# Package 'PhysioSpaceMethods'

September 25, 2020

**Type** Package

**Title** Creates and uses physio spaces as a dimension reduction mapping

**Version** 0.99.66

**Description** This package uses 'Big Data' to make robust 'Physiological Vectors' in N dimension space, within which it is possible to map new unknown data. The calculation pipeline is a robust statistical method for relating high dimensional omics data sets from heterogeneous sources using shared physiological processes. It is designed to take advantage of the vast availability of public omics data, which in combination with statistical approaches makes a potent tool capable of analyzing heterogeneous biological data sets. 'calculatePhysioMap' is the main analytical function of the package. It uses a nonlinear mapping function to relate the unknown input data with a physiological space. Physiological spaces are mathematical spaces built upon known physiological data, and generated using the 'spaceMaker' function.

**biocViews** DimensionReduction, Clustering, GeneExpression, Software

**License** GPL-3

**Encoding** UTF-8

**LazyData** FALSE

**Imports** progress,
parallel,
missMDA,
DMwR,
grDevices,
graphics,
stats,
utils,
DESeq2,
limma,
SummarizedExperiment,
BiocParallel

**BugReports** https://github.com/JRC-COMBINE/PhysioSpaceMethods/issues

**URL** https://github.com/JRC-COMBINE/PhysioSpaceMethods

**RoxygenNote** 7.1.0

**Suggests** knitr,
BiocStyle,
rmarkdown,
testthat,
ExperimentHub,
biomaRt,
EnrichmentBrowser,
org.Hs.eg.db

**VignetteBuilder** knitr

# R topics documented:



---

`.imputeMissingGeneExpression` *Imputing Missing Data*

---

## Description

.imputeMissingGeneExpression is an internal function called by inputChecker. It uses different methods to impute any missing value of the InputData or Space (or else missing values would break the pipeline in calculatePhysioMap).

## Usage

```
.imputeMissingGeneExpression(InptGEX, METHOD = "PCA")
```

## Arguments

| | |
|---|---|
| `InptGEX` | Input matrix with missng values. |
| `METHOD` | Method to use in imputation. Available methods are KNN and PCA. Default is 'PCA'. |

## Value

A matrix with the same dimensions as InptGEX, with missing values imputed.

## Examples

```
## Not run:
 MatToImpute <-
   matrix(
     rnorm(n = 100000, mean = 0, sd = 100),
     ncol = 10,
     dimnames = list(1:10000, 1:10)
   )
 MatToImpute[sample(x = 1:length(MatToImpute),
                    size = length(MatToImpute) / 20)] <- NA
 ImputedMat <-
   PhysioSpaceMethods:::.imputeMissingGeneExpression(InptGEX = MatToImpute,
   METHOD = "PCA")

## End(Not run)
```

---

# .inptChecker *Checking calculatePhysioMap Inputs*

---

## Description

.inptChecker is an internal function used by calculatePhysioMap for checking the format of inputs, 'InputData' and 'Space' to be exact. It 1- checks to see if both 'InputData' and 'Space' are matrices, and 2- matches the rows of 'InputData' or 'Space' based on their row names.

## Usage

```
.inptChecker(InputData, Space)
```

## Arguments

| | |
|---|---|
| InputData | A matrix, SummarizedExperiment object or a list, based on the gene expression data (or any other type of high dimensional data, e.g. protein abundance, SNP, Methylation, etc.), to be analysed. InputData has to have a specific format to be properly analysed, these requirements are thoroughly explained in the 'Details' section of calculatePhysioMap() function. |
| Space | The space in which the 'InputData' will be mapped. Just as 'InputData', it should be a matrix with genes as rows and samples as columns, with corresponding Entrez Gene IDs in 'rownames' of the matrix, and name of each axis of the space written in 'colnames'. |

## Value

.inptChecker returns corrected 'InputData' and 'Space' directly to the environment it was called from (By assigning new matrices to parent.frame()).

