl ID as indicated by ensembl.
- map         The cytoband locations of the gene.

**Author(s)**

Argiris Sakellariou

**Examples**

```
## We use the "exemplars" from the mAPKL.Rd example

exemplrs <- c(24, 26, 42, 45, 63, 81, 95, 99, 102, 113, 134, 135, 145, 152, 168)
names(exemplrs) <- c("215717_s_at", "1561358_at", "222752_s_at", "233922_at",
"218871_x_at", "33323_r_at", "244311_at", "220932_at", "205508_at", "209596_at",
"215180_at", "1560638_a_at", "201852_x_at", "229947_at", "221731_x_at")

gene.info <- annotate(exemplrs, "hgu133plus2.db")
```

classification

*Classify samples according to the SVM algorithm***Description**

This function performs classification through the Support Vector Machines (SVM) algorithm. The algorithm applies on the "exemplars" dataset. It produces a classification result either on the training set or on a validation set. This function estimates how well the selected "genes" from mAP-KL method discriminate between two phenotypes. The default SVM settings are: "linear" kernel and 5-folds cross-validation. Regarding the parameters for the "linear" kernel, cost parameter, and for the "radial" kernel, cost and gamma parameters, are estimated automatically through the tune.svm function as described in e1071 r-package.

**Usage**

```
classification(trExemplObj, classLabels, valExemplObj=NULL, kf=5, kernel="linear")
```

**Arguments**

trExemplObj	The exemplars train eSet object.
classLabels	The varLabels name in the eSet object where the class labels are stored e.g "type".
valExemplObj	The exemplars validation eSet object (if not NULL).
kf	The k-folds value of the cross-validation parameter. The default value is 5-folds. By setting "Loo" or "LOO" a Leave-One-Out Cross Validation is performed
kernel	The type of kernel used for the classification analysis. The default kernel is "linear"

**Value**

classL	The labels of the train set
valClassL	The labels of the validation set if not NULL
predLbls	The predicted labels according to the classification analysis

**Author(s)**

Argiris Sakellariou

**Examples**

```

library(mAPKLData)
data(mAPKLData)
breast <- sampling(Data=mAPKLData, valPercent=40, classLabels="type", seed=135)
normTrainData <- preprocess(breast$trainData)
normTestData <- preprocess(breast$testData)

exprs(breast$trainData)<-normTrainData$c1L2.normdata
exprs(breast$testData)<-normTestData$c1L2.normdata

out.c1L2 <- mAPKL(trObj=breast$trainData, classLabels="type",
valObj=breast$testData,dataType=7)

clasPred <- classification(trExemplObj=out.c1L2@exemplTrain, classLabels="type",
valExemplObj=out.c1L2@exemplTest)

```

---

Classify-class

*Class "Classify"*


---

**Description**

S4 class for storing Classify analysis results

**Slots**

The following slots are defined for [Classify](#) objects:

**classL:** Number of samples in the data set

**valClassL:** Subset of samples used for leveraged clustering

**predLb1s:** Vector containing indices of exemplars

**AUC:** The Area Under the ROC curve as a degree of samples discrimination

**Accuracy:** The classification accuracy

**MCC:** The MCC classification measure

**Specificity:** The degree of true negative's identification

**Sensitivity:** The degree of true positive's identification

**Author(s)**

Argiris Sakellariou <a.sakellariou@gonkhosp.gr>

---

DataLD-class	<i>Class "DataLD"</i>
--------------	-----------------------

---

**Description**

S4 class for storing DataLD analysis results

**Slots**

The following slots are defined for [DataLD](#) objects:

`trainObj`: An eSet class object of a training set

`valObj`: An eSet class object of a validation set

**Author(s)**

Argiris Sakellariou <a.sakellariou@gonkhosp.gr>

---

loadFiles	<i>Imports gene expression data</i>
-----------	-------------------------------------

---

**Description**

This function loads the train set, the class labels files as well as the test or validation file if any. Then we may perform normalization and (or) log2 transformation.

**Usage**

```
loadFiles(filesPath, trainFile, labelsFile, validationFile=NULL,
validationLabels=NULL)
```

**Arguments**

<code>filesPath</code>	The path where the files are stored
<code>trainFile</code>	The genes and the relevant intensity values for feature selection analysis. The file should be of tab-delimited format
<code>labelsFile</code>	The class labels of the samples
<code>validationFile</code>	A further file with genes and intensity values used for validation purposes
<code>validationLabels</code>	The class labels of the validation samples

**Value**

An object of Class [DataLD](#)

**Author(s)**

Argiris Sakellariou

**Description**

We first employ the `mt.maxT` function from the "multtest" package to rank the genes of the training set and then we reserve the top N genes e.g.(N = 200) for further exploitation. Prior to clustering analysis with Affinity Propagation we apply the index of Krzanowski and Lai as included in the "ClusterSim" package to determine the number of clusters solely on the disease samples of the training test set. The final step involves the cluster analysis with the AP clustering method as in the "apcluster" package, which detects n (n = k, the Krzanowski and Lai index) clusters among the top N genes and provides us with a list of the most representative genes of each cluster, the so called exemplars.

