

# Package ‘diggit’

March 6, 2025

**Version** 1.38.0

**Date** 2014-08-22

**Title** Inference of Genetic Variants Driving Cellular Phenotypes

**Author** Mariano J Alvarez <reef103@gmail.com>

**Maintainer** Mariano J Alvarez <reef103@gmail.com>

**Depends** R (>= 3.0.2), Biobase, methods

**Imports** ks, viper(>= 1.3.1), parallel

**Suggests** diggitdata

**Description** Inference of Genetic Variants Driving Cellular Phenotypes  
by the DIGGIT algorithm

**License** file LICENSE

**biocViews** SystemsBiology, NetworkEnrichment, GeneExpression,  
FunctionalPrediction, GeneRegulation

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

**git\_branch** RELEASE\_3\_20

**git\_last\_commit** 826d656

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2025-03-06

## Contents

aecdf . . . . .	2
aqtl . . . . .	2
conditional . . . . .	3
correlation . . . . .	4
diggit-class . . . . .	5
fCNV . . . . .	6
marina . . . . .	7
mutualInfo . . . . .	9
plot,diggit-method . . . . .	10
print,diggit-method . . . . .	10
<b>Index</b>	<b>13</b>

---

aecdf *Approximate empirical commulative distribution function*

---

### Description

This function generates an empirical null model that computes a normalized statistics and p-value

### Usage

```
aecdf(dnull, symmetric = FALSE)
```

### Arguments

dnull	Numerical vector representing the null model
symmetric	Logical, whether the distribution should be treated as symmetric around zero and only one tail should be approximated

### Value

function with two parameters, x and alternative

---

aqtl *Inference of aQTL*

---

### Description

This function infers aQTLs from F-CNVs and VIPER activity

### Usage

```
aqtl(x, ...)

## S4 method for signature 'diggit'
aqtl(x, mr = 0.01, mr.adjust = c("none", "fdr",
  "bonferroni"), fcnv = 0.01, fcnv.adjust = c("none", "fdr", "bonferroni"),
  method = c("spearman", "mi", "pearson", "kendall"), mindy = FALSE,
  cores = 1, verbose = TRUE)
```

### Arguments

x	Object of class diggit
...	Additional parameters to pass to the function
mr	Either a numerical value between 0 and 1 indicating the p-value threshold for the Master Regulator (MR) selection, or a vector of character strings listing the MRs
mr.adjust	Character string indicating the multiple hypothesis test correction for the MRs
fcnv	Either a numerical value between 0 and 1 indicating the p-value threshold for the F-CNV, or a vector of character strings listing the F-CNVs

fcnv.adjust	Character string indicating the multiple hypothesis test correction for the F-CNVs
method	Character string indicating the method for computing the association between F-CNV and regulator activity (aQTL analysis)
mindy	Logical, whether only post-translational modulators of each evaluated TF should be considered as putative genetic driver
cores	Integer indicating the number of cores to use (1 for Windows-based systems)
verbose	Logical, whether progress should be reported

### Value

Updated diggit object with viper and aqtl slots

### Examples

```
data(gbm.expression, package="diggitdata")
data(gbm.cnv, package="diggitdata")
data(gbm.aracne, package="diggitdata")
dobj <- diggitClass(expset=gbmExprs, cnv=gbmCNV, regulon=gbmTFregulon)
dobj <- fCNV(dobj)
dobj <- aqtl(dobj, mr=c("CEBPD", "STAT3"), fcnv.adjust="fdr")
dobj
diggitAqtl(dobj)[, 1:4]
```

---

conditional

*Conditional analysis of CNVs*

---

### Description

This function performs the conditional analysis of fCNVs

### Usage

```
conditional(x, ...)

## S4 method for signature 'diggit'
conditional(x, pheno = "cond", group1, group2 = NULL,
  cnv = 0.2, mr = 0.01, mr.adjust = c("none", "fdr", "bonferroni"),
  modul = 0.01, modul.adjust = c("none", "fdr", "bonferroni"),
  fet.pval = 0.05, cores = 1, verbose = TRUE)
```

### Arguments

x	Object of class diggit
...	Additional parameters to pass to the function
pheno	Character string indicating the feature for sample groups
group1	Character string indicating the treatment group
group2	Optional character string indicating the reference group
cnv	Single number or vector of two numbers indicating the thresholds for CNVs