## Examples

```
## Not run:
 SimulatedGeneExpressionData <-
   matrix(
     rnorm(n = 100000, mean = 0, sd = 100),
     ncol = 10,
     dimnames = list(1:10000, 1:10)
   )
 PhysioSpaceMethods:::.inptChecker(InputData =
                                     SimulatedGeneExpressionData[, 1:5],
              Space = SimulatedGeneExpressionData[sample(1:10000), 6:10])

## End(Not run)
```

---

`.inptPreparer` *Preparing InputData for Calculation*

---

## Description

.inptPreparer is an internal function used by .inptChecker and spaceMaker to set up row and column names of 'InputData', and convert it to a matrix, if necessary.

## Usage

```
.inptPreparer(InputData)
```

## Arguments

| | |
|---|---|
| `InputData` | A matrix, SummarizedExperiment object or a list, based on the gene expression data (or any other type of high dimensional data, e.g. protein abundance, SNP, Methylation, etc.), to be analysed. InputData has to have a specific format to be properly analysed, these requirements are thoroughly explained in the 'Details' section of calculatePhysioMap() function. |

## Value

.inptPreparer returns InputData as a matrix with proper row and column names.

## Examples

```
## Not run:
 SimulatedGeneExpressionData <- matrix(
   rnorm(n = 100000, mean = 0,
         sd = 100),
   ncol = 10,
   dimnames = list(1:10000, 1:10)
 )
 SimulatedGeneExpressionData_checked <-
     PhysioSpaceMethods:::.inptPreparer(SimulatedGeneExpressionData)

 library(SummarizedExperiment)
 SimulatedGeneExpressionData_SE <- SummarizedExperiment(
```

```
    assays = list(GEX = SimulatedGeneExpressionData),
    rowData = data.frame("EntrezID" =
                         rownames(SimulatedGeneExpressionData)),
    colData = data.frame("SampleName" =
                         colnames(SimulatedGeneExpressionData))
)
SimulatedGeneExpressionData_SE_checked <-
    PhysioSpaceMethods:::.inptPreparer(SimulatedGeneExpressionData_SE)

## End(Not run)
```

---

`.singleThreadOfPhysioCalc` *Mapping one sample into a physiological-space*

---

## Description

.singleThreadOfPhysioCalc is an internal function of calculatePhysioMap, computing the main mapping. We don't recommend the use of .singleThreadOfPhysioCalc outside of calculatePhysioMap().

## Usage

```
.singleThreadOfPhysioCalc(
  SampleNum,
  InputData,
  Space,
  GenesRatio,
  NGenes,
  STATICResponse,
  pb,
  TTEST
)
```

## Arguments

| | |
|---|---|
| SampleNum | A sample (column) number of InputData. |
| InputData | A matrix, SummarizedExperiment object or a list, based on the gene expression data (or any other type of high dimensional data, e.g. protein abundance, SNP, Methylation, etc.), to be analysed. InputData has to have a specific format to be properly analysed, these requirements are thoroughly explained in the 'Details' section of calculatePhysioMap() function. |
| Space | The space in which the 'InputData' will be mapped. Just as 'InputData', it should be a matrix with genes as rows and samples as columns, with corresponding Entrez Gene IDs in 'rownames' of the matrix, and name of each axis of the space written in 'colnames'. |
| GenesRatio | The ratio of gene expression values to be considered in the calculation. In high dimensional omics data, signal to noise ratio has a direct relation with the relative magnitude of expressions. We aim to remove the noisy genes, hence we only keep the "GenesRatio*100" percent highest and lowest gene expression |

| | |
|---|---|
| | values of each sample. GenesRatio should be a numerical value between 0 and 1. Default value is 0.05. |
| NGenes | Number of genes (rows) in Space. |
| STATICResponse | Logical value indicating if 'statistic' should be returned rather than the default 'signed p value'. Default value is FALSE. |
| pb | Progress bar made by progress::progress_bar$new. |
| TTEST | Logical value indicating if t.test should be done in place of the default wilcoxon rank-sum test (more info can be found in the original PhysioSpace: Lenz et. al., PLOS One 2013). Using t.test will speed up calculations. Default value is FALSE. |

## Value

Vector of mapped 'InputData[,SampleNum]' values in 'Space'. Mapped values are signed p value when STATICResponse==FALSE, and are 'statistic' value when STATICResponse==TRUE (more info can be found in the original PhysioSpace: Lenz et. al., PLOS One 2013).