**Usage**

```
mAPKL(trObj, classLabels, valObj=NULL, dataType=6, statTest="t", permutations=1000,
features=200, minClusters=2, maxClusters=50, FC="limma", bimaxit=50, r=2)
```

**Arguments**

<code>trObj</code>	The train eSet object.
<code>classLabels</code>	The varLabels name in the eSet object where the class labels are stored e.g "type".
<code>valObj</code>	The validation eSet object (if not NULL).
<code>dataType</code>	The type of the data e.g 6-ratio data without normalization and 7-interval or mixed (ratio & interval) data without normalization as described in "clusterSim" package.
<code>statTest</code>	The statistical test applied to the geneIntensities. The available tests described in <code>mt.maxT</code> documentation in "multtest" package.
<code>permutations</code>	The number of permutations.
<code>features</code>	The top N genes to be kept.
<code>minClusters</code>	The minimum number of clusters that can be identified.
<code>maxClusters</code>	The maximum number of clusters that can be identified.
<code>FC</code>	The Fold Change of the exemplars according to "Limma" (default). Alternatively the "SAM" approach may be computed.
<code>bimaxit</code>	The maximum number of bisection steps performed by the AP algorithm. The (default) value is "50".
<code>r</code>	The argument r is used to transform the resulting distances by computing the r-th power. To obtain negative squared distances as in Frey's and Dueck's (use <code>r=2</code> as default).

**Value**

<code>rankedIntens</code>	The top N ranked genes with their intensity values
<code>exemplTrain</code>	The intensity values of the exemplars in the training set
<code>exemplTest</code>	The intensity values of the exemplars in the validation set if not NULL

statistic	A list with the overall results of the "mt.maxT" analysis
adjp	The adjusted p-values according to the statistical analysis
pVal	The raw p-values according to the statistical analysis
fc	The Fold Change of the exemplars
exemplars	The selected "significant" probe ids/genes
clusters	The probe ids/genes per cluster

**Author(s)**

Argiris Sakellariou

**References**

A. Sakellariou, D. Sanoudou, and G. Spyrou, "Combining multiple hypothesis testing and affinity propagation clustering leads to accurate, robust and sample size independent classification on gene expression data," BMC Bioinformatics, vol. 13, p. 270, 2012.

**Examples**

```
## Using separate train-test samples
## Load the necessary files based on Breast cancer data as included in the
## package mAPKLData

library(mAPKLData)
data(mAPKLData)
breast <- sampling(Data=mAPKLData, valPercent=40, classLabels="type", seed=135)
normTrainData <- preprocess(breast$trainData)
normTestData <- preprocess(breast$testData)

exprs(breast$trainData) <- normTrainData$cLL2.normdata
exprs(breast$testData) <- normTestData$cLL2.normdata

out.cLL2 <- mAPKL(trObj=breast$trainData, classLabels="type",
valObj=breast$testData, dataType=7)
```

---

mAPKLRes-class

*Class "mAPKLRes"*


---

**Description**

S4 class for storing mAPKL analysis results

**Slots**

The following slots are defined for [mAPKLRes](#) objects:

**rankedIntens:** The top N ranked genes along with their intensity values

**exemplTrain:** An eSet class object formed with the exemplars of the training set

**exemplTest:** An eSet class object formed with the exemplars of the validation set if not NULL



**statistic:** A list with the overall results of the "mt.maxT" analysis  
**adjp:** The adjusted p-values according to the statistical analysis  
**pVal:** The raw p-values according to the statistical analysis  
**fc:** The Fold Change of the exemplars  
**exemplars:** The selected "significant" probe ids/genes  
**clusters:** The probe ids/genes per cluster

### Author(s)

Argiris Sakellariou <a.sakellariou@gonkhosp.gr>

---

metrics

*Computes several clasification metrics*

---

### Description

This function calculates several classification related metrics. It uses the original and the predicted samples' labels to quantify the quality of the classification process. Those meassures give us a direct outlook of the selected "genes" and how well discriminate between two phenotypes.

### Usage

```
metrics(classLbIs, predLbIs)
```

### Arguments

classLbIs	The initial class labels.
predLbIs	The predicted class labels.

