

# Package ‘GeoDiff’

May 8, 2026

**Type** Package

**Title** Count model based differential expression and normalization on  
GeoMx RNA data

**Version** 1.19.0

**Description** A series of statistical models using count generating distributions  
for background modelling, feature and sample QC, normalization and differential  
expression analysis on GeoMx RNA data. The application of these methods are  
demonstrated by example data analysis vignette.

**Imports** Matrix, robust, plyr, lme4, Rcpp (>= 1.0.4.6), withr, methods,  
graphics, stats, testthat, GeomxTools, NanoStringNCTools

**LinkingTo** Rcpp, RcppArmadillo, roptim

**License** MIT + file LICENSE

**URL** <https://github.com/Nanostring-Biostats/GeoDiff>

**BugReports** <https://github.com/Nanostring-Biostats/GeoDiff>

**Encoding** UTF-8

**Suggests** knitr, rmarkdown, dplyr

**VignetteBuilder** knitr

**Depends** R (>= 4.1.0), Biobase

**RoxygenNote** 7.1.2

**biocViews** GeneExpression, DifferentialExpression, Normalization

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

**git\_branch** devel

**git\_last\_commit** ea81008

**git\_last\_commit\_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-05-07

**Author** Maddy Griswold [cre],  
Lei Yang [aut],  
Zhi Yang [aut]

**Maintainer** Maddy Griswold <[mgriswold@nanostring.com](mailto:mgriswold@nanostring.com)>

## Contents

aggprobe	2
BGScoreTest	3
BGScoreTest_sp	5
coefNBth	6
contrastNBth	6
demoData	7
DENBth	8
diagPoisBG	8
fitNBth	10
fitNBthDE	12
fitNBthmDE	16
fitPoisBG	19
fitPoisBG_sp	21
fitPoisthNorm	22
fitPoisthNorm_sp	25
kidney	27
NBthDEmod2	28
NBthmDEmod2	28
NBthmDEmod2slope	29
QuanRange	29
<b>Index</b>	<b>31</b>

---

aggprobe	<i>Generate aggregated counts of probes for the same target</i>
----------	---

---

### Description

Generate Generate aggregated counts of probes for the same target, based on their score test results or correlation

Generate Generate aggregated counts of probes for the same target, based on their score test results or correlation

### Usage

```
aggprobe(object, ...)
```

```
## S4 method for signature 'NanoStringGeoMxSet'
```

```
aggprobe(
  object,
  split,
  use = c("score", "cor", "both"),
  corcutoff = 0.85,
  ...
)
```

```
## S4 method for signature 'matrix'
```

```
aggprobe(
  object,
  probenames,
```

```

    featurenames,
    negmod,
    use = c("score", "cor", "both"),
    corcutoff = 0.85,
    ...
)

```

### Arguments

object	matrix of probes
...	additional argument list that might be used
split	indicator variable on whether it is for multiple slides (Yes, TRUE; No, FALSE)
use	the method to determine outliers including score, cor, and both
corcutoff	the cutoff value for correlation
probenames	vector of names of probe
featurenames	vector of names of features each probe corresponding to
negmod	Poisson Background model object for negative probes

### Value

- remain, the list of remaining probes of targets
- probenumber, numerical vector of probe numbers of targets
- featuremat, the matrix of features
- remain, the list of remaining probes of targets
- probenumber, numerical vector of probe numbers of targets
- featuremat, the matrix of features

### Examples

```

data("demoData")
demoData <- aggregprobe(demoData, use = "cor")

```

---

 BGScoreTest

*Testing for features above the background*


---

### Description

Testing for features above the background using Poisson background model as reference

Testing for features above the background using Poisson background model as reference

**Usage**

```

BGScoreTest(object, ...)

## S4 method for signature 'NanoStringGeoMxSet'
BGScoreTest(
  object,
  split = FALSE,
  adj = 1,
  removeoutlier = FALSE,
  useprior = FALSE
)

## S4 method for signature 'matrix'
BGScoreTest(
  object,
  BGmod,
  adj = 1,
  probenum,
  removeoutlier = FALSE,
  useprior = FALSE
)

```

**Arguments**

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
split	indicator variable on whether it is for multiple slides (Yes, TRUE; No, FALSE)
adj	adjustment factor for the number of feature in each gene, default =1 i.e. each target only consists of one probe
removeoutlier	whether to remove outlier
useprior	whether to use the prior that the expression level of background follows a Beta distribution, leading to a more conservative test
BGmod	a list of sizefact, sizefact, and countmat
probenum	a vector of numbers of probes in each gene

**Value**

a valid GeoMx S4 object including the following items

- pvalues - Background score test pvalues, in featureData
- scores - Background score test statistics, in featureData

if split is TRUE, a valid GeoMx S4 object including the following items

- pvalues\_XX - Background score test pvalues vector, column name (denoted as XX) the same as slide names, in featureData
- scores\_XX - Background score test statistics vector, column name (denoted as XX) the same as slide names, in featureData

a list of following items

- pvalues - Background score test pvalues
- scores - Background score test statistics

**Examples**

```

data(demoData)
demoData <- fitPoisBG(demoData, size_scale = "sum")
demoData <- aggprobe(demoData, use = "cor")
demoData <- BGScoreTest(demoData, adj = 1, useprior = FALSE)
demoData <- fitPoisBG(demoData, size_scale = "sum", groupvar = "slide name")
demoData <- BGScoreTest(demoData, adj = 1, useprior = TRUE, split = TRUE)

```

---

BGScoreTest\_sp

*Testing for features above the background, multiple slides case*


---

**Description**

Testing for features above the background using Poisson background model as reference, multiple slides case

**Usage**

```

BGScoreTest_sp(object, ...)

## S4 method for signature 'matrix'
BGScoreTest_sp(
  object,
  BGmod,
  adj = 1,
  probenum,
  removeoutlier = FALSE,
  useprior = FALSE
)