---

# .tTest *T-testing Between Plus and Minus Genes*

---

## Description

.tTest is an internal function used by calculatePhysioMap. It does a t-test between iplus and iminus indexed genes in ReferencesJ.

## Usage

```
.tTest(ReferencesJ, iplus, iminus, STATICResponse)
```

## Arguments

| | |
|---|---|
| ReferencesJ | Vector of gene expressions to do statistical test on. |
| iplus | Index of first group of genes for statistical testing. |
| iminus | Index of second group of genes for statistical testing. |
| STATICResponse | Same STATICResponse as in calculatePhysioMap. Check calculatePhysioMap's help for more info. |

## Value

Log2 signed p value of t-test if STATICResponse==FALSE, t-test statisitic if STATICResponse==TRUE.

## Examples

```
## Not run:
 SimulatedReferenceSpace <-
   matrix(
     rnorm(n = 100000, mean = 0, sd = 100),
     ncol = 10,
     dimnames = list(1:10000, 11:20)
   )
```

```
 PhysioSpaceMethods:::.tTest(
   ReferencesJ = SimulatedReferenceSpace[, 4],
   iplus = sample(
     1:nrow(SimulatedReferenceSpace),
     size = nrow(SimulatedReferenceSpace) / 20
   ),
   iminus = sample(
     1:nrow(SimulatedReferenceSpace),
     size = nrow(SimulatedReferenceSpace) / 20
   ),
   STATICResponse = FALSE
 )

## End(Not run)
```

---

`.wilTest` *Wilcoxon Rank Sum testing Between Plus and Minus Genes*

---

## Description

.wilTest is an internal function used by calculatePhysioMap that does rank sum test (equivalent to Mann-Whitney test) between iplus and iminus indexed genes in ReferencesJ.

## Usage

```
.wilTest(ReferencesJ, iplus, iminus, STATICResponse)
```

## Arguments

| | |
|---|---|
| `ReferencesJ` | Vector of gene expressions to do statistical test on. |
| `iplus` | Index of first group of genes for statistical testing. |
| `iminus` | Index of second group of genes for statistical testing. |
| `STATICResponse` | Same STATICResponse as in calculatePhysioMap. Check calculatePhysioMap's help for more info. |

## Value

Log2 signed p value of Rank sum test if STATICResponse==FALSE, Rank sum statisitic normalized between -1 and 1 if STATICResponse==TRUE.

## Examples

```
## Not run:
 SimulatedReferenceSpace <-
   matrix(
     rnorm(n = 100000, mean = 0, sd = 100),
     ncol = 10,
     dimnames = list(1:10000, 11:20)
   )
 PhysioSpaceMethods:::.wilTest(
   ReferencesJ = SimulatedReferenceSpace[, 9],
   iplus = sample(
```

```
    1:nrow(SimulatedReferenceSpace),
    size = nrow(SimulatedReferenceSpace) / 20
  ),
  iminus = sample(
    1:nrow(SimulatedReferenceSpace),
    size = nrow(SimulatedReferenceSpace) / 20
  ),
  STATICResponse = FALSE
)

## End(Not run)
```

---

`calculatePhysioMap` *Mapping new data into a physiological-space*

---

## Description

calculatePhysioMap computes mapped values of each input sample inside of a space, calculated prior from a compendium of known samples.