<code>mr</code>	Either vector of character strings indicating the MR genes, or number indicating the corrected p-value threshold for selecting the MRs
<code>mr.adjust</code>	Character string indicating the multiple-hypothesis correction to apply to the MR p-values
<code>modul</code>	Number indicating the p-value threshold for a modulator to be considered associated with the MR activity
<code>modul.adjust</code>	Character string indicating the multiple-hypothesis correction to apply to the aQTL results
<code>fet.pval</code>	Number indicating the FET p-value threshold for the association between CNVs and sample groups
<code>cores</code>	Integer indicating the number of cores to use (1 for Windows-based systems)
<code>verbose</code>	Logical, whether progress should be reported

**Value**

Object of class `diggit` with conditional analysis results

**Examples**

```
data(gbm.expression, package="diggitdata")
data(gbm.cnv, package="diggitdata")
data(gbm.aracne, package="diggitdata")
dobj <- diggitClass(expset=gbmExprs, cnv=gbmCNV, regulon=gbmTFregulon)
dobj <- fCNV(dobj)
dobj <- aqtl(dobj, mr=c("CEBPD", "STAT3"), fcnv.adjust="fdr", verbose=FALSE)
dobj <- conditional(dobj, pheno="subtype", group1="MES", group2="PN", mr="STAT3", verbose=FALSE)
dobj
```

---

correlation

*Correlation test*

---

**Description**

This function computes the correlation between `x` and `y` given both are numeric vectors, between the columns of `x` if it is a numeric matrix, or between the columns of `x` and `y` if both are numeric matrixes

**Usage**

```
correlation(x, y = NULL, method = c("pearson", "spearman", "kendall"),
  pairwise = FALSE)
```

**Arguments**

<code>x</code>	Numeric vector or matrix
<code>y</code>	Optional numeric vector or matrix
<code>method</code>	Character string indicating the correlation method
<code>pairwise</code>	Logical, whether columns of <code>x</code> and <code>y</code> should be compared in a pairwise manner. <code>x</code> and <code>y</code> must have the same number of columns

**Details**

This function computes correlation and associated p-values

**Value**

Numeric value, vector or matrix of results

**Examples**

```
x <- seq(0, 10, length=50)
y <- x+rnorm(length(x), sd=2)
correlation(x, y)
```

---

diggitt-class	<i>The diggitt class</i>
---------------	--------------------------

---

**Description**

This class stores parameters and results of the diggitt algorithm

This function generates diggitt class objects

**Usage**

```
diggittClass(expset = NULL, cnv = NULL, regulon = NULL, mindy = NULL,
             fcnv = NULL, mr = NULL, viper = NULL, aqtl = NULL,
             conditional = NULL)
```

**Arguments**

expset	ExpressionSet object or numeric matrix of expression data, with features in rows and samples in columns
cnv	Numeric matrix of CNV data
regulon	Regulon class object containing the transcriptional interactome
mindy	Regulon class object containing the post-translational interactome
fcnv	Vector of F-CNV p-values
mr	Vector of master regulator Z-score (NES)
viper	Numeric matrix of VIPER results
aqtl	Numeric matrix of aQTL p-values
conditional	List containing the conditional analysis results

**Details**

see [diggitt-methods](#) for related methods

**Value**

Object of class diggitt

**Slots**

**expset:** ExpressionSet object containing the gene expression data  
**cnv:** Matrix containing the CNV data  
**regulon:** Regulon object containing the transcriptional interactome  
**mindy:** Regulon object containing the post-translational interactome  
**fcnv:** Numeric vector containing the p-values for functional CNVs  
**mr:** Numeric vector of normalized enrichment scores for the MARINA analysis  
**viper:** Numeric matrix of normalized enrichment scores for the VIPER analysis  
**aqtl:** Numeric matrix of association p-values for the aQTL analysis  
**conditional:** List containing the conditional analysis results

**Examples**

```

data(gbm.expression, package="diggitdata")
data(gbm.aracne, package="diggitdata")
dobj <- diggitClass(expset=gbmExprs, regulon=gbmTFregulon)
print(dobj)

```

fCNV

*Inference of functional CNVs***Description**

This function infers functional CNVs by computing their association with gene expression

**Usage**

```

fCNV(x, ...)

## S4 method for signature 'diggit'
fCNV(x, expset = NULL, cnv = NULL,
     method = c("spearman", "mi", "pearson", "kendall"), cores = 1,
     verbose = TRUE)

## S4 method for signature 'ExpressionSet'
fCNV(x, cnv, method = c("spearman", "mi", "pearson",
                       "kendall"), cores = 1, verbose = TRUE)

## S4 method for signature 'matrix'
fCNV(x, cnv, method = c("spearman", "mi", "pearson",
                       "kendall"), cores = 1, verbose = TRUE)

## S4 method for signature 'data.frame'
fCNV(x, cnv, method = c("spearman", "mi", "pearson",
                       "kendall"), cores = 1, verbose = TRUE)

```

**Arguments**

x	Object of class <code>diggit</code> , <code>expressionSet</code> object or numeric matrix of expression data, with features in rows and samples in columns
...	Additional arguments
expset	Optional numeric matrix of expression data
cnv	Optional numeric matrix of CNVs
method	Character string indicating the method for computing the association between CNVs and expression
cores	Integer indicating the number of cores to use (1 for Windows-based systems)
verbose	Logical, whether to report analysis progress

**Value**

Objet of class `diggit` with updated `fCNV` slot

**Examples**

```

data(gbm.expression, package="diggitdata")
data(gbm.cnv, package="diggitdata")
genes <- intersect(rownames(gbmExprs), rownames(gbmCNV))[1:100]
gbmCNV <- gbmCNV[match(genes, rownames(gbmCNV)), ]
dgo <- diggitClass(expset=gbmExprs, cnv=gbmCNV)

dgo <- fCNV(dgo)
dgo
diggitFcv(dgo)[1:5]
dgo <- fCNV(gbmExprs, gbmCNV)
print(dgo)
diggitFcv(dgo)[1:5]
dgo <- fCNV(exprs(gbmExprs), gbmCNV)
dgo
diggitFcv(dgo)[1:5]
dgo <- fCNV(as.data.frame(exprs(gbmExprs)), gbmCNV)
dgo
diggitFcv(dgo)[1:5]

```

**Description**

This function infers the master regulators for the transition between two phenotypes

**Usage**

```
marina(x, ...)
```

```

## S4 method for signature 'matrix'
marina(x, y = NULL, mu = 0, regulon, per = 1000,
       cores = 1, verbose = TRUE)

```

```
## S4 method for signature 'ExpressionSet'
marina(x, pheno = "cond", group1, group2 = NULL,
       mu = 0, regulon, per = 1000, cores = 1, verbose = TRUE)

## S4 method for signature 'diggit'
marina(x, pheno, group1, group2 = NULL, mu = 0,
       regulon = NULL, per = 1000, cores = 1, verbose = TRUE)
```

### Arguments

x	Object of class diggit, expressionSet object or numerical matrix containing the test samples
...	Additional arguments
y	Numerical matrix containing the control samples
mu	Number indicating the control mean when y is omitted
regulon	Transcriptional interactome
per	Integer indicating the number of permutations to compute the marina null model
cores	Integer indicating the number of cores to use (1 for Windows-based systems)
verbose	Logical, whether progress should be reported
pheno	Character string indicating the phenotype data to use
group1	Vector of character strings indicating the category from phenotype pheno to use as test group
group2	Vector of character strings indicating the category from phenotype pheno to use as control group

### Value

Updated diggit object with Master Regulator results

### Examples

```
cores <- 3*(Sys.info()[1] != "Windows")+1
data(gbm.expression, package="diggitdata")
data(gbm.aracne, package="diggitdata")

eset <- exprs(gbmExprs)
samples <- pData(gbmExprs)[["subtype"]]
x <- eset[, samples=="MES"]
y <- eset[, samples=="PN"]
dgo <- marina(x, y, regulon=gbmTFregulon, per=100, cores=cores)
dgo
diggitMR(dgo)[1:5]
dgo <- marina(gbmExprs, pheno="subtype", group1="MES", group2="PN", regulon=gbmTFregulon, per=100, cores=cores)
dgo
diggitMR(dgo)[1:5]
x <- diggitClass(expset=gbmExprs, regulon=gbmTFregulon)
dgo <- marina(x, pheno="subtype", group1="MES", group2="PN", per=100, cores=cores)
dgo
diggitMR(dgo)[1:5]
```



---

mutualInfo	<i>Mutual information</i>
------------	---------------------------

---

### Description

This function estimates the mutual information between  $x$  and  $y$  given both are numeric vectors, between the columns of  $x$  if it is a numeric matrix, or between the columns of  $x$  and  $y$  if both are numeric matrixes

### Usage

```
mutualInfo(x, y = NULL, per = 0, pairwise = FALSE, bw = 100,  
           cores = 1, verbose = TRUE)
```

### Arguments

<code>x</code>	Numeric vector or matrix
<code>y</code>	Optional numeric vector or matrix
<code>per</code>	Integer indicating the number of permutations to compute p-values
<code>pairwise</code>	Logical, wether columns of $x$ and $y$ should be compared in a pairwise maner. $x$ and $y$ must have the same number of columns
<code>bw</code>	Integer indicating the grid size for integrating the joint probability density
<code>cores</code>	Integer indicating the number of cores to use (1 for Windows-based systems)
<code>verbose</code>	Logical, whether progression bars should be shown

### Details

This function estimates the mutual information between continuous variables using a fix bandwidth implementation

### Value

Numeric value, vector or matrix of results

### Examples

```
x <- seq(0, pi, length=100)  
y <- 5*sin(x)+rnorm(100)  
cor.test(x, y)  
mutualInfo(x, y, per=100)
```

---

plot,diggit-method     *Diggit plot*

---

### Description

This function generate plots for the diggit conditional analysis

### Usage

```
## S4 method for signature 'diggit'
plot(x, mr = NULL, cluster = NULL, sub = NULL, ...)
```

### Arguments

x	Diggit class object
mr	Optional vector of character strings indicating the MR names
cluster	Optional vector of cluster names
sub	Optional sub-title for the plot
...	Additional parameters to pass to the plot function

### Value

Nothing, plots are generated in the default output device

### Examples

```
data(gbm.expression, package="diggitdata")
data(gbm.cnv, package="diggitdata")
data(gbm.aracne, package="diggitdata")
dobj <- diggitClass(expset=gbmExprs, cnv=gbmCNV, regulon=gbmTFregulon)
dobj <- fCNV(dobj)
dobj <- aqtl(dobj, mr=c("CEBPD", "STAT3"), fcnv.adjust="fdr", verbose=FALSE)
dobj <- conditional(dobj, pheno="subtype", group1="MES", group2="PN", mr="STAT3", verbose=FALSE)
plot(dobj, cluster="3")
```

---

print,diggit-method     *Basic methods for class diggit*

---

### Description

This document lists a series of basic methods for the class diggit

**Usage**

```
## S4 method for signature 'diggit'  
print(x, pval = 0.05)  
  
## S4 method for signature 'diggit'  
show(object)  
  
## S4 method for signature 'diggit'  
exprs(object)  
  
## S4 method for signature 'diggit'  
diggitCNV(x)  
  
## S4 method for signature 'diggit'  
diggitRegulon(x)  
  
## S4 method for signature 'diggit'  
diggitMindy(x)  
  
## S4 method for signature 'diggit'  
diggitFcnv(x)  
  
## S4 method for signature 'diggit'  
diggitMR(x)  
  
## S4 method for signature 'diggit'  
diggitViper(x)  
  
## S4 method for signature 'diggit'  
diggitAqtl(x)  
  
## S4 method for signature 'diggit'  
diggitConditional(x)  
  
## S4 method for signature 'diggit'  
summary(object)  
  
## S4 method for signature 'diggit'  
head(x, rows = 4, cols = 4)  
  
## S4 method for signature 'diggit'  
mindyFiltering(x, mr = 0.01, mr.adjust = c("none", "fdr",  
      "bonferroni"))
```

**Arguments**

x	Object of class diggit
pval	P-value threshold for the conditional analysis
object	Object of class diggit
rows	Integer indicating the maximum number of rows to show
cols	Integer indicating the maximum number of columns to show

<code>mr</code>	Either a numerical value between 0 and 1 indicating the p-value threshold for the Master Regulator (MR) selection, or a vector of character strings listing the MRs
<code>mr.adjust</code>	Character string indicating the multiple hypothesis test correction for the MRs

**Value**

`print` returns summary information about the diggit object

`show` returns summary information about the object of class diggit

`exprs` returns the ExpressionSet object containing the expression profile data

`diggitCNV` returns a matrix containing the CNV data

`diggitRegulon` returns a regulon object containing the transcriptional interactome

`diggitMindy` returns a regulon object containing the post-translational interactome

`diggitFcv` returns a vector of p-values for the F-CNVs

`diggitMR` returns a vector of master regulators NES

`diggitViper` returns a matrix of VIPER results

`diggitAqtl` returns a matrix of aQTLs (p-value)

`diggitConditional` returns a list containing the conditional analysis results

`summary` returns the integrated results from the conditional analysis

`head` returns a list containing a reduced view for an object of class diggit

`mindyFiltering` returns a diggit class object with CNV and aQTL slots filtered to contain only MINDY post-translational modulators of the MRs

**Examples**

```
data(gbm.expression, package="diggitdata")
data(gbm.cnv, package="diggitdata")
data(gbm.aracne, package="diggitdata")
dobj <- diggitClass(expset=gbmExprs, cnv=gbmCNV, regulon=gbmTFregulon)
print(dobj)
show(dobj)
exprs(dobj)
diggitCNV(dobj)[1:3, 1:3]
diggitRegulon(dobj)
diggitMindy(dobj)
diggitFcv(dobj)
diggitMR(dobj)
diggitViper(dobj)
diggitAqtl(dobj)
diggitConditional(dobj)
head(dobj)
data(gbm.expression, package="diggitdata")
data(gbm.cnv, package="diggitdata")
data(gbm.mindy, package="diggitdata")
dobj <- diggitClass(expset=gbmExprs, cnv=gbmCNV, mindy=gbmMindy)
dobj <- fCNV(dobj)
dobj
dobj <- mindyFiltering(dobj, mr=c("STAT3", "CEBPD"))
dobj
```

# Index

aecdf, 2  
aql, 2  
aql, diggit-method (aql), 2

conditional, 3  
conditional, diggit-method  
    (condition), 3  
correlation, 4

diggit-class, 5  
diggitAql (print, diggit-method), 10  
diggitAql, diggit-method  
    (print, diggit-method), 10  
diggitClass (diggit-class), 5  
diggitCNV (print, diggit-method), 10  
diggitCNV, diggit-method  
    (print, diggit-method), 10  
diggitConditional  
    (print, diggit-method), 10  
diggitConditional, diggit-method  
    (print, diggit-method), 10  
diggitFcv (print, diggit-method), 10  
diggitFcv, diggit-method  
    (print, diggit-method), 10  
diggitMindy (print, diggit-method), 10  
diggitMindy, diggit-method  
    (print, diggit-method), 10  
diggitMR (print, diggit-method), 10  
diggitMR, diggit-method  
    (print, diggit-method), 10  
diggitRegulon (print, diggit-method), 10  
diggitRegulon, diggit-method  
    (print, diggit-method), 10  
diggitViper (print, diggit-method), 10  
diggitViper, diggit-method  
    (print, diggit-method), 10

exprs, diggit-method  
    (print, diggit-method), 10  
exprs.diggit (print, diggit-method), 10

fCNV, 6  
fCNV, data.frame-method (fCNV), 6  
fCNV, diggit-method (fCNV), 6

fCNV, ExpressionSet-method (fCNV), 6  
fCNV, matrix-method (fCNV), 6

head, diggit-method  
    (print, diggit-method), 10  
head.diggit (print, diggit-method), 10

marina, 7  
marina, diggit-method (marina), 7  
marina, ExpressionSet-method (marina), 7  
marina, matrix-method (marina), 7  
mindyFiltering (print, diggit-method), 10  
mindyFiltering, diggit-method  
    (print, diggit-method), 10  
mutualInfo, 9

plot (plot, diggit-method), 10  
plot, diggit-method, 10  
print, diggit-method, 10  
print.diggit (print, diggit-method), 10

show, diggit-method  
    (print, diggit-method), 10  
show.diggit (print, diggit-method), 10  
summary, diggit-method  
    (print, diggit-method), 10  
summary.diggit (print, diggit-method), 10