### Value

AUC	The Area Under the ROC curve as a degree of samples discrimination
Accuracy	The classification accuracy
MCC	The MCC classification meassure
Specificity	The degree of true negative's identification
Sensitivity	The degree of true positive's identification

### Author(s)

Argiris Sakellariou

### Examples

```

## Suppose 'val' represent the correct validation set labels
## and 'predictions' the predicted labels according to an SVM model

val <- c(rep(0,8),rep(1,4))
predictions <- c(rep(0,6),1,1,rep(1,3),0)
perfMetrics <- metrics(classLbIs=val, predLbIs=predictions)

```

---

NetAttr-class	<i>Class "NetAttr"</i>
---------------	------------------------

---

### Description

S4 class for storing some network characteristics of the top ranked genes

### Slots

The following slots are defined for [NetAttr](#) objects:

**edgelist:** The Node1 & Node2 & Weight edgelist matrix

**degree:** The Local, Global and their Weighted values of the "degree" characteristic

**closeness:** The Weighted values of the Local and Global "closeness" characteristic

**betweenness:** The Weighted values of the Local and Global "betweenness" characteristic

**transitivity:** The Weighted values of the Local and Global "transitivity" characteristic

### Author(s)

Argiris Sakellariou <a.sakellariou@gonkhosp.gr>

---

netwAttr	<i>Calculates network characteristics</i>
----------	---

---

### Description

Calculate some basic network characteristics of the top ranked genes

### Usage

```
netwAttr(mAPKLObj, net="clr")
```

### Arguments

mAPKLObj	An object of mAPKL class.
net	The network reconstruction method to be employed. The user may select between "clr" (default), "aracne.a" and "aracne.m".

### Details

It calculates some basic network characteristics. Those include the "degree", the "closeness", the "betweenness", and finally the "transitivity" or else clustering coefficient. We calculate the weighted values for both local and global scores.

The three available network reconstruction options are:

clr: Context Likelihood or Relatedness Network

aracne.a: Algorithm for the Reconstruction of Accurate Cellular Networks (additive model)

aracne.m: Algorithm for the Reconstruction of Accurate Cellular Networks (multiplicative model)

**Value**

Upon successful completion, the function returns an `NetAttr` object.

**Author(s)**

Argiris Sakellariou

**Examples**

```
library(mAPKLData)
data(mAPKLData)
breast <- sampling(Data=mAPKLData, valPercent=40, classLabels="type", seed=135)
normTrainData <- preprocess(breast$trainData)
normTestData <- preprocess(breast$testData)

exprs(breast$trainData) <- normTrainData$c1L2.normdata
exprs(breast$testData) <- normTestData$c1L2.normdata

out.c1L2 <- mAPKL(trObj=breast$trainData, classLabels="type",
valObj=breast$testData, dataType=7)

net.attr <- netwAttr(mAPKLObj=out.c1L2)
```

---

```
preprocess
```

*Performs normalization and/or log2 transformation*

---

**Description**

This function performs normalization and/or log2 transformation on gene expression data.

**Usage**

```
preprocess(exprsObj, log2=TRUE, norm="ALL", destname=NULL)
```

**Arguments**

<code>exprsObj</code>	An <code>eSet</code> object where its assay data will be normalized
<code>log2</code>	Performs logarithmic transformation of base 2 prior to any normalization. The default value is <code>TRUE</code>
<code>norm</code>	The user may define a specific normalization method rather than "ALL" which is the default case. The available abbreviations are described in the details section
<code>destname</code>	Here we define the destination path and the name of the jpeg file with the density plots. The default path is the working directory

**Details**

The available normalization methods are:

Mean-centering normalization "mc"

z-score normalization "z"

Quantile normalization "q"

Cyclic loess normalization "cl"

Mean-centering normalization and log2 transformation "mcL2"

z-score normalization and log2 transformation "zL2"

Quantile normalization and log2 transformation "qL2"

Cyclic loess normalization and log2 transformation "clL2"

**Value**

rawdata	The initial gene expression values
mc.normdata	The values after 'mean-centering' normalization
z.normdata	The values after 'z-score' normalization
q.normdata	The values after 'quantile' normalization
cl.normdata	The values after 'cyclic loess' normalization
mcL2.normdata	The values after 'mean-centering' normalization and log2
zL2.normdata	The values after 'z-score' normalization and log2
qL2.normdata	The values after 'quantile' normalization and log2
clL2.normdata	The values after 'cyclic loess' normalization and log2

**Author(s)**

Argiris Sakellariou

**Examples**

```
library(mAPKLData)
data(mAPKLData)
varLabels(mAPKLData)
breast <- sampling(Data=mAPKLData, valPercent=40, classLabels="type", seed=135)
normTrainData <- preprocess(exprsObj=breast$trainData)
```

---

probes2pathways

*Extract pathways from "exemplars"*

---

**Description**

The user may extract the pathways where the "exemplars" are involved with the aid of the reactome database.

**Usage**

```
probes2pathways(annotObj)
```

**Arguments**

annotObj            The "Annot" class object.

**Details**

This function utilizes the "Annot" class object as returned by the "annotate" function to extract the pathways where the "exemplars" are involved with the aid of the "reactome" database. We employ the probe Ids for the matching.

**Author(s)**

Argiris Sakellariou

**Examples**

```
## We use the "exemplars" from the mAPKL.Rd example

exemplrs <- c(24, 26, 42, 45, 63, 81, 95, 99, 102, 113, 134, 135, 145, 152, 168)
names(exemplrs) <- c("215717_s_at", "1561358_at", "222752_s_at", "233922_at",
"218871_x_at", "33323_r_at", "244311_at", "220932_at", "205508_at", "209596_at",
"215180_at", "1560638_a_at", "201852_x_at", "229947_at", "221731_x_at")

gene.info <- annotate(exemplrs, "hgu133plus2.db")

## We now use the "gene.info" to identify the relevant pathways

probes2pathways(gene.info)
```

---

report

---

*Produce an HTML report of the mAP-KL analysis*


---

**Description**

This function gathers the results of several analysis sessions ( feature selection, classification, annotation and network) and produces a report in HTML format.

**Usage**

```
report(mAPKLObj, ClassifyObj, AnnotObj=NULL, netObj=NULL, file)
```

**Arguments**

mAPKLObj            An object of mAPKL class.  
ClassifyObj         An object of mAPKL class.  
AnnotObj            An object of Annot class.  
netObj              An object of NetAttr class.  
file                 The full path and the name of the produced report

**Details**

It presents the data samples with their phenotype labels, the exemplars with their statistical scores (adj.p-value, p-value and fc), and network characteristics (like weighted local degree, closeness, betweenness, transitivity) if such a network analysis has been performed. In addition, the report presents the classification performance achieved by those exemplars (either cross-validation or hold-out), and finally several annotation information (symbol, entrez id, ensemble id, and their chromosomal location) if an annotation analysis has been carried out.

**Value**

Upon successful completion an HTML report is saved in the working directory.

**Author(s)**

Argiris Sakellariou

**Examples**

```
## When a network attributes object is present
## Not run: report(out,class.pred,class.metrics,netObj=net.attr,
"C:/.../.../mAPKLreport.html")#Define a local path to store your report
## End(Not run)

## When an annotation object is present
## Not run: report(out,class.pred,class.metrics,gene.info,
"C:/.../.../mAPKLreport.html")#Define a local path to store your report
## End(Not run)

## Without annotation and network attributes objects
## Not run: report(out,class.pred,class.metrics,
file="C:/.../.../mAPKLreport.html")#Define a local path to store your report
## End(Not run)
```

---

sampling

*Splits a dataset to a train and a test sets of a user defined percentage*

---

**Description**

This function takes as input a dataset and splits it into a train and a test set based to a user defined percentage.

**Usage**

```
sampling(Data, valPercent, classLabels, seed)
```

**Arguments**

Data	The input dataset to be split as an eSet object.
valPercent	The percentage of the input dataset used for validation purposes e.g. 40.
classLabels	The varLabels name in the eSet object where the class labels are stored e.g "type".
seed	Setting the seed number for reproducible sampling. The default value is 1.

**Value**

trainData	The data used for training as an eSet object
testData	The data used for validation as an eSet object

**Author(s)**

Argiris Sakellariou

**Examples**

```
library(mAPKLData)
data(mAPKLData)
breast <- sampling(Data=mAPKLData, valPercent=40, classLabels="type", seed=135)
```

# Index

- \* **IO**
    - loadFiles, 6
  - \* **classes**
    - Annot-class, 2
    - Classify-class, 5
    - DataLD-class, 6
    - mAPKLRes-class, 8
    - NetAttr-class, 10
  - \* **classif**
    - classification, 4
  - \* **datagen**
    - sampling, 14
  - \* **htest**
    - mAPKL, 7
    - mAPKL-package, 2
  - \* **methods**
    - annotate, 3
    - metrics, 9
    - netwAttr, 10
    - preprocess, 11
    - probes2pathways, 12
    - report, 13
- Annot, 3
- Annot (Annot-class), 2
- Annot-class, 2
- annotate, 3
- classification, 4
- Classify, 5
- Classify (Classify-class), 5
- Classify-class, 5
- DataLD, 6
- DataLD (DataLD-class), 6
- DataLD-class, 6
- loadFiles, 6
- mAPKL, 7
- mAPKL-package, 2
- mAPKLRes, 8
- mAPKLRes (mAPKLRes-class), 8
- mAPKLRes-class, 8
- metrics, 9
- NetAttr, 10, 11
- NetAttr (NetAttr-class), 10
- NetAttr-class, 10
- netwAttr, 10
- preprocess, 11
- probes2pathways, 12
- report, 13
- sampling, 14