## Usage

```
calculatePhysioMap(
  InputData,
  Space,
  GenesRatio = 0.05,
  NumbrOfCores = 1,
  TTEST = FALSE,
  STATICResponse = FALSE,
  ImputationMethod = "PCA"
)
```

## Arguments

| | |
|---|---|
| `InputData` | A matrix, SummarizedExperiment object or a list, based on the gene expression data (or any other type of high dimensional data, e.g. protein abundance, SNP, Methylation, etc.), to be analysed. InputData has to have a specific format to be properly analysed, these requirements are thoroughly explained in the 'Details' section. |
| `Space` | The space in which the 'InputData' will be mapped. Just as 'InputData', it should be a matrix with genes as rows and samples as columns, with corresponding Entrez Gene IDs in 'rownames' of the matrix, and name of each axis of the space written in 'colnames'. |
| `GenesRatio` | The ratio of gene expression values to be considered in the calculation. In high dimensional omics data, signal to noise ratio has a direct relation with the relative magnitude of expressions. We aim to remove the noisy genes, hence we only keep the "GenesRatio*100" percent highest and lowest gene expression values of each sample. GenesRatio should be a numerical value between 0 and 1. Default value is 0.05. |

`NumbrOfCores` Number of cpu-cores to be used. Default is 1 which will result in the program running in serial. If you assign a number higher than 1, BiocParallel::MulticoreParam is called to make a parallel back-end to use. Assigning a number higher than parallel::detectCores() will result in an error. You can also pass a BiocParallel-Param instance to be used as parallel back-end. Remember that on Windows, the default MulticoreParam back-end doesn't work so you have to use another back-end, e.g. Snow by calling BiocParallel::SnowParam(). For more information, check the examples at the end of this help page or documentation of BiocParallel package.

`TTEST` Logical value indicating if t.test should be done in place of the default wilcoxon rank-sum test (more info can be found in the original PhysioSpace: Lenz et. al., PLOS One 2013). Using t.test will speed up calculations. Default value is FALSE.

`STATICResponse` Logical value indicating if 'statistic' should be returned rather than the default 'signed p value'. Default value is FALSE.

`ImputationMethod`
Imputation method to use in case of missing values. Available methods are "PCA" and "KNN". Default is "PCA".

## Details

PhysioSpace is a robust statistical method for relating high dimensional omics data sets from heterogeneous sources using shared physiological processes. It is designed to take advantage of the vast availability of public omics data, which in combination with statistical approaches makes a potent tool capable of analyzing heterogeneous biological data sets. 'calculatePhysioMap' is the main analytical function of the package. It uses a nonlinear mapping function to relate the unknown input data with a physiological space. Physiological spaces are mathematical spaces build upon known physiological data, using the 'spaceMaker' function.

When preparing the InputData, specific requirements are needed to be met:

1- In case of a matrix, InputData is supposed to be the gene expressions matrix to be analyzed, with genes as rows and samples as columns. Corresponding Entrez Gene IDs must be assigned to 'rownames' of the matrix, and name of each sample/column should be written in 'colnames'. REMEMBER that the gene expressions in 'InputData' should be relative; e.g. fold change or signed p value of a statistical test.

2- In case of a SummarizedExperiment object, InputData must have a component named 'EntrezID' in its rowData. It is also expected (but not mandatory) for InputData to have a component named 'SampleName' in its colData. The gene expressions in 'InputData' is extracted by the function 'assay()', meaning in case 'InputData' contains multiple assays, only the first one is used. REMEMBER that the assay should contain relative gene expression data; e.g. fold change or signed p value of a statistical test.

3- In case user has their own list of significantly up and down regulated genes, it is also possible for InputData to be a list, containing Entrez IDs (or any other identifier which is used as rownames in 'Space') of up regulated genes in InputData[[1]] and Entrez IDs (or any other identifier which is used as rownames in 'Space') of down regulated genes in InputData[[2]]. Having a list InputData is usually slow and restrictive, hence, list input it is not recommended.

## Value

Matrix of mapped 'InputData' values in 'Space', with rows corresponding to axes of 'Space' and columns representing samples in 'InputData'. Mapped values are signed p value when STATICResponse==FALSE, and are 'statistic' value when STATICResponse==TRUE (more info can be found in the original PhysioSpace paper: Lenz et. al., PLOS One 2013).

## References

Lenz, M., Schuldt, B. M., Müller, F. J., & Schuppert, A. (2013). PhysioSpace: relating gene expression experiments from heterogeneous sources using shared physiological processes. PLoS One, 8(10), e77627.

## Examples

```
SimulatedGeneExpressionData <- matrix(
  rnorm(n = 10000, mean = 0,
        sd = 100),
  ncol = 10,
  dimnames = list(1:1000, 1:10)
)
SimulatedReferenceSpace <- matrix(
  rnorm(n = 10000, mean = 0,
        sd = 100),
  ncol = 10,
  dimnames = list(1:1000, 11:20)
)
calculatePhysioMap(InputData = SimulatedGeneExpressionData,
                   Space = SimulatedReferenceSpace)
if (parallel::detectCores() > 1) {
  #More than one core is needed for parallel processing
  calculatePhysioMap(
    InputData = SimulatedGeneExpressionData,
    Space = SimulatedReferenceSpace,
    NumbrOfCores = 2,
    GenesRatio = 0.01,
    STATICResponse = FALSE,
    TTEST = TRUE
  )
}

library(SummarizedExperiment)
SimulatedGeneExpressionData_SE <- SummarizedExperiment(
    assays = list(GEX = SimulatedGeneExpressionData),
    rowData = data.frame("EntrezID" =
                           rownames(SimulatedGeneExpressionData)),
    colData = data.frame("SampleName" =
                           colnames(SimulatedGeneExpressionData))
)
calculatePhysioMap(InputData = SimulatedGeneExpressionData_SE,
                   Space = SimulatedReferenceSpace)

#Examples for user-defined parallel back-ends:
if (parallel::detectCores() > 1) {
  #More than one core is needed for parallel processing
  library(BiocParallel)
  calculatePhysioMap(
    InputData = SimulatedGeneExpressionData,
    Space = SimulatedReferenceSpace,
    NumbrOfCores = SnowParam(2), #Use this on Windows
    GenesRatio = 0.01,
    STATICResponse = FALSE,
    TTEST = TRUE
  )
```

```
  calculatePhysioMap(
    InputData = SimulatedGeneExpressionData,
    Space = SimulatedReferenceSpace,
    NumbrOfCores = MulticoreParam(2),
    GenesRatio = 0.01,
    STATICResponse = FALSE,
    TTEST = TRUE
  )
}
```

---

PhysioHeatmap *Drawing Heatmap of CalculatePhysioMap's Output*

---

## Description

Draws a custom heatmap based on the result matrix generated by CalculatePhysioMap() function.

## Usage

```
PhysioHeatmap(
  PhysioResults,
  ColorLevels = 100,
  Width = 7,
  Height = 7,
  main = "",
  PlotSize = NA,
  SymmetricColoring = FALSE,
  RowColCex = NA,
  KeyLabelCex = NA,
  SpaceClustering = FALSE,
  Space = NA,
  ReducedPlotting = FALSE
)
```

## Arguments

| | |
|---|---|
| PhysioResults | Matrix of scores generated by CalculatePhysioMap(). |
| ColorLevels | An integer indicating how many colors to use when plotting the heatmap. Default is 100. |
| Width | Width of the output plot, in inches. Default is 7. |
| Height | Height of the output plot, in inches. Default is 7. |
| main | The title of the heatmap. Default is an empty string (no title). |
| PlotSize | A numerical value with which you can zoom in and out of the heatmap. Default is NA, which makes PhysioHeatmap choose the PlotSize automatically. |
| SymmetricColoring | Logical value that determines if color coding should distribute symmetrically around 0. Default is false, which means colors will be distributed from minimum to maximum value of PhysioResults. |

RowColCex
: Row and column cex (a numerical value giving the amount by which plotting text and symbols should be magnified). Default is NA, in which case PhysioHeatmap itself assigns a value to RowColCex based on PhysioResults size.

KeyLabelCex
: Colorkey text labels cex (a numerical value giving the amount by which plotting text and symbols should be magnified relative). Default is NA, in which case PhysioHeatmap itself assigns a value to KeyLabelCex based on PhysioResults size.

SpaceClustering
: Logical value for choosing if the rows of PhysioResults (Space axes) should be ordered using hierarchical clustering. Default is FALSE.

Space
: Space with which PhysioResults is calculated. It is needed if SpaceClustering is TRUE.

ReducedPlotting
: Logical or numeric value indicating if only important rows in PhysioResults should be plotted. If ReducedPlotting is FALSE, all rows of PhysioResults are plotted. If ReducedPlotting is TRUE, for each sample (column in PhysioResults) only the 10 most important rows (axes in Space) are selected and plotted. And the case of ReducedPlotting being a numerical value, e.g. N, is similar to ReducedPlotting == TRUE, except rather than 10, the N most important rows are kept.

## Value

PhysioHeatmap returns(Invisibly) a 'TRUE' logical value.

## Examples

```
randMatInpt <-
   matrix(data = rnorm(n = 4000, mean = 10, sd = 20), nrow = 400)
rownames(randMatInpt) <- paste("ROWS", 1:400)
colnames(randMatInpt) <- paste("Sample", 1:10)

randMatRef <-
   matrix(data = rnorm(n = 12000, mean = 10, sd = 20), nrow = 400)
rownames(randMatRef) <- paste("ROWS", 1:400)
colnames(randMatRef) <- paste("Space", 1:30)

res <-
   calculatePhysioMap(InputData = randMatInpt, Space = randMatRef)

PhysioHeatmap(PhysioResults = res,
            main = "Heatmap Testing")
PhysioHeatmap(
   PhysioResults = res,
   main = "Heatmap Testing",
   ColorLevels = 3
)
PhysioHeatmap(
   PhysioResults = res,
   main = "Heatmap Testing",
   SpaceClustering = TRUE,
   Space = randMatRef
)
PhysioHeatmap(
   PhysioResults = res,
```

```
    main = "Heatmap Testing",
    ReducedPlotting = 2
)
```

---

spaceMaker *Creates PhysioSpaces*

---

## Description

This function uses 'Big Data' to make robust 'Physiological Vectors' in N dimensional spaces, within which you can map new data to extract physiological information from a new data set.

## Usage

```
spaceMaker(
  GeneExMatrix,
  DESIGN = NA,
  CONTRASTs = NA,
  Output = "PhysioScore",
  LinearOrRNASeq = "Linear",
  NumbrOfCores = 1
)
```

## Arguments

| | |
|---|---|
| GeneExMatrix | A matrix of input gene expressions or a SummarizedExperiment object, based on which the Physiological Space is made.<br>In case of a matrix, GeneExMatrix is supposed to have genes as rows and samples as columns. Corresponding Entrez Gene IDs must be assigned to 'rownames', and name of each sample should be written in 'colnames' of the matrix.<br>In case of a SummarizedExperiment object, GeneExMatrix must have a component named 'EntrezID' in its rowData. It is also expected (but not mandatory) for GeneExMatrix to have a component named 'SampleName' in its colData. The gene expressions in GeneExMatrix is extracted by the function assay(), meaning in case GeneExMatrix contains multiple assays, only the first one is used.<br>Unless 'DESIGN' and 'CONTRASTs' inputs are provided by the user, spaceMaker supposes the label of the first column (colnames(GeneExMatrix)[1]) to be the reference of the experiment and uses all the samples with this label as control. |
| DESIGN | (Optional) Design matrix of GeneExMatrix, made by the function model.matrix(). If it's not provided, spaceMaker() will make a design matrix based on sample names of GeneExMatrix. |
| CONTRASTs | (Optional) character vector or list specifying contrasts. If it's not provided, spaceMaker() will make the CONTRASTs with the assumption that sample names of first column is the label of the control or reference. REMEMBER that expected user-defined CONTRASTs format changes based on the LinearOrRNASeq input: in case LinearOrRNASeq='Linear', CONTRASTs is expected to work as an input for limma::makeContrasts(). And when LinearOrRNASeq='RNASeq', CONTRASTs is used as an input for DESeq2::results(). |

`Output` A character specifying the output format of spaceMaker(). The default value is 'PhysioScore', which will return -log2(p value)*sign(fold change). It is also possible to obtain fold change by Output='FoldChange', or obtain the fitted model by having Output ='Model'.

`LinearOrRNASeq` A character which determines what type of modelling is ought to be used when making the PhysioSpace. If it's possible to do linear modelling on the data, e.g. data is log normal micro-array gene expression data or limma::voom-transformed RNA-seq data, then LinearOrRNASeq should be 'Linear'. In this case limma package is used in the calculations. But in case your GeneExMatrix input is an RNA-seq count matrix, you should pass 'RNASeq' to LinearOrRNASeq. In this case DESeq2 package is used for calculations.

`NumbrOfCores` Number of cpu-cores to be used (only in RNASeq mode). Default is 1 which will result in the program running in serial. If you assign a number higher than 1, BiocParallel::MulticoreParam is called to make a parallel back-end to use. Assigning a number higher than parallel::detectCores() will result in an error. You can also pass a BiocParallelParam instance to be used as parallel back-end. Remember that on Windows, the default MulticoreParam back-end doesn't work so you have to use another back-end, e.g. Snow by calling BiocParallel::SnowParam(). For more information, check the documentation of BiocParallel package.

## Value

Depending on the 'Output' argument, the returned value is either a matrix, or a model. If Output = "PhysioScore", a matrix is returned, with genes in rows and Physiological axes on the columns. In this case, values inside this matrix are PhysioScores (-log2(p value)*sign(fold change)). In case of Output = "FoldChange", a matrix of fold changes is returned. And if Output = "Model", the fitted model by limma::lmFit() or DESeq2::DESeq() is returned. REMEMBER that when user provides 'DESIGN' input argument, colnames of the returned matrix remains empty and are needed to be assigned by the user.

## Examples

```
INPTMat <-
  matrix(
    data = rnorm(n = 18000, mean = 8, sd = 6),
    nrow = 2000,
    dimnames = list(paste0("g", 1:2000), c(
      rep("Ctrl", 3), rep("Cancer1", 3), rep("Cancer2", 3)
    ))
  ) #Simulated DNA-array gene expression matrix
LinearSpaceOfINPTMat <- spaceMaker(GeneExMatrix = INPTMat)

INPTMatRNASeq <-
  matrix(
    data = rnbinom(n = 18000, size = 1.5, prob = 0.01),
    nrow = 2000,
    dimnames = list(paste0("g", 1:2000), c(
      rep("Ctrl", 3), rep("Cancer1", 3), rep("Cancer2", 3)
    ))
  ) #Simulated RNA-seq gene expression matrix
NotLinearSpaceOfINPTMatRNASeq <-
  spaceMaker(GeneExMatrix = INPTMatRNASeq, LinearOrRNASeq = "RNASeq")
```

```
library(SummarizedExperiment)
INPTMat_SE <- SummarizedExperiment(
    assays = list(GEX = INPTMat),
    rowData = data.frame("EntrezID" = rownames(INPTMat)),
    colData = data.frame("SampleName" = colnames(INPTMat))
) #Simulated DNA-array gene expression SummarizedExperiment obj.
LinearSpaceOfINPTMat_SE <- spaceMaker(GeneExMatrix = INPTMat_SE)
```

# Index