```

**Arguments**

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
BGmod	fitted background model, multiple slides case
adj	adjustment factor for the number of probes in each feature, default =1 i.e. each target only consists of one probe
probenum	a vector of numbers of probes in each gene
removeoutlier	whether to remove outlier
useprior	whether to use the prior that the expression level of background follows the Beta distribution, leading to a more conservative test

**Value**

a list of following items

- pvalues - Background score test pvalues matrix, columns the same as slide names
- scores\_sp - Background score test statistics matrix, columns the same as slide names

---

coefNBth	<i>Generate list of Wald test inference results on model coefficients</i>
----------	---

---

**Description**

Generate list of Wald test inference results including parameter estimation and p value

**Usage**

```
coefNBth(object, ...)

## S4 method for signature 'list'
coefNBth(object, fullpara = FALSE)
```

**Arguments**

object	DE model, output by fitNBthDE or fitNBthmDE
...	additional argument list that might be used
fullpara	whether to generate results on all parameters

**Value**

- estimate, coefficients estimate
- wald\_stat, Wald test statistics
- p\_value, p value of Wald test
- se, standard error

**Examples**

```
data(NBthmDEmod2)
coeff <- coefNBth(NBthmDEmod2)
```

---

contrastNBth	<i>Generate list of Wald test inference results on user specified contrasts</i>
--------------	---

---

**Description**

Generate list of Wald test inference results including contrast estimation and p value

**Usage**

```
contrastNBth(object, ...)

## S4 method for signature 'list'
contrastNBth(
  object,
  test = c("two-sided", ">", "<"),
  method = diag(1, ncol(object$X)),
  baseline = rep(0, ncol(method))
)
```

**Arguments**

object	DE model, output by fitNBthDE or fitNBthmDE
...	additional argument list that might be used
test	type of statistical test, choose from c("two-sided", ">", "<")
method	contrasts methods, only matrix of contrast vector is allowed for now, default=diag(1,ncol(object\$X)), i.e. testing the regression coefficients
baseline	testing baseline, default=0.

**Value**

- estimate, contrasts estimate
- wald\_stat, Wald test statistics
- p\_value, p value of Wald test
- se, standard error

**Examples**

```
data(NBthmDEmod2)
coeff <- contrastNBth(NBthmDEmod2)
```

---

demoData

*A demo dataset for GeoMx Cancer Transcriptome Atlas (CTA) panel*

---

**Description**

A demo dataset contains 88 ROIs and 8707 features

**Usage**

```
data(demoData)
```

**Format**

A NanoStringGeoMxSet S4 object with 8707 features and 88 samples

**Examples**

```
data(demoData)
```

---

DENBth	<i>Generate DE table using the inference list generated by coefNBth or contrastNBth</i>
--------	---

---

### Description

Generate DE table using the inference list generated by coefNBth or contrastNBth

### Usage

```
DENBth(object, ...)
```

```
## S4 method for signature 'list'
```

```
DENBth(object, variable, NAto1 = TRUE, padj = TRUE, padj_method = "BH")
```

### Arguments

object	inference list from coefNBth or contrastNBth
...	additional argument list that might be used
variable	needed to construct
NAto1	whether to replace NA in pvalue by 1
padj	whether to adjust p value
padj_method	p value adjustment method, default="BH"

### Value

DEtab, DE table

### Examples

```
data(NBthmDEmod2)
coeff <- coefNBth(NBthmDEmod2)
DEtab <- DENBth(coeff, variable = "regiontubule")
```

---

diagPoisBG	<i>Perform diagnosis on Poisson background model</i>
------------	--

---

### Description

Perform diagnosis on Poisson background model

Perform diagnosis on Poisson background model

**Usage**

```
diagPoisBG(object, ...)

## S4 method for signature 'NanoStringGeoMxSet'
diagPoisBG(
  object,
  split = FALSE,
  padj = FALSE,
  padj_method = "BH",
  cutoff = 1e-06,
  generate_ppplot = TRUE
)

## S4 method for signature 'list'
diagPoisBG(
  object,
  padj = FALSE,
  padj_method = "BH",
  cutoff = 1e-06,
  generate_ppplot = TRUE
)
```

**Arguments**

object	a list of sizefact, featfact, countmat, or id (if it is for mutiple slides)
...	additional argument list that might be used
split	indicator variable on whether it is for multiple slides (Yes, TRUE; No, FALSE)
padj	whether to adjust p value for outlier detection, default =TRUE
padj_method	p value adjustment method, default ="BH"
cutoff	p value(or adjusted p value) cutoff to determine outliers
generate_ppplot	whether to generate ppplot, default =TRUE

**Value**

a valid S4 object

- lowtail - A matrix of lower tail probabily, in assay slot
- uptail - A matrix of upper tail probability, in assay slot
- disper (or disper\_sp if non single-valued groupvar is provided) - dispersion parameter in experimenetData
- low\_outlier - A matrix to indicate lower outliers (0:False, 1:True) in assay slot
- upper\_outlier - A matrix to indicate upper outliers (0:False, 1:True) in assay slot

a list of following items

- lowtail - A matrix of lower tail probabily
- uptail - A matrix of upper tail probability
- disper - dispersion parameter
- outlier - A list of coodinates of lower and upper outliers

**Examples**

```

data(demoData)
demoData <- fitPoisBG(demoData, size_scale = "sum")
demoData <- diagPoisBG(demoData)
Biobase::notes(demoData)$disper
demoData <- fitPoisBG(demoData, groupvar = "slide name")
demoData <- diagPoisBG(demoData, split = TRUE)
Biobase::notes(demoData)$disper_sp

```

fitNBth

*Negative Binomial threshold model***Description**

Estimate the signal size factor for features above the background

Estimate the signal size factor for features above the background

**Usage**

```

fitNBth(object, ...)

## S4 method for signature 'NanoStringGeoMxSet'
fitNBth(
  object,
  split = TRUE,
  features_high = NULL,
  sizefact_BG = NULL,
  sizefact_start = sizefact_BG,
  size_scale = c("sum", "first"),
  threshold_start = NULL,
  threshold_fix = FALSE,
  tol = 1e-07,
  iterations = 8,
  start_para = c(threshold_start, 0.5),
  lower_sizefact = 0,
  lower_threshold = threshold_start/5
)

## S4 method for signature 'matrix'
fitNBth(
  object,
  features_high,
  probenum,
  sizefact_BG,
  sizefact_start = sizefact_BG,
  size_scale = c("sum", "first"),
  threshold_start,
  threshold_fix = FALSE,
  tol = 1e-07,
  iterations = 8,
  start_para = c(threshold_start, 1),

```

```

    lower_sizefact = 0,
    lower_threshold = threshold_start/5
  )

```

### Arguments

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
split	indicator variable on whether it is for multiple slides (Yes, TRUE; No, FALSE)
features_high	subset of features which are well above the background
sizefact_BG	size factors for the background
sizefact_start	initial value for size factors
size_scale	method to scale the sizefact, sum(sizefact)=1 when size_scale="sum", sizefact[1]=1 when size_scale="first"
threshold_start	initial value for threshold
threshold_fix	whether to fix the threshold, default=FALSE
tol	tolerance to determine convergence, default=1e-3
iterations	maximum iterations to be run, default=5
start_para	starting values for parameter estimation, default=c(threshold_start, 1)
lower_sizefact	lower limit for sizefact, default=0
lower_threshold	lower limit for threshold
probenum	a vector of numbers of probes in each gene

### Value

a valid GeoMx S4 object

- para0 = "NA", in experimentData
- para, estimated parameters, "signal" "r" in rows and features in columns, in featureData
- sizefact, estimated size factor, in phenoData
- preci1 = "NA", in experimentData
- conv0 = "NA", in experimentData
- conv = "NA", in experimentData
- Im = "NA", in experimentData
- features\_high, a vector of indicators, in featureData (0: No; 1: Yes; NA: not included in features\_high)
- features\_all = "NA", in experimentData
- threshold, estimated threshold, when threshold\_fix, equals to threshold\_start, in experimentData

a list of following items, some items are place holders = NA

- para0 = NA,
- para, estimated parameters, "signal" "r" in rows and features in columns
- sizefact, estimated size factor

- `prec1 = NA`
- `conv0 = NA`
- `conv = NA`
- `Im = NA`
- `features_high = features_high`
- `features_all = NA`
- `threshold`, estimated threshold, when `threshold_fix`, equals to `threshold_start`

### Examples

```
library(Biobase)
library(dplyr)
data(demoData)
demoData <- fitPoisBG(demoData, size_scale = "sum")
demoData <- aggprobe(demoData, use = "cor")
demoData <- BGScoreTest(demoData)
thmean <- 1 * mean(fData(demoData)$featfact, na.rm = TRUE)
demo_pos <- demoData[which(!fData(demoData)$CodeClass == "Negative"), ]
demo_neg <- demoData[which(fData(demoData)$CodeClass == "Negative"), ]
sc1_scores <- fData(demo_pos)[, "scores"]
names(sc1_scores) <- fData(demo_pos)[, "TargetName"]
features_high <- ((sc1_scores > quantile(sc1_scores, probs = 0.4)) &
  (sc1_scores < quantile(sc1_scores, probs = 0.95))) |>
  which() |>
  names()
set.seed(123)
features_high <- sample(features_high, 100)
demoData <- fitNBth(demoData,
  features_high = features_high,
  sizefact_BG = demo_neg$sizefact,
  threshold_start = thmean,
  iterations = 5,
  start_para = c(200, 1),
  lower_sizefact = 0,
  lower_threshold = 100,
  tol = 1e-8)
```

---

fitNBthDE

*Negative Binomial threshold model for differential expression analysis*

---

### Description

Negative Binomial threshold model for differential expression analysis

Negative Binomial threshold model for differential expression analysis

### Usage

```
fitNBthDE(object, ...)
```

```
## S4 method for signature 'NanoStringGeoMxSet'
```

```

fitNBthDE(
  object,
  form,
  split,
  ROIs_high = NULL,
  features_high = NULL,
  features_all = NULL,
  sizefact_start = NULL,
  sizefact_BG = NULL,
  threshold_mean = NULL,
  preci2 = 10000,
  lower_threshold = 0.01,
  prior_type = c("contrast", "equal"),
  sizefactrec = TRUE,
  size_scale = c("sum", "first"),
  sizescalebythreshold = FALSE,
  iterations = 2,
  covrob = FALSE,
  preci1con = 1/25,
  cutoff = 10,
  confac = 1
)

```

```

## S4 method for signature 'matrix'
fitNBthDE(
  form,
  annot,
  object,
  probenum,
  features_high,
  features_all,
  sizefact_start,
  sizefact_BG,
  threshold_mean,
  preci2 = 10000,
  lower_threshold = 0.01,
  prior_type = c("contrast", "equal"),
  sizefactrec = TRUE,
  size_scale = c("sum", "first"),
  sizescalebythreshold = FALSE,
  iterations = 2,
  covrob = FALSE,
  preci1con = 1/25,
  cutoff = 10,
  confac = 1
)

```

### Arguments

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
form	model formula

split	indicator variable on whether it is for multiple slides (Yes, TRUE; No, FALSE)
ROIs_high	ROIs with high expressions defined based on featfact and featfact
features_high	subset of features which are well above the background
features_all	full list of features
sizefact_start	initial value for size factors
sizefact_BG	size factor for background
threshold_mean	average threshold level
preci2	precision for the background, default=10000
lower_threshold	lower limit for the threshold, default=0.01
prior_type	empirical bayes prior type, choose from c("contrast", "equal")
sizefactrec	whether to recalculate sizefact, default=TRUE
size_scale	method to scale the sizefact, sum(sizefact)=1 when size_scale="sum", sizefact[1]=1 when size_scale="first"
sizescalebythreshold	XXXX, default = FALSE
iterations	how many iterations need to run to get final results, default=2, the first iteration apply the model only on features_high and construct the prior then refit the model using this prior for all genes.
covrob	whether to use robust covariance in calculating covariance. default=FALSE
preci1con	The user input constant term in specifying precision matrix 1, default=1/25
cutoff	term in calculating precision matrix 1, default=10
confac	The user input factor for contrast in precision matrix 1, default=1
annot	annotations files with variables in the formula
probenum	a vector of numbers of probes in each gene, default = rep(1, NROW(object))

### Value

a list of

- X, design matrix
- para0, estimated parameters for the first iteration, including regression coefficients, r and threshold in rows and features in columns
- para, estimated parameters, including regression coefficients, r and threshold in rows and features in columns
- sizefact, estimated sizefact
- sizefact0, estimated sizefact in iter=1
- preci1, precision matrix for regression coefficients estimated in iter=1
- Im0, Information matrix of parameters in iter=1
- Im, Information matrix of parameters in iter=2
- conv0, vector of convergence for iter=1, 0 converged, 1 not converged
- conv, vector of convergence for iter=2, 0 converged, 1 not converged
- features\_high, same as the input features\_high
- features\_all, same as the input features\_all

a list of

- X, design matrix
- para0, estimated parameters for the first iteration, including regression coefficients, r and threshold in rows and features in columns
- para, estimated parameters, including regression coefficients, r and threshold in rows and features in columns
- sizefact, estimated sizefact
- sizefact0, estimated sizefact in iter=1
- preci1, precision matrix for regression coefficients estimated in iter=1
- Im0, Information matrix of parameters in iter=1
- Im, Information matrix of parameters in iter=2
- conv0, vector of convergence for iter=1, 0 converged, 1 not converged
- conv, vector of convergence for iter=2, 0 converged, 1 not converged
- features\_high, same as the input features\_high
- features\_all, same as the input features\_all

## Examples

```

library(Biobase)
library(dplyr)
data(demoData)
demoData <- demoData[, c(1:5, 33:37)]
demoData <- fitPoisBG(demoData, size_scale = "sum")
demoData <- aggregprobe(demoData, use = "cor")
demoData <- BGScoreTest(demoData)
demoData$slidename <- substr(demoData[["slide name"]], 12, 17)
thmean <- 1 * mean(fData(demoData)$featfact, na.rm = TRUE)
demo_pos <- demoData[which(!fData(demoData)$CodeClass == "Negative"), ]
demo_neg <- demoData[which(fData(demoData)$CodeClass == "Negative"), ]
sc1_scores <- fData(demo_pos)[, "scores"]
names(sc1_scores) <- fData(demo_pos)[, "TargetName"]
features_high <- ((sc1_scores > quantile(sc1_scores, probs = 0.4)) &
  (sc1_scores < quantile(sc1_scores, probs = 0.95))) |>
  which() |>
  names()
set.seed(123)
demoData <- fitNBth(demoData,
  features_high = features_high,
  sizefact_BG = demo_neg$sizefact,
  threshold_start = thmean,
  iterations = 5,
  start_para = c(200, 1),
  lower_sizefact = 0,
  lower_threshold = 100,
  tol = 1e-8)
ROIs_high <- sampleNames(demoData)[which(demoData$sizefact_fitNBth * thmean > 2)]
features_all <- rownames(demo_pos)

pData(demoData)$group <- c(rep(1, 5), rep(2, 5))

NBthDEmod1 <- fitNBthDE(
  form = ~group,

```

```

split = FALSE,
object = demoData,
ROIs_high = ROIs_high,
features_high = features_high,
features_all = features_all,
sizefact_start = demoData[, ROIs_high][["sizefact_fitNBth"]],
sizefact_BG = demoData[, ROIs_high][["sizefact"]],
preci2 = 10000,
prior_type = "contrast",
covrob = FALSE,
precilcon = 1/25,
scalescalebythreshold = TRUE
)

```

---

fitNBthmDE

*Negative Binomial threshold mixed model for differential expression analysis*


---

### Description

Negative Binomial threshold mixed model for differential expression analysis

Negative Binomial threshold mixed model for differential expression analysis

### Usage

```

fitNBthmDE(object, ...)

## S4 method for signature 'NanoStringGeoMxSet'
fitNBthmDE(
  object,
  form,
  split,
  ROIs_high = NULL,
  features_all = NULL,
  sizefact = NULL,
  sizefact_BG = NULL,
  precil,
  threshold_mean = NULL,
  preci2 = 10000,
  scalescalebythreshold = TRUE,
  controlRandom = list()
)

## S4 method for signature 'matrix'
fitNBthmDE(
  form,
  annot,
  object,
  probenum = rep(1, NROW(object)),
  features_all,

```

```

    sizefact,
    sizefact_BG,
    preci1,
    threshold_mean = NULL,
    preci2 = 10000,
    sizescalebythreshold = TRUE,
    controlRandom = list()
)

```

### Arguments

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
form	model formula
split	indicator variable on whether it is for multiple slides (Yes, TRUE; No, FALSE)
ROIs_high	ROIs with high expressions defined based on featfact and featfact
features_all	vector of all features to be run
sizefact	size factor
sizefact_BG	size factor for background
preci1	precision matrix for regression coefficients
threshold_mean	average background level
preci2	precision for the background, default=10000
sizescalebythreshold	whether to scale the size factor, default=TRUE
controlRandom	list of random effect control parameters
annot	annotations files with variables in the formula
probenum	a vector of numbers of probes in each gene, default = rep(1, NROW(object))

### Value

a list with parameter estimation #'

- X, design matrix for fixed effect
- Z, design matrix for random effect
- rt, random effect terms
- para0, =NA
- para, estimated parameters, including regression coefficients, r and threshold in rows and features in columns
- sizefact, same as input sizefact
- sizefact0, NA
- preci1, input precision matrix for regression coefficients
- Im0, NA
- Im, Information matrix of parameters
- conv0, NA
- conv, vector of convergence, 0 converged, 1 not converged

- features\_high, NA
- features\_all, same as the input features\_all
- theta, list of estimated random effect parameters
- MAP random effect

a list with parameter estimation #'

- X, design matrix for fixed effect
- Z, design matrix for random effect
- rt, random effect terms
- para0, =NA
- para, estimated parameters, including regression coefficients, r and threshold in rows and features in columns
- sizefact, same as input sizefact
- sizefact0, NA
- preci1, input precision matrix for regression coefficients
- Im0, NA
- Im, Information matrix of parameters
- conv0, NA
- conv, vector of convergence, 0 converged, 1 not converged
- features\_high, NA
- features\_all, same as the input features\_all
- theta, list of estimated random effect parameters(for relative covariance matrix)
- varcov, list of estimated variance covariance parameter estimation
- MAP random effect

## Examples

```
library(Biobase)
library(dplyr)
data(demoData)
demoData <- demoData[, c(1:5, 33:37)]
demoData <- fitPoisBG(demoData, size_scale = "sum")
demoData <- aggreprobe(demoData, use = "cor")
demoData <- BGScoreTest(demoData)
demoData$slidename <- substr(demoData[["slide name"]], 12, 17)
thmean <- 1 * mean(fData(demoData)$featfact, na.rm = TRUE)
demo_pos <- demoData[which(!fData(demoData)$CodeClass == "Negative"), ]
demo_neg <- demoData[which(fData(demoData)$CodeClass == "Negative"), ]
sc1_scores <- fData(demo_pos)[, "scores"]
names(sc1_scores) <- fData(demo_pos)[, "TargetName"]
features_high <- ((sc1_scores > quantile(sc1_scores, probs = 0.4)) &
  (sc1_scores < quantile(sc1_scores, probs = 0.95))) |>
  which() |>
  names()
set.seed(123)
demoData <- fitNBth(demoData,
  features_high = features_high,
  sizefact_BG = demo_neg$sizefact,
```

```

        threshold_start = thmean,
        iterations = 5,
        start_para = c(200, 1),
        lower_sizefact = 0,
        lower_threshold = 100,
        tol = 1e-8)
ROIs_high <- sampleNames(demoData)[which(demoData$sizefact_fitNBth * thmean > 2)]
features_all <- rownames(demo_pos)

pData(demoData)$group <- c(rep(1, 5), rep(2, 5))

NBthDEmod2 <- fitNBthDE(form = ~group,
                        split = FALSE,
                        object = demoData,
                        ROIs_high = ROIs_high,
                        features_high = features_high,
                        features_all = features_all,
                        sizefact_start = demoData[, ROIs_high][['sizefact_fitNBth']],
                        sizefact_BG = demoData[, ROIs_high][['sizefact']],
                        threshold_mean = notes(demoData)[["threshold"]],
                        preci2=10000,
                        prior_type="contrast",
                        covrob=FALSE,
                        preci1con=1/25,
                        sizescalebythreshold=TRUE)

set.seed(123)
NBthmDEmod1 <- fitNBthmDE(
  form = ~ group + (1 | `slide name`),
  split = FALSE,
  object = demoData,
  ROIs_high = ROIs_high,
  features_all = features_all[1:5],
  sizefact = demoData[, ROIs_high][["sizefact_fitNBth"]],
  sizefact_BG = demoData[, ROIs_high][["sizefact"]],
  preci1=NBthDEmod2$preci1,
  threshold_mean = thmean,
  preci2=10000,
  sizescale = TRUE,
  controlRandom=list(nu=12, nmh_e=400, thin_e=60))

```

---

fitPoisBG

*Estimate Poisson background model for either single slide or multiple slides*


---

## Description

Estimate Poisson background model for either single slide or multiple slides

Estimate Poisson background model:

**Usage**

```
fitPoisBG(object, ...)

## S4 method for signature 'NanoStringGeoMxSet'
fitPoisBG(
  object,
  groupvar = NULL,
  iterations = 10,
  tol = 0.001,
  size_scale = c("sum", "first"),
  ...
)

## S4 method for signature 'matrix'
fitPoisBG(object, iterations = 10, tol = 0.001, size_scale = c("sum", "first"))
```

**Arguments**

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
groupvar	the group variable name for slide
iterations	maximum iterations to be run, default=10
tol	tolerance to determine convergence, default = 1e-3
size_scale	method to scale the sizefact, sum(sizefact)=1 when size_scale="sum", sizefact[1]=1 when size_scale="first"

**Value**

a valid GeoMx S4 object if split is FALSE

- sizefact - estimated size factor in phenoData
- featfact - estimated feature factor in featureData

a valid GeoMx S4 object if split is TRUE,

- sizefact - estimated size factor in phenoData
- featfact\_XX - estimated feature factor vector, column name (denoted as XX) the same as the slide id, in featureData for each unique slide
- fitPoisBG\_sp\_var - the column name for slide, in experimentData

a list of following items

- sizefact - estimated size factor
- featfact - estimated feature factor
- countmat - the input count matrix

**Examples**

```
data(demoData)
demoData <- fitPoisBG(demoData, size_scale = "sum")
data(demoData)
demoData <- fitPoisBG(demoData, groupvar = "slide name", size_scale = "sum")
```

---

`fitPoisBG_sp`*Estimate Poisson background model for multiple slides*

---

### Description

Estimate Poisson background model for multiple slides:

### Usage

```
fitPoisBG_sp(object, ...)  
  
## S4 method for signature 'matrix'  
fitPoisBG_sp(  
  object,  
  id,  
  iterations = 10,  
  tol = 0.001,  
  size_scale = c("sum", "first")  
)
```

### Arguments

<code>object</code>	count matrix with features in rows and samples in columns
<code>...</code>	additional argument list that might be used
<code>id</code>	character vector same size as sample size representing slide names of each sample
<code>iterations</code>	maximum iterations to be run, default=10
<code>tol</code>	tolerance to determine convergence, default = 1e-3
<code>size_scale</code>	method to scale the sizefact, sum(sizefact)=1 when size_scale="sum", sizefact[1]=1 when size_scale="first"

### Value

a list of following items

- sizefact - estimated size factor
- featfact - estimated feature factor matrix, column names the same as the slide id
- countmat - the input count matrix
- id - the input id

---

fitPoisthNorm	<i>Poisson threshold model based normalization-log2 transformation for single slide or for multiple slides</i>
---------------	--

---

### Description

Poisson threshold model based normalization-log2 transformation for single slide or for multiple slides

### Usage

```
fitPoisthNorm(object, ...)

## S4 method for signature 'NanoStringGeoMxSet'
fitPoisthNorm(
  object,
  split = FALSE,
  ROIs_high = NULL,
  features_high = NULL,
  features_all = NULL,
  sizefact_start = NULL,
  sizefact_BG = NULL,
  threshold_mean = NULL,
  preci2 = 10000,
  iterations = 2,
  prior_type = c("contrast", "equal"),
  sizefactrec = TRUE,
  size_scale = c("sum", "first"),
  sizescalebythreshold = FALSE,
  covrob = FALSE,
  precilcon = 1/25,
  cutoff = 15,
  confac = 1,
  calhes = FALSE
)

## S4 method for signature 'matrix'
fitPoisthNorm(
  object,
  probenum = rep(1, NROW(object)),
  features_high,
  features_all,
  sizefact_start,
  sizefact_BG,
  threshold_mean,
  preci2 = 10000,
  iterations = 2,
  prior_type = c("contrast", "equal"),
  sizefactrec = TRUE,
  size_scale = c("sum", "first"),
  sizescalebythreshold = FALSE,
```

```

covrob = FALSE,
prec1con = 1/25,
cutoff = 15,
confac = 1,
calhes = FALSE
)

```

### Arguments

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
split	indicator variable on whether it is for multiple slides (Yes, TRUE; No, FALSE)
ROIs_high	ROIs with high expressions defined based on featfact and featfact
features_high	subset of features which are well above the background
features_all	full feature vector to apply the normalization on
sizefact_start	initial value for size factors
sizefact_BG	size factor for background
threshold_mean	average threshold level
preci2	precision for threshold, default=10000
iterations	iteration number, default=2, the first iteration using the features_high to construct the prior for parameters then refit the model on all features. precision matrix for threshold: preci2
prior_type	prior type for preci1, "equal" or "contrast", default="contrast"
sizefactrec	XXXX, default = TRUE
size_scale	method to scale the sizefact, sum(sizefact)=1 when size_scale="sum", sizefact[1]=1 when size_scale="first"
sizescalebythreshold	XXXX, default = FALSE
covrob	whether to use robust covariance in calculating the prior precision matrix 1, default = FALSE
prec1con	The user input constant term in specifying precision matrix 1, default=1/25
cutoff	term in calculating precision matrix 1, default=15
confac	The user input factor for contrast in precision matrix 1, default=1
calhes	The user input whether to calculate hessian: calhes, default=FALSE
probenum	a vector of numbers of probes in each gene

### Value

if split is FALSE, a valid GeoMx S4 object including the following items

- para0\_norm, matrix of estimated parameters for iter=1, features in columns and parameters(log2 expression, threshold) in rows, in featureData.
- para\_norm, matrix of estimated parameters for iter=2, features in columns and parameters(log2 expression, threshold) in rows, in featureData.
- normmat0, matrix of log2 expression for iter=1, features in columns and log2 expression in rows, in assay slot.

- normmat, matrix of log2 expression for iter=2, features in columns and log2 expression in rows, in assay lot.
- sizefact\_norm, estimated sizefact, in phenoData.
- sizefact0\_norm, estimated sizefact in iter=1, in phenoData.
- preci1, precision matrix 1, in experimentData.
- conv0, vector of convergence for iter=1, 0 converged, 1 not converged, in featureData
- conv, vector of convergence for iter=2, 0 converged, 1 not converged, in featureData
- features\_high, same as the input features\_high, in featureData
- features\_all, same as the input features\_all, in featureData

if split is TRUE, a valid GeoMx S4 object with the following items appended.

- threshold0, matrix of estimated threshold for iter=1, features in columns and threshold for different slides in rows, in featureData.
- threshold, matrix of estimated threshold for iter=2, features in columns and threshold for different slides in rows, in featureData.
- normmat0\_sp, matrix of log2 expression for iter=1, features in columns and log2 expression in rows, in assay slot.
- normmat\_sp, matrix of log2 expression for iter=2, features in columns and log2 expression in rows, in assay slot.
- sizefact\_norm\_sp, estimated sizefact, in phenoData
- sizefact0\_norm\_sp, estimated sizefact in iter=1, in phenoData
- preci1, precision matrix 1, in experimentData
- conv0\_sp\_XX, vector of convergence for each unique slide value for iter=1, 0 converged, 1 not converged, in featureData for each unique slide.
- conv\_sp\_XX, vector of convergence for each unique slide value for iter=2, 0 converged, 1 not converged, in featureData for each unique slide.
- features\_high\_sp, same as the input features\_high, in featureData.
- features\_all\_sp, same as the input features\_all, in featureData.

a list of following items

- para0, matrix of estimated parameters for iter=1, features in columns and parameters(log2 expression, threshold) in rows.
- para, matrix of estimated parameters for iter=2, features in columns and parameters(log2 expression, threshold) in rows.
- normmat0, matrix of log2 expression for iter=1, features in columns and log2 expression in rows.
- normmat, matrix of log2 expression for iter=2, features in columns and log2 expression in rows.
- sizefact, estimated sizefact
- sizefact0, estimated sizefact in iter=1
- preci1, precision matrix 1
- Im0, Information matrix of parameters in iter=1
- Im, Information matrix of parameters in iter=2
- conv0, vector of convergence for iter=1, 0 converged, 1 not converged
- conv, vector of convergence for iter=2, 0 converged, 1 not converged
- features\_high, same as the input features\_high
- features\_all, same as the input features\_all

**Examples**

```

library(Biobase)
library(dplyr)
data(demoData)
demoData <- fitPoisBG(demoData, size_scale = "sum")
demoData <- aggprobe(demoData, use = "cor")
demoData <- BGScoreTest(demoData)
thmean <- 1 * mean(fData(demoData)$featfact, na.rm = TRUE)
demo_pos <- demoData[which(!fData(demoData)$CodeClass == "Negative"), ]
demo_neg <- demoData[which(fData(demoData)$CodeClass == "Negative"), ]
sc1_scores <- fData(demo_pos)[, "scores"]
names(sc1_scores) <- fData(demo_pos)[, "TargetName"]
features_high <- ((sc1_scores > quantile(sc1_scores, probs = 0.4)) &
  (sc1_scores < quantile(sc1_scores, probs = 0.95))) |>
  which() |>
  names()
set.seed(123)
features_high <- sample(features_high, 100)
demoData <- fitNBth(demoData,
  features_high = features_high,
  sizefact_BG = demo_neg$sizefact,
  threshold_start = thmean,
  iterations = 5,
  start_para = c(200, 1),
  lower_sizefact = 0,
  lower_threshold = 100,
  tol = 1e-8)
ROIs_high <- sampleNames(demoData)[which((quantile(fData(demoData)[["para"]][, 1],
  probs = 0.90, na.rm = TRUE) -
  notes(demoData)[["threshold"]]) * demoData$sizefact_fitNBth > 2)]
features_all <- rownames(demo_pos)
thmean <- mean(fData(demo_neg)[["featfact"]])
demoData <- fitPoisthNorm(
  object = demoData,
  split = FALSE,
  ROIs_high = ROIs_high,
  features_high = features_high,
  features_all = features_all,
  sizefact_start = demoData[, ROIs_high][["sizefact_fitNBth"]],
  sizefact_BG = demoData[, ROIs_high][["sizefact"]],
  threshold_mean = thmean,
  preci2 = 10000,
  prior_type = "contrast",
  covrob = FALSE,
  prec1con = 1 / 25
)

```

fitPoisthNorm\_sp

*Poisson threshold model based normalization-log2 transformation for multiple slides***Description**

Poisson threshold model based normalization-log2 transformation for multiple slides

**Usage**

```
fitPoisthNorm_sp(object, ...)

## S4 method for signature 'matrix'
fitPoisthNorm_sp(
  object,
  probenum,
  features_high,
  features_all = colnames(object),
  sizefact_start,
  sizefact_BG,
  threshold_mean,
  preci2 = 10000,
  id,
  iterations = 2,
  prior_type = c("contrast", "equal"),
  sizefactrec = TRUE,
  size_scale = c("sum", "first"),
  sizescalebythreshold = FALSE,
  covrob = FALSE,
  preci1con = 1/25,
  cutoff = 15,
  confac = 1
)
```

**Arguments**

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
probenum	a vector of numbers of probes in each gene
features_high	subset of features which are well above the background
features_all	full feature vector to apply the normalization on
sizefact_start	initial value for size factors
sizefact_BG	size factor for background
threshold_mean	average threshold level
preci2	precision for threshold, default=10000
id	character vector of slide name of each sample
iterations	iteration number, default=2, the first iteration using the features_high to construct the prior for parameters then refit the model on all features. precision matrix for threshold: preci2
prior_type	prior type for preci1, "equal" or "contrast", default="contrast"
sizefactrec	XXXX, default = TRUE
size_scale	method to scale the sizefact, sum(sizefact)=1 when size_scale="sum", sizefact[1]=1 when size_scale="first"
sizescalebythreshold	XXXX, default = FALSE
covrob	whether to use robust covariance in calculating the prior precision matrix 1, default = FALSE

preci1con	The user input constant term in specifying precision matrix 1, default=1/25
cutoff	term in calculating precision matrix 1, default=15
confac	The user input factor for contrast in precision matrix 1, default=1

**Value**

a list of following items

- threshold0, matrix of estimated threshold for iter=1, features in columns and threshold for different slides in rows.
- threshold, matrix of estimated threshold for iter=2, features in columns and threshold for different slides in rows.
- normmat0, matrix of log2 expression for iter=1, features in columns and log2 expression in rows.
- normmat, matrix of log2 expression for iter=2, features in columns and log2 expression in rows.
- sizefact, estimated sizefact
- sizefact0, estimated sizefact in iter=1
- preci1, precision matrix 1
- Im0, Information matrix in iter=1
- Im, Information matrix in iter=2
- conv0, vector of convergence for iter=1, 0 converged, 1 not converged
- conv, vector of convergence for iter=2, 0 converged, 1 not converged
- features\_high, same as the input features\_high
- features\_all, same as the input features\_all

---

kidney	<i>A demo dataset for GeoMx Human Whole Transcriptome Atlas (WTA) panel</i>
--------	---

---

**Description**

A demo dataset contains 276 ROIs and 18642 features

**Usage**

```
data(kidney)
```

**Format**

A NanoStringGeoMxSet S4 object with 18642 features and 276 samples

**Examples**

```
data(kidney)
```

---

NBthDEmod2

*A demo example output list returned by function fitNBthDE*

---

**Description**

A list used to demonstrate the function coefNBth

**Usage**

```
data(NBthDEmod2)
```

**Format**

A list

**Examples**

```
data(NBthDEmod2)
```

---

NBthmDEmod2

*A demo example output list returned by function fitNBthmDE*

---

**Description**

A list used to demonstrate the function coefNBth

**Usage**

```
data(NBthmDEmod2)
```

**Format**

A list

**Examples**

```
data(NBthmDEmod2)
```

---

NBthmDEmod2slope      *A demo example output list returned by function fitNBthmDE*

---

**Description**

A list used to demonstrate the function coefNBth

**Usage**

```
data(NBthmDEmod2slope)
```

**Format**

A list

**Examples**

```
data(NBthmDEmod2slope)
```

---

QuanRange      *Compute Quantile Range*

---

**Description**

Compute Quantile Range, a metric representing signal strength for QC purpose

Compute Quantile Range, a metric representing signal strength for QC purpose

**Usage**

```
QuanRange(object, ...)
```

```
## S4 method for signature 'NanoStringGeoMxSet'
```

```
QuanRange(object, split = FALSE, probs, removeoutlier = FALSE, ...)
```

```
## S4 method for signature 'matrix'
```

```
QuanRange(object, probenum, BGmod, probs, removeoutlier = FALSE)
```

**Arguments**

object	count matrix with features in rows and samples in columns
...	additional argument list that might be used
split	indicator variable on whether it is for multiple slides
probs	numeric vector of probabilities with values in [0,1] passed to quantile
removeoutlier	indicator on whether to remove outliers, default: FALSE
probenum	a vector of numbers of probes in each gene
BGmod	a list of sizefact, sizefact, countmat, and id (if it is for multiple slides)

**Value**

a valid S4 object with probabilities in phenoData  
a matrix of quantile range in rows and probs in columns

**Examples**

```
data(demoData)
demoData <- fitPoisBG(demoData, size_scale = "sum")
demoData <- diagPoisBG(demoData)
demoData <- aggreprobe(demoData, use = "cor")
Biobase::notes(demoData)$disper
demoData <- QuanRange(demoData, split = FALSE, probs = c(0.75, 0.8, 0.9, 0.95))
```

```
data(demoData)
demoData <- fitPoisBG(demoData, groupvar = "slide name")
demoData <- diagPoisBG(demoData, split = TRUE)
demoData <- aggreprobe(demoData, use = "cor")
Biobase::notes(demoData)$disper_sp
demoData <- QuanRange(demoData, split = TRUE, probs = c(0.75, 0.8, 0.9, 0.95))
```

# Index

## \* datasets

- demoData, 7
- kidney, 27
- NBthDEmod2, 28
- NBthmDEmod2, 28
- NBthmDEmod2slope, 29
  
- aggreprobe, 2
- aggreprobe, matrix-method (aggreprobe), 2
- aggreprobe, NanoStringGeoMxSet-method (aggreprobe), 2
  
- BGScoreTest, 3
- BGScoreTest, matrix-method (BGScoreTest), 3
- BGScoreTest, NanoStringGeoMxSet-method (BGScoreTest), 3
- BGScoreTest\_sp, 5
- BGScoreTest\_sp, matrix-method (BGScoreTest\_sp), 5
  
- coefNBth, 6
- coefNBth, list-method (coefNBth), 6
- contrastNBth, 6
- contrastNBth, list-method (contrastNBth), 6
  
- demoData, 7
- DENBth, 8
- DENBth, list-method (DENBth), 8
- diagPoisBG, 8
- diagPoisBG, list-method (diagPoisBG), 8
- diagPoisBG, NanoStringGeoMxSet-method (diagPoisBG), 8
  
- fitNBth, 10
- fitNBth, matrix-method (fitNBth), 10
- fitNBth, NanoStringGeoMxSet-method (fitNBth), 10
- fitNBthDE, 12
- fitNBthDE, matrix-method (fitNBthDE), 12
- fitNBthDE, NanoStringGeoMxSet-method (fitNBthDE), 12
- fitNBthmDE, 16
- fitNBthmDE, matrix-method (fitNBthmDE), 16
- fitNBthmDE, NanoStringGeoMxSet-method (fitNBthmDE), 16
- fitPoisBG, 19
- fitPoisBG, matrix-method (fitPoisBG), 19
- fitPoisBG, NanoStringGeoMxSet-method (fitPoisBG), 19
- fitPoisBG\_sp, 21
- fitPoisBG\_sp, matrix-method (fitPoisBG\_sp), 21
- fitPoisthNorm, 22
- fitPoisthNorm, matrix-method (fitPoisthNorm), 22
- fitPoisthNorm, NanoStringGeoMxSet-method (fitPoisthNorm), 22
- fitPoisthNorm\_sp, 25
- fitPoisthNorm\_sp, matrix-method (fitPoisthNorm\_sp), 25
  
- kidney, 27
  
- NBthDEmod2, 28
- NBthmDEmod2, 28
- NBthmDEmod2slope, 29
  
- QuanRange, 29
- QuanRange, matrix-method (QuanRange), 29
- QuanRange, NanoStringGeoMxSet-method (QuanRange), 29