

# Package ‘CytoGLMM’

March 13, 2025

**Type** Package

**Title** Conditional Differential Analysis for Flow and Mass Cytometry Experiments

**Version** 1.15.0

**Description** The CytoGLMM R package implements two multiple regression strategies: A bootstrapped generalized linear model (GLM) and a generalized linear mixed model (GLMM). Most current data analysis tools compare expressions across many computationally discovered cell types. CytoGLMM focuses on just one cell type. Our narrower field of application allows us to define a more specific statistical model with easier to control statistical guarantees. As a result, CytoGLMM finds differential proteins in flow and mass cytometry data while reducing biases arising from marker correlations and safeguarding against false discoveries induced by patient heterogeneity.

**License** LGPL-3

**URL** <https://christofseiler.github.io/CytoGLMM>,  
<https://github.com/ChristofSeiler/CytoGLMM>

**BugReports** <https://github.com/ChristofSeiler/CytoGLMM/issues>

**Encoding** UTF-8

**LazyData** true

**Imports** stats, methods, BiocParallel, RColorBrewer, cowplot, doParallel, dplyr, factoextra, flexmix, ggplot2, magrittr, mbest, pheatmap, stringr, strucchange, tibble, ggrepel, MASS, logging, Matrix, tidyr, caret, rlang, grDevices

**Suggests** knitr, rmarkdown, testthat, BiocStyle

**VignetteBuilder** knitr

**RoxygenNote** 7.2.3

**biocViews** FlowCytometry, Proteomics, SingleCell, CellBasedAssays, CellBiology, ImmunoOncology, Regression, StatisticalMethod, Software

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

**git\_branch** devel

**git\_last\_commit** a3b4bb6

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

**Repository** Bioconductor 3.21

**Date/Publication** 2025-03-13

**Author** Christof Seiler [aut, cre] (ORCID:  
<<https://orcid.org/0000-0001-8802-3642>>)

**Maintainer** Christof Seiler <christof.seiler@maastrichtuniversity.nl>

## Contents

cytoflexmix . . . . .	3
cytoglm . . . . .	4
cytoglmm . . . . .	6
cytogroup . . . . .	7
cytostab . . . . .	8
cyto_check . . . . .	9
generate_data . . . . .	10
glmm_moment . . . . .	10
is_unpaired . . . . .	11
plot.cytoflexmix . . . . .	12
plot.cytoglm . . . . .	12
plot.cytoglmm . . . . .	13
plot.cytogroup . . . . .	14
plot_coeff . . . . .	15
plot_heatmap . . . . .	15
plot_lda . . . . .	16
plot_mds . . . . .	17
plot_model_selection . . . . .	18
plot_prcomp . . . . .	19
print.cytoglm . . . . .	20
print.cytoglmm . . . . .	21
remove_samples . . . . .	21
summary.cytoglm . . . . .	22
summary.cytoglmm . . . . .	23

**Index**

**24**

---

cytoflexmix                      *Logistic mixture regression*

---

## Description

Logistic mixture regression

## Usage

```
cytoflexmix(
  df_samples_subset,
  protein_names,
  condition,
  group = "donor",
  cell_n_min = Inf,
  cell_n_subsample = 0,
  ks = seq_len(10),
  num_cores = 1
)
```

## Arguments

<code>df_samples_subset</code>	Data frame or tibble with proteins counts, cell condition, and group information
<code>protein_names</code>	A vector of column names of protein to use in the analysis
<code>condition</code>	The column name of the condition variable
<code>group</code>	The column name of the group variable
<code>cell_n_min</code>	Remove samples that are below this cell counts threshold
<code>cell_n_subsample</code>	Subsample samples to have this maximum cell count
<code>ks</code>	A vector of cluster sizes
<code>num_cores</code>	Number of computing cores

## Value

A list of class `cytoglm` containing

<code>flexmixfits</code>	list of <code>flexmix</code> objects
<code>df_samples_subset</code>	possibly subsampled <code>df_samples_subset</code> table
<code>protein_names</code>	input protein names
<code>condition</code>	input condition variable
<code>group</code>	input group names
<code>cell_n_min</code>	input <code>cell_n_min</code>

```

cell_n_subsample      input cell_n_subsample
ks                    input ks
num_cores             input num_cores

```

### Examples

```

set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
mix_fit <- CytoGLMM::cytoflexmix(df,
                                protein_names = protein_names,
                                condition = "condition",
                                group = "donor",
                                ks = 2)

mix_fit

```

---

cytoglm

*Fit GLM with bootstrap resampling*

---

### Description

Fit GLM with bootstrap resampling

### Usage

```

cytoglm(
  df_samples_subset,
  protein_names,
  condition,
  group = "donor",
  covariate_names = NULL,
  cell_n_min = Inf,
  cell_n_subsample = 0,
  num_boot = 100,
  num_cores = 1
)

```

### Arguments

<code>df_samples_subset</code>	Data frame or tibble with proteins counts, cell condition, and group information
<code>protein_names</code>	A vector of column names of protein to use in the analysis
<code>condition</code>	The column name of the condition variable
<code>group</code>	The column name of the group variable

covariate_names	The column names of covariates
cell_n_min	Remove samples that are below this cell counts threshold
cell_n_subsample	Subsample samples to have this maximum cell count
num_boot	Number of bootstrap samples
num_cores	Number of computing cores

**Value**

A list of class `cytoglm` containing

tb_coef	coefficent table
df_samples_subset	possibly subsampled <code>df_samples_subset</code> table
protein_names	input protein names
condition	input condition variable
group	input group names
covariate_names	input covariates
cell_n_min	input <code>cell_n_min</code>
cell_n_subsample	input <code>cell_n_subsample</code>
unpaired	true if unpaired samples were provided as input
num_boot	input <code>num_boot</code>
num_cores	input <code>num_cores</code>
formula_str	formula use in the regression model

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
glm_fit <- CytoGLMM::cytoglm(df,
                             protein_names = protein_names,
                             condition = "condition",
                             group = "donor",
                             num_boot = 10) # in practice >=1000
glm_fit
```

cytoglmm

*Fit GLMM with method of moments***Description**

Fit GLMM with method of moments

**Usage**

```
cytoglmm(
  df_samples_subset,
  protein_names,
  condition,
  group = "donor",
  covariate_names = NULL,
  cell_n_min = Inf,
  cell_n_subsample = 0,
  num_cores = 1
)
```

**Arguments**

<code>df_samples_subset</code>	Data frame or tibble with proteins counts, cell condition, and group information
<code>protein_names</code>	A vector of column names of protein to use in the analysis
<code>condition</code>	The column name of the condition variable
<code>group</code>	The column name of the group variable
<code>covariate_names</code>	The column names of covariates
<code>cell_n_min</code>	Remove samples that are below this cell counts threshold
<code>cell_n_subsample</code>	Subsample samples to have this maximum cell count
<code>num_cores</code>	Number of computing cores

**Value**A list of class `cytoglm` containing

<code>glmmfit</code>	<code>mbest</code> object
<code>df_samples_subset</code>	possibly subsampled <code>df_samples_subset</code> table
<code>protein_names</code>	input protein names
<code>condition</code>	input condition variable
<code>group</code>	input group names

```

covariate_names      input covariates
cell_n_min           input cell_n_min
cell_n_subsample     input cell_n_subsample
num_cores            input num_cores

```

### Examples

```

set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
glmm_fit <- CytoGLMM::cytoglmm(df,
                                protein_names = protein_names,
                                condition = "condition",
                                group = "donor")

glmm_fit

```

---

cytgroup	<i>Group-specific fixed effects model</i>
----------	---

---

### Description

Group-specific fixed effects model

### Usage

```

cytgroup(
  df_samples_subset,
  protein_names,
  condition,
  group = "donor",
  cell_n_min = Inf,
  cell_n_subsample = 0
)

```

### Arguments

df_samples_subset	Data frame or tibble with proteins counts, cell condition, and group information
protein_names	A vector of column names of protein to use in the analysis
condition	The column name of the condition variable
group	The column name of the group variable
cell_n_min	Remove samples that are below this cell counts threshold
cell_n_subsample	Subsample samples to have this maximum cell count

**Value**

A list of class `cytoglm` containing

```

groupfit      glm object
df_samples_subset
              possibly subsampled df_samples_subset table
protein_names input protein names
condition     input condition variable
group        input group names
cell_n_min   input cell_n_min
cell_n_subsample
              input cell_n_subsample

```

**Examples**

```

set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
group_fit <- CytoGLMM::cytogroup(df,
                                protein_names = protein_names,
                                condition = "condition",
                                group = "donor")

group_fit

```

---

cytostab

*Evaluate parameter stability with respect to gating scheme*

---

**Description**

Evaluate parameter stability with respect to gating scheme

**Usage**

```

cytostab(
  df_samples_subset,
  protein_names,
  condition,
  group = "donor",
  cell_n_min = Inf,
  cell_n_subsample = 0
)

```



**Arguments**

df\_samples\_subset      Data frame or tibble with proteins counts, cell condition, and group information

protein\_names      A vector of column names of protein to use in the analysis

condition      The column name of the condition variable

group      The column name of the group variable

cell\_n\_min      Remove samples that are below this cell counts threshold

cell\_n\_subsample      Subsample samples to have this maximum cell count

**Value**

A data frame

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
stab <- CytoGLMM::cytostab(df,
                           protein_names = protein_names,
                           condition = "condition",
                           group = "donor")

stab
```

---

cyto\_check

*Check if input to cytoxxx function have errors*

---

**Description**

Check if input to cytoxxx function have errors

**Usage**

```
cyto_check(cell_n_subsample, cell_n_min, protein_names)
```

**Arguments**

cell\_n\_subsample      Subsample samples to have this maximum cell count

cell\_n\_min      A vector of column names of protein to use in the analysis

protein\_names      A vector of column names of protein to use in the analysis

**Value**

NULL.

generate\_data      *Generate dataset for vignettes and simulation studies*

---

### Description

Generate dataset for vignettes and simulation studies

### Usage

```
generate_data()
```

### Value

[tibble](#) data frame

### Examples

```
set.seed(23)
df <- generate_data()
str(df)
df
```

---

glmm\_moment      *Generalized linear mixed model with maximum likelihood*

---

### Description

Generalized linear mixed model with maximum likelihood

### Usage

```
glmm_moment(
  df_samples,
  protein_names,
  response,
  group = "donor",
  covariate_names = NULL,
  num_cores = 1
)
```

**Arguments**

df_samples	Data frame or tibble with proteins counts, cