

# Package ‘GeoTcgaData’

April 15, 2024

**Type** Package

**Title** Processing Various Types of Data on GEO and TCGA

**Version** 2.2.0

**Description** Gene Expression Omnibus(GEO) and The Cancer Genome Atlas (TCGA) provide us with a wealth of data, such as RNA-seq, DNA Methylation, SNP and Copy number variation data. It's easy to download data from TCGA using the gdc tool, but processing these data into a format suitable for bioinformatics analysis requires more work. This R package was developed to handle these data.

**Depends** R (>= 4.2.0)

**License** Artistic-2.0

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**Suggests** knitr, rmarkdown, DESeq2, S4Vectors, ChAMP, impute, tidyR, clusterProfiler, org.Hs.eg.db, edgeR, limma, quantreg, minfi, IlluminaHumanMethylation450kanno.ilmn12.hg19, dearseq, NOISeq, testthat (>= 3.0.0), CATT, TCGAbiolinks, enrichplot, GEOquery, BiocGenerics

**VignetteBuilder** knitr

**Imports** utils, data.table, plyr, cqn, topconfects, stats, SummarizedExperiment, methods

**Language** en-US

**URL** <https://github.com/YuLab-SMU/GeoTcgaData>

**BugReports** <https://github.com/YuLab-SMU/GeoTcgaData/issues>

**biocViews** GeneExpression, DifferentialExpression, RNASeq, CopyNumberVariation, Microarray, Software, DNAMethylation, DifferentialMethylation, SNP, ATACSeq, MethylationArray

**Config/testthat/edition** 3

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

**git\_branch** RELEASE\_3\_18

**git\_last\_commit** e239614

**git\_last\_commit\_date** 2023-10-24

**Repository** Bioconductor 3.18

**Date/Publication** 2024-04-15

**Author** Erqiang Hu [aut, cre] (<<https://orcid.org/0000-0002-1798-7513>>)

**Maintainer** Erqiang Hu <13766876214@163.com>

## R topics documented:

|                                 |           |
|---------------------------------|-----------|
| array_preprocess . . . . .      | 3         |
| cal_mean_module . . . . .       | 3         |
| cluster_array . . . . .         | 4         |
| combine_pvalue . . . . .        | 5         |
| countToFpkm . . . . .           | 5         |
| countToTpm . . . . .            | 6         |
| differential_array . . . . .    | 7         |
| differential_CNV . . . . .      | 8         |
| differential_limma . . . . .    | 9         |
| differential_methy . . . . .    | 10        |
| differential_RNA . . . . .      | 12        |
| differential_SNP . . . . .      | 15        |
| differential_SNP_GEO . . . . .  | 16        |
| differential_SNP_tcga . . . . . | 17        |
| fpkmToTpm . . . . .             | 18        |
| geneExpress . . . . .           | 18        |
| gene_ave . . . . .              | 19        |
| gene_cov . . . . .              | 19        |
| get_geo_array . . . . .         | 20        |
| GSE66705_sample2 . . . . .      | 20        |
| id_conversion_TCGA . . . . .    | 21        |
| kegg_liver . . . . .            | 21        |
| Merge_methy_tcga . . . . .      | 22        |
| module . . . . .                | 22        |
| prepare_chi . . . . .           | 23        |
| profile . . . . .               | 23        |
| repAssign . . . . .             | 24        |
| repRemove . . . . .             | 25        |
| SNP_QC . . . . .                | 25        |
| ventricle . . . . .             | 26        |
| <b>Index</b>                    | <b>27</b> |

---

|                  |                                      |
|------------------|--------------------------------------|
| array_preprocess | <i>Preprocess of Microarray data</i> |
|------------------|--------------------------------------|

---

**Description**

Preprocess of Microarray data

**Usage**

```
array_preprocess(x, missing_value = "knn", string = " /// ")
```

**Arguments**

|               |   |
|---------------|---|
| x             | matrix of Microarray data, each column is a sample, and each row is a gene. |
| missing_value | Method to impute missing expression data, one of "zero" and "knn".          |
| string        | a string, sep of the gene   |

**Value**

matrix

**Examples**

```
arraylist <- get_geo_array("GSE781")  
arraylist <- lapply(arraylist, array_preprocess)
```

---

|                 |   |
|-----------------|---|
| cal_mean_module | <i>Find the mean value of the gene in each module</i> |
|-----------------|---|

---

**Description**

Find the mean value of the gene in each module

**Usage**

```
cal_mean_module(geneExpress, module)
```

**Arguments**

|             |  |
|-------------|--|
| geneExpress | a data.frame of gene expression data. Each column is a sample, and each row is a gene.                   |
| module      | a data.frame of two column. The first column is module name, the second column are genes in this module. |

**Value**

a data.frame, means the mean of gene expression value in the same module

**Examples**

```
data(geneExpress)
data(module)
result <- cal_mean_module(geneExpress, module)
```

---

|               |  |
|---------------|--|
| cluster_array | <i>cluster probes of Microarray data</i> |
|---------------|--|

---

**Description**

cluster probes of Microarray data

**Usage**

```
cluster_array(x, clusterCutoff = 0.7)
```

**Arguments**

**x** matrix of Microarray data, the first is the name of the gene, and the others are the expression value.

**clusterCutoff** Pearson correlation threshold to cut off the hierarchical tree.

**Value**

data.frame

**Examples**

```
arraylist <- get_geo_array("GSE781")
arraylist <- lapply(arraylist, array_preprocess)
arraylist_cluster <- lapply(arraylist, cluster_array)
```

---

|                |  |
|----------------|--|
| combine_pvalue | <i>combine pvalues of SNP difference analysis result</i> |
|----------------|--|

---

**Description**

combine pvalues of SNP difference analysis result

**Usage**

```
combine_pvalue(snpResult, snp2gene, combineMethod = min)
```

**Arguments**

|               |   |
|---------------|---|
| snpResult     | data.frame of SNP difference analysis result.             |
| snp2gene      | data frame of two column: snp and gene.                   |
| combineMethod | Method of combining the pvalue of multiple snp in a gene. |

**Value**

data.frame

**Examples**

```
snpResult <- data.frame(pvalue = runif(100), estimate = runif(100))
rownames(snpResult) <- paste0("snp", seq_len(100))
snp2gene <- data.frame(snp = rownames(snpResult),
  gene = rep(paste0("gene", seq_len(20)), 5))
result <- combine_pvalue(snpResult, snp2gene)
```

---

|             |                              |
|-------------|------------------------------|
| countToFpkm | <i>Convert count to FPKM</i> |
|-------------|------------------------------|

---

**Description**

Convert count to FPKM

**Usage**

```
countToFpkm(counts_matrix, keyType = "SYMBOL", gene_cov)
```

**Arguments**

|               |   |
|---------------|---|
| counts_matrix | a matrix, colnames of counts_matrix are sample name, rownames of counts_matrix are gene symbols |
| keyType       | keyType, one of keytypes(org.Hs.eg.db).   |
| gene_cov      | data.frame of two column, the first column is gene length, the second column is gene GC content |

**Value**

a matrix

**Examples**

```
data(gene_cov)
lung_squ_count2 <- matrix(c(1, 2, 3, 4, 5, 6, 7, 8, 9), ncol = 3)
rownames(lung_squ_count2) <- c("DISC1", "TCOF1", "SPPL3")
colnames(lung_squ_count2) <- c("sample1", "sample2", "sample3")
result <- countToFpm(lung_squ_count2,
  keyType = "SYMBOL",
  gene_cov = gene_cov
)
```

---

countToTpm

*Convert count to Tpm*

---

**Description**

Convert count to Tpm

**Usage**

```
countToTpm(counts_matrix, keyType = "SYMBOL", gene_cov)
```

**Arguments**

`counts_matrix` a matrix, colnames of counts\_matrix are sample name, rownames of counts\_matrix are gene symbols

`keyType` keyType, one of keytypes(org.Hs.eg.db).

`gene_cov` data.frame of two column, the first column is gene length, the second column is gene GC content

**Value**

a matrix

**Examples**

```
data(gene_cov)
lung_squ_count2 <- matrix(c(1, 2, 3, 4, 5, 6, 7, 8, 9), ncol = 3)
rownames(lung_squ_count2) <- c("DISC1", "TCOF1", "SPPL3")
colnames(lung_squ_count2) <- c("sample1", "sample2", "sample3")
result <- countToTpm(lung_squ_count2,
  keyType = "SYMBOL",
  gene_cov = gene_cov
)
```

---

differential\_array      *Differential analysis of Microarray data*

---

## Description

Differential analysis of Microarray data

## Usage

```
differential_array(df, group, method = "limma", adjust.method = "BH")
```

## Arguments

|               |  |
|---------------|--|
| df            | data.frame of the omic data, each column is a sample, and each row is a gene.                    |
| group         | a vector, group of samples.  |
| method        | method to do differential analysis, one of "limma", "ttest", "wilcox".                           |
| adjust.method | adjust.method, one of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", and "none". |

## Value

data.frame

## Examples

```
library(GeoTcgaData)
library(data.table)
# Use real GEO data as example
arrayData <- read.table("GSE54807_series_matrix.txt.gz",
  sep = "\t", header = TRUE,
  fill=TRUE, comment.char = "!", check.names=FALSE)
gpl <- fread("GPL6244-17930.txt", sep = "\t", header = TRUE)
gpl <- gpl[, c("ID", "gene_assignment")]
class(gpl) <- "data.frame"

for (i in seq_len(nrow(gpl))) {
  aa <- strsplit(gpl[i, 2], " // ")[[1]][5]
  gpl[i, 2] <- as.character(strsplit(aa, " /// ")[[1]][1])
}
gpl[,1] <- as.character(gpl[,1])
arrayData[, 1] <- as.character(arrayData[, 1])
rownames(gpl) <- gpl[, 1]
arrayData[, 1] <- gpl[arrayData[, 1], 2]

arrayData <- repRemove(arrayData, " /// ")

# Remove rows that do not correspond to genes
```

```

arrayData <- arrayData[!is.na(arrayData[, 1]), ]
arrayData <- arrayData[!arrayData[, 1] == "", ]
arrayData <- arrayData[!arrayData[, 1] == "---", ]

arrayData <- arrayData[order(arrayData[, 1]), ]
arrayData <- gene_ave(arrayData, 1)

keep <- apply(arrayData, 1, function(x) sum(x < 1) < (length(x)/2))
arrayData <- arrayData[keep, ]

group <- c(rep("group1", 12), rep("group2", 12))
result <- differential_array(df = arrayData, group = group)

# Use random data as example
arrayData <- matrix(runif(200), 25, 8)
rownames(arrayData) <- paste0("gene", 1:25)
colnames(arrayData) <- paste0("sample", 1:8)
group <- c(rep("group1", 4), rep("group2", 4))
result <- differential_array(df = arrayData, group = group)

```

---

differential\_CNV

*Do difference analysis of gene level copy number variation data*


---

## Description

Do difference analysis of gene level copy number variation data

## Usage

```

differential_CNV(
  cnvData,
  sampleGroup,
  method = "Chisquare",
  adjust.method = "BH",
  ...
)

```

## Arguments

|               |   |
|---------------|---|
| cnvData       | data.frame of CNV data, each column is a sample, and each row is a CNV.                                   |
| sampleGroup   | vector of sample group  |
| method        | method to do diffenential analysis, one of "Chisquare", "fisher", and "CATT"(Cochran-Armitage trend test) |
| adjust.method | adjust.method, one of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", and "none".          |
| ...           | parameters for "Chisquare", "fisher", and "CATT"(Cochran-Armitage trend test)                             |



**Value**

data.frame with pvalue and estimate

**Examples**

```
# use TCGAbiolinks data as example
library(TCGAbiolinks)
query <- GDCquery(
  project = "TCGA-ACC",
  data.category = "Copy Number Variation",
  data.type = "Gene Level Copy Number",
  access = "open"
)
GDCdownload(query)
cnvData <- GDCprepare(query)
aa <- assays(cnvData)$copy_number
bb <- aa
aa[bb == 2] <- 0
aa[bb < 2] <- -1
aa[bb > 2] <- 1
sampleGroup <- sample(c("A", "B"), ncol(cnvData), replace = TRUE)
diffCnv <- differential_CNV(aa, sampleGroup)

# Use sangerbox CNV data as example
cnvData <- fread("Merge_GeneLevelCopyNumber.txt")
class(cnvData) <- "data.frame"
rownames(cnvData) <- cnvData[, 1]
cnvData <- cnvData[, -c(1, 2, 3)]
sampleGroup <- sample(c("A", "B"), ncol(cnvData), replace = TRUE)
diffCnv <- differential_CNV(cnvData, sampleGroup)

# use random data as example
aa <- matrix(sample(c(0, 1, -1), 200, replace = TRUE), 25, 8)
rownames(aa) <- paste0("gene", 1:25)
colnames(aa) <- paste0("sample", 1:8)
sampleGroup <- sample(c("A", "B"), ncol(aa), replace = TRUE)
diffCnv <- differential_CNV(aa, sampleGroup)
```

---

differential\_limma      *differential\_limma*

---

**Description**

differential\_limma

**Usage**

```
differential_limma(df, group, adjust.method = "BH")
```

**Arguments**

df                    data.frame of the omic data  
group                a vector, group of samples.  
adjust.method      adjust.method.

**Value**

data.frame

**Examples**

```
df <- matrix(runif(200), 25, 8)
df <- as.data.frame(df)
rownames(df) <- paste0("gene", 1:25)
colnames(df) <- paste0("sample", 1:8)
group <- sample(c("group1", "group2"), 8, replace = TRUE)
result <- differential_limma(df = df, group = group)
```

---

differential\_methy      *differential\_methy*

---

**Description**

Get methylation difference gene

**Usage**

```
differential_methy(  
  cpaData,  
  sampleGroup,  
  groupCol,  
  combineMethod = "stouffer",  
  missing_value = "knn",  
  cpG2gene = NULL,  
  normMethod = "PBC",  
  region = "TSS1500",  
  model = "gene",  
  adjust.method = "BH",  
  adjPvalCutoff = 0.05,  
  ucscData = FALSE  
)
```

**Arguments**

|               |   |
|---------------|---|
| cpgData       | data.frame of cpg beta value, , or SummarizedExperiment object  |
| sampleGroup   | vector of sample group  |
| groupCol      | group column  |
| combineMethod | method to combine the cpg pvalues, a function or one of "stouffer", "fisher" and "rhoScores".   |
| missing_value | Method to impute missing expression data, one of "zero" and "knn".  |
| cpg2gene      | data.frame to annotate cpg locus to gene  |
| normMethod    | Method to do normalization: "PBC" or "BMIQ".  |
| region        | region of genes, one of "Body", "TSS1500", "TSS200", "3'UTR", "1stExon", "5'UTR", and "IGR". Only used when cpg2gene is NULL.   |
| model         | if "cpg", step1: calculate difference cpgs; step2: calculate difference genes. if "gene", step1: calculate the methylation level of genes; step2: calculate difference genes. |
| adjust.method | character string specifying the method used to adjust p-values for multiple testing. See <a href="#">p.adjust</a> for possible values.  |
| adjPvalCutoff | adjusted pvalue cutoff  |
| ucscData      | Logical, whether the data comes from UCSC Xena.   |

**Value**

data.frame

**Examples**

```
# use TCGAbiolinks data
library(TCGAbiolinks)
query <- GDCquery(project = "TCGA-ACC",
  data.category = "DNA Methylation",
  data.type = "Methylation Beta Value",
  platform = "Illumina Human Methylation 450")
GDCdownload(query, method = "api", files.per.chunk = 5,
  directory = Your_Path)
merge_result <- Merge_methy_tcga(Your_Path_to_DNA_Methylation_data)
library(ChAMP) # To avoid reporting errors
differential_gene <- differential_methy(cpgData = merge_result,
  sampleGroup = sample(c("C", "T"),
  ncol(merge_result[[1]]), replace = TRUE))

# use user defined data
library(ChAMP)
cpaData <- matrix(runif(2000), nrow = 200, ncol = 10)
rownames(cpaData) <- paste0("cpg", seq_len(200))
colnames(cpaData) <- paste0("sample", seq_len(10))
sampleGroup <- c(rep("group1", 5), rep("group2", 5))
names(sampleGroup) <- colnames(cpaData)
cpa2gene <- data.frame(cpg = rownames(cpaData),
```

```

    gene = rep(paste0("gene", seq_len(20)), 10))
result <- differential_methy(cpgData, sampleGroup,
    cpg2gene = cpg2gene, normMethod = NULL)
# use SummarizedExperiment object input
library(ChAMP)
cpgData <- matrix(runif(2000), nrow = 200, ncol = 10)
rownames(cpgData) <- paste0("cpg", seq_len(200))
colnames(cpgData) <- paste0("sample", seq_len(10))
sampleGroup <- c(rep("group1", 5), rep("group2", 5))
names(sampleGroup) <- colnames(cpgData)
cpg2gene <- data.frame(cpg = rownames(cpgData),
    gene = rep(paste0("gene", seq_len(20)), 10))
colData <- S4Vectors::DataFrame(
    row.names = colnames(cpgData),
    group = sampleGroup
)
data <- SummarizedExperiment::SummarizedExperiment(
    assays=S4Vectors::SimpleList(counts=cpgData),
    colData = colData)
result <- differential_methy(cpgData = data,
    groupCol = "group", normMethod = NULL,
    cpg2gene = cpg2gene)

```

---

differential\_RNA      *differential\_RNA*

---

## Description

Do difference analysis of RNA-seq data

## Usage

```

differential_RNA(
  counts,
  group,
  groupCol,
  method = "limma",
  geneLength = NULL,
  gccontent = NULL,
  filter = TRUE,
  edgeRNorm = TRUE,
  adjust.method = "BH",
  useTopconfects = TRUE,
  ucscData = FALSE
)

```

**Arguments**

|                |  |
|----------------|--|
| counts         | a dataframe or numeric matrix of raw counts data, or SummarizedExperiment object   |
| group          | sample groups  |
| groupCol       | group column   |
| method         | one of "DESeq2", "edgeR", "limma", "dearseq", "NOISeq", "Wilcoxon", and "auto".  |
| geneLength     | a vector of gene length.   |
| gccontent      | a vector of gene GC content.   |
| filter         | if TRUE, use filterByExpr to filter genes.   |
| edgeRNorm      | if TRUE, use edgeR to do normalization for dearseq method.   |
| adjust.method  | character string specifying the method used to adjust p-values for multiple testing. See <a href="#">p.adjust</a> for possible values. |
| useTopconfects | if TRUE, use topconfects to provide a more biologically useful ranked gene list.   |
| ucscData       | Logical, whether the data comes from UCSC Xena.  |

**Value**

data.frame

**Examples**

```
library(TCGAbiolinks)

query <- GDCquery(
  project = "TCGA-ACC",
  data.category = "Transcriptome Profiling",
  data.type = "Gene Expression Quantification",
  workflow.type = "STAR - Counts"
)

GDCdownload(query,
  method = "api", files.per.chunk = 3,
  directory = Your_Path
)

dataRNA <- GDCprepare(
  query = query, directory = Your_Path,
  save = TRUE, save.filename = "dataRNA.RData"
)
## get raw count matrix
dataPrep <- TCGAanalyze_Preprocessing(
  object = dataRNA,
  cor.cut = 0.6,
  datatype = "STAR - Counts"
)
```

```

# Use `differential_RNA` to do difference analysis.
# We provide the data of human gene length and GC content in `gene_cov`.
group <- sample(c("grp1", "grp2"), ncol(dataPrep), replace = TRUE)
library(cqn) # To avoid reporting errors: there is no function "rq"
## get gene length and GC content
library(org.Hs.eg.db)
genes_bitr <- bitr(rownames(gene_cov),
  fromType = "ENTREZID", toType = "ENSEMBL",
  OrgDb = org.Hs.eg.db, drop = TRUE
)
genes_bitr <- genes_bitr[!duplicated(genes_bitr[, 2]), ]
gene_cov2 <- gene_cov[genes_bitr$ENTREZID, ]
rownames(gene_cov2) <- genes_bitr$ENSEMBL
genes <- intersect(rownames(dataPrep), rownames(gene_cov2))
dataPrep <- dataPrep[genes, ]
geneLength <- gene_cov2(genes, "length")
gccontent <- gene_cov2(genes, "GC")
names(geneLength) <- names(gccontent) <- genes
## Difference analysis
DEGAll <- differential_RNA(
  counts = dataPrep, group = group,
  geneLength = geneLength, gccontent = gccontent
)
# Use `clusterProfiler` to do enrichment analytics:
diffGenes <- DEGAll$logFC
names(diffGenes) <- rownames(DEGAll)
diffGenes <- sort(diffGenes, decreasing = TRUE)
library(clusterProfiler)
library(enrichplot)
library(org.Hs.eg.db)
gsego <- gseGO(gene = diffGenes, OrgDb = org.Hs.eg.db, keyType = "ENSEMBL")
dotplot(gsego)

# use user-defined data
df <- matrix(rnbinom(400, mu = 4, size = 10), 25, 16)
df <- as.data.frame(df)
rownames(df) <- paste0("gene", 1:25)
colnames(df) <- paste0("sample", 1:16)
group <- sample(c("group1", "group2"), 16, replace = TRUE)
result <- differential_RNA(counts = df, group = group,
  filte = FALSE, method = "Wilcoxon")
# use SummarizedExperiment object input
df <- matrix(rnbinom(400, mu = 4, size = 10), 25, 16)
rownames(df) <- paste0("gene", 1:25)
colnames(df) <- paste0("sample", 1:16)
group <- sample(c("group1", "group2"), 16, replace = TRUE)

nrows <- 200; ncols <- 20
counts <- matrix(
  runif(nrows * ncols, 1, 1e4), nrows,
  dimnames = list(paste0("cg", 1:200), paste0("S", 1:20))
)

```

```
colData <- S4Vectors::DataFrame(
  row.names = paste0("sample", 1:16),
  group = group
)
data <- SummarizedExperiment::SummarizedExperiment(
  assays=S4Vectors::SimpleList(counts=df),
  colData = colData)

result <- differential_RNA(counts = data, groupCol = "group",
  filte = FALSE, method = "Wilcoxon")
```

---

differential\_SNP      *Do difference analysis of SNP data*

---

## Description

Do difference analysis of SNP data

## Usage

```
differential_SNP(snpDf, sampleGroup, combineMethod = min)
```

## Arguments

snpDf                data.frame of SNP data, each column is a sample, and each row is a SNP.  
sampleGroup        vector of sample group.  
combineMethod     Method of combining the pvalue of multiple snp in a gene.

## Value

data.frame

## Examples

```
library(TCGAbiolinks)
query <- GDCquery(
  project = "TCGA-CHOL",
  data.category = "Simple Nucleotide Variation",
  access = "open",
  legacy = FALSE,
  data.type = "Masked Somatic Mutation",
  workflow.type = "Aliquot Ensemble Somatic Variant Merging and Masking"
)
GDCdownload(query)
data_snp <- GDCprepare(query)
samples <- unique(data_snp$Tumor_Sample_Barcode)
sampleGroup <- sample(c("A", "B"), length(samples), replace = TRUE)
names(sampleGroup) <- samples
pvalue <- differential_SNP_tcga(snpData = data_snp,
```

```

    sampleGroup = sampleGroup)

# use demo data
snpDf <- matrix(sample(c("mutation", NA), 100, replace = TRUE), 10, 10)
snpDf <- as.data.frame(snpDf)
sampleGroup <- sample(c("A", "B"), 10, replace = TRUE)
result <- differential_SNP(snpDf, sampleGroup)

```

---

differential\_SNP\_GEO *Do difference analysis of SNP data downloaded from GEO*

---

### Description

Do difference analysis of SNP data downloaded from GEO

### Usage

```
differential_SNP_GEO(snpData, sampleGroup, method = "Chisquare")
```

### Arguments

|             |   |
|-------------|---|
| snpData     | data.frame of SNP data downloaded from GEO                            |
| sampleGroup | vector of sample group  |
| method      | one of "Chisquare", "fisher", and "CATT"(Cochran-Armitage trend test) |

### Value

data.frame

### Examples

```

file1 <- read.table("GSE66903_series_matrix.txt.gz",
  fill=TRUE, comment.char="!", header = TRUE)
rownames(file1) <- file1[, 1]
snpData <- file1[, -1]
sampleGroup <- sample(c("A", "B"), ncol(snpData ), replace = TRUE)
names(sampleGroup) <- colnames(snpData)
snpData <- SNP_QC(snpData)
sampleGroup <- sample(c("A", "B"), ncol(snpData ), replace = TRUE)
result1 <- differential_SNP_GEO(snpData = snpData,
  sampleGroup = sampleGroup, method = "Chisquare")

# use demo data
snpDf <- matrix(sample(c("AA", "Aa", "aa"), 100, replace = TRUE), 10, 10)
snpDf <- as.data.frame(snpDf)
sampleGroup <- sample(c("A", "B"), 10, replace = TRUE)
result <- differential_SNP_GEO(snpDf, sampleGroup, method = "fisher")

```



---

differential\_SNP\_tcga *Do difference analysis of SNP data downloaded from TCGAbiolinks*

---

## Description

Do difference analysis of SNP data downloaded from TCGAbiolinks

## Usage

```
differential_SNP_tcga(snpData, sampleGroup, combineMethod = NULL)
```

## Arguments

|               |   |
|---------------|---|
| snpData       | data.frame of SNP data downloaded from TCGAbiolinks       |
| sampleGroup   | vector of sample group                                    |
| combineMethod | Method of combining the pvalue of multiple snp in a gene. |

## Value

data.frame

## Examples

```
library(TCGAbiolinks)
query <- GDCquery(
  project = "TCGA-CHOL",
  data.category = "Simple Nucleotide Variation",
  access = "open",
  legacy = FALSE,
  data.type = "Masked Somatic Mutation",
  workflow.type = "Aliquot Ensemble Somatic Variant Merging and Masking"
)
GDCdownload(query)
data_snp <- GDCprepare(query)
samples <- unique(data_snp$Tumor_Sample_Barcode)
sampleGroup <- sample(c("A", "B"), length(samples), replace = TRUE)
names(sampleGroup) <- samples
pvalue <- differential_SNP_tcga(snpData = data_snp,
  sampleGroup = sampleGroup)

# use demo data
snpDf <- matrix(sample(c("mutation", NA), 100, replace = TRUE), 10, 10)
snpDf <- as.data.frame(snpDf)
sampleGroup <- sample(c("A", "B"), 10, replace = TRUE)
result <- differential_SNP(snpDf, sampleGroup)
```

fpkmToTpm                      *Convert fpkm to Tpm*

---

**Description**

Convert fpkm to Tpm

**Usage**

```
fpkmToTpm(fpkm_matrix)
```

**Arguments**

fpkm\_matrix      a matrix, colnames of fpkm\_matrix are sample name, rownames of fpkm\_matrix are genes

**Value**

a matrix

**Examples**

```
lung_squ_count2 <- matrix(c(0.11, 0.22, 0.43, 0.14, 0.875,  
                          0.66, 0.77, 0.18, 0.29), ncol = 3)  
rownames(lung_squ_count2) <- c("DISC1", "TCOF1", "SPPL3")  
colnames(lung_squ_count2) <- c("sample1", "sample2", "sample3")  
result <- fpkmToTpm(lung_squ_count2)
```

---

geneExpress                      *a data.frame of gene expression data*

---

**Description**

It is a randomly generated expression data used as an example of functions in this package. the rowname is gene symbols the columns are gene expression values

**Usage**

```
geneExpress
```

**Format**

A data.frame with 10779 rows and 2 column

---

|          |  |
|----------|--|
| gene_ave | <i>Average the values of same genes in gene expression profile</i> |
|----------|--|

---

**Description**

Average the values of same genes in gene expression profile

**Usage**

```
gene_ave(file_gene_ave, k = 1)
```

**Arguments**

`file_gene_ave` a data.frame of gene expression data, each column is a sample, and each row is a gene.

`k` a number, indicates which is the gene column.

**Value**

a data.frame, the values of same genes in gene expression profile

**Examples**

```
aa <- c("MARCH1", "MARC1", "MARCH1", "MARCH1", "MARCH1")
bb <- c(2.969058399, 4.722410064, 8.165514853, 8.24243893, 8.60815086)
cc <- c(3.969058399, 5.722410064, 7.165514853, 6.24243893, 7.60815086)
file_gene_ave <- data.frame(aa = aa, bb = bb, cc = cc)
colnames(file_gene_ave) <- c("Gene", "GSM1629982", "GSM1629983")

result <- gene_ave(file_gene_ave, 1)
```

---

|          |   |
|----------|---|
| gene_cov | <i>a data.frame of gene length and GC content</i> |
|----------|---|

---

**Description**

the gene length and GC content data comes from TxDb.Hsapiens.UCSC.hg38.knownGene and BSgenome.Hsapiens.UCSC.hg38

**Usage**

```
gene_cov
```

**Format**

A data.frame with 27341 rows and 2 column

---

|               |  |
|---------------|--|
| get_geo_array | <i>Get Microarray matrix data from GEO</i> |
|---------------|--|

---

**Description**

Get Microarray matrix data from GEO

**Usage**

```
get_geo_array(gse)
```

**Arguments**

gse                    GSE number, such as GSE781.

**Value**

a list of matrix

**Examples**

```
arraylist <- get_geo_array("GSE781")
```

---

|                  |  |
|------------------|--|
| GSE66705_sample2 | <i>a matrix of gene expression data in GEO</i> |
|------------------|--|

---

**Description**

the first column represents the gene symbol

**Usage**

```
GSE66705_sample2
```

**Format**

A matrix with 999 rows and 3 column

**Details**

the other columns represent the expression of genes

---

|                    |   |
|--------------------|---|
| id_conversion_TCGA | <i>Convert ENSEMBL gene id to gene Symbol in TCGA</i> |
|--------------------|---|

---

**Description**

Convert ENSEMBL gene id to gene Symbol in TCGA

**Usage**

```
id_conversion_TCGA(profiles, toType = "SYMBOL")
```

**Arguments**

|          |  |
|----------|--|
| profiles | a data.frame of gene expression data, each column is a sample, and each row is a gene. |
| toType   | one of 'keytypes(org.Hs.eg.db)'  |

**Value**

a data.frame, gene symbols and their expression value

**Examples**

```
library(org.Hs.eg.db)
data(profile)
result <- id_conversion_TCGA(profile)
```

---

|            |   |
|------------|---|
| kegg_liver | <i>a matrix of gene expression data in TCGA</i> |
|------------|---|

---

**Description**

It is a randomly generated expression data used as an example of functions in this package. the first column represents the gene symbol

**Usage**

```
kegg_liver
```

**Format**

A matrix with 100 rows and 150 column

**Details**

the other columns represent the expression(count) of genes

---

|                  |  |
|------------------|--|
| Merge_methy_tcga | <i>Merge methylation data downloaded from TCGA</i> |
|------------------|--|

---

**Description**

When the methylation data is downloaded from TCGA, each sample is saved in a folder, which contains the methylation value file and the descriptive file. This function can directly extract and consolidate all folders.

**Usage**

```
Merge_methy_tcga(dirr = NULL)
```

**Arguments**

|      |   |
|------|---|
| dirr | a string for the directory of methylation data download from tcga using the tools gdc |
|------|---|

**Value**

a matrix, a combined methylation expression spectrum matrix

**Examples**

```
merge_result <- Merge_methy_tcga(system.file(file.path("extdata", "methy"),
  package = "GeoTcgaData"))
```

---

|        |  |
|--------|--|
| module | <i>a matrix of module name, gene symbols, and the number of gene symbols</i> |
|--------|--|

---

**Description**

It is a randomly generated expression data used as an example of functions in this package.

**Usage**

```
module
```

**Format**

A matrix with 176 rows and 3 column

---

|             |  |
|-------------|--|
| prepare_chi | <i>Preparer file for chi-square test</i> |
|-------------|--|

---

**Description**

Preparer file for chi-square test

**Usage**

```
prepare_chi(cnv)
```

**Arguments**

cnv                    result of ann\_merge()

**Value**

a matrix

**Examples**

```
cnv <- matrix(c(
  -1.09150, -1.47120, -0.87050, -0.50880,
  -0.50880, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.601962, 2.621332, 2.621332,
  2.621332, 2.621332, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0,
  2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0
), nrow = 5)
cnv <- as.data.frame(cnv)
rownames(cnv) <- c("AJAP1", "FHAD1", "CLCNKB", "CROCCP2", "AL137798.3")
colnames(cnv) <- c(
  "TCGA-DD-A4NS-10A-01D-A30U-01", "TCGA-ED-A82E-01A-11D-A34Y-01",
  "TCGA-WQ-A9G7-01A-11D-A36W-01", "TCGA-DD-AADN-01A-11D-A40Q-01",
  "TCGA-ZS-A9CD-10A-01D-A36Z-01", "TCGA-DD-A1EB-11A-11D-A12Y-01"
)
cnv_chi_file <- prepare_chi(cnv)
```

---

|         |   |
|---------|---|
| profile | <i>a matrix of gene expression data in TCGA</i> |
|---------|---|

---

**Description**

It is a randomly generated expression data used as an example of functions in this package. the first column represents the gene symbol

**Usage**

```
profile
```

**Format**

A matrix with 10 rows and 10 column

**Details**

the other columns represent the expression(FPKM) of genes

---

|           |   |
|-----------|---|
| repAssign | <i>Handle the case where one id corresponds to multiple genes</i> |
|-----------|---|

---

**Description**

Handle the case where one id corresponds to multiple genes

**Usage**

```
repAssign(input_file, string)
```

**Arguments**

|            |   |
|------------|---|
| input_file | input file, a data.frame or a matrix, the first column should be genes. |
| string     | a string, sep of the gene   |

**Value**

a data.frame, when an id corresponds to multiple genes, the expression value is assigned to each gene

**Examples**

```
aa <- c("MARCH1 /// MMA", "MARCH1", "MARCH2 /// MARCH3",  
        "MARCH3 /// MARCH4", "MARCH1")  
bb <- c("2.969058399", "4.722410064", "8.165514853",  
        "8.24243893", "8.60815086")  
cc <- c("3.969058399", "5.722410064", "7.165514853",  
        "6.24243893", "7.60815086")  
input_file <- data.frame(aa = aa, bb = bb, cc = cc)  
  
repAssign_result <- repAssign(input_file, " /// ")
```



---

|           |   |
|-----------|---|
| repRemove | <i>Handle the case where one id corresponds to multiple genes</i> |
|-----------|---|

---

**Description**

Handle the case where one id corresponds to multiple genes

**Usage**

```
repRemove(input_file, string)
```

**Arguments**

|            |   |
|------------|---|
| input_file | input file, a data.frame or a matrix, the first column should be genes. |
| string     | a string, sep of the gene   |

**Value**

a data.frame, when an id corresponds to multiple genes, the expression value is deleted

**Examples**

```
aa <- c("MARCH1 /// MMA", "MARC1", "MARCH2 /// MARCH3",  
       "MARCH3 /// MARCH4", "MARCH1")  
bb <- c("2.969058399", "4.722410064", "8.165514853",  
       "8.24243893", "8.60815086")  
cc <- c("3.969058399", "5.722410064", "7.165514853",  
       "6.24243893", "7.60815086")  
input_file <- data.frame(aa = aa, bb = bb, cc = cc)  
repRemove_result <- repRemove(input_file, " /// ")
```

---

|        |  |
|--------|--|
| SNP_QC | <i>Do quality control of SNP data downloaded from TCGAbiolinks</i> |
|--------|--|

---

**Description**

Do quality control of SNP data downloaded from TCGAbiolinks

**Usage**

```
SNP_QC(  
  snpData,  
  geon = 0.02,  
  mind = 0.02,  
  maf = 0.05,  
  hwe = 1e-06,  
  miss = "NoCall"  
)
```

**Arguments**

|         |  |
|---------|--|
| snpData | data.frame of SNP data downloaded from TCGAbiolinks  |
| geon    | filters out all variants with missing call rates exceeding the provided value (default 0.02) to be removed     |
| mind    | filters out all samples with missing call rates exceeding the provided value (default 0.02) to be removed      |
| maf     | filters out all variants with minor allele frequency below the provided threshold                              |
| hwe     | filters out all variants which have Hardy-Weinberg equilibrium exact test p-value below the provided threshold |
| miss    | character of miss value  |

**Value**

data.frame

**Examples**

```
# use demo data
snpDf <- matrix(sample(c("AA", "Aa", "aa"), 100, replace = TRUE), 10, 10)
snpDf <- as.data.frame(snpDf)
sampleGroup <- sample(c("A", "B"), 10, replace = TRUE)
result <- SNP_QC(snpDf)
```

---

ventricle

*a matrix of gene expression data in GEO*

---

**Description**

It is a randomly generated expression data used as an example of functions in this package. the first column represents the gene symbol

**Usage**

```
ventricle
```

**Format**

A matrix with 32 rows and 20 column

**Details**

the other columns represent the expression of genes

# Index

- \* **datasets**
  - gene\_cov, [19](#)
  - geneExpress, [18](#)
  - GSE66705\_sample2, [20](#)
  - kegg\_liver, [21](#)
  - module, [22](#)
  - profile, [23](#)
  - ventricle, [26](#)
- array\_preprocess, [3](#)
- cal\_mean\_module, [3](#)
- cluster\_array, [4](#)
- combine\_pvalue, [5](#)
- countToFpkm, [5](#)
- countToTpm, [6](#)
- differential\_array, [7](#)
- differential\_CNV, [8](#)
- differential\_limma, [9](#)
- differential\_methy, [10](#)
- differential\_RNA, [12](#)
- differential\_SNP, [15](#)
- differential\_SNP\_GEO, [16](#)
- differential\_SNP\_tcga, [17](#)
- fpkmToTpm, [18](#)
- gene\_ave, [19](#)
- gene\_cov, [19](#)
- geneExpress, [18](#)
- get\_geo\_array, [20](#)
- GSE66705\_sample2, [20](#)
- id\_conversion\_TCGA, [21](#)
- kegg\_liver, [21](#)
- Merge\_methy\_tcga, [22](#)
- module, [22](#)
- p.adjust, [11](#), [13](#)
- prepare\_chi, [23](#)
- profile, [23](#)
- repAssign, [24](#)
- repRemove, [25](#)
- SNP\_QC, [25](#)
- ventricle, [26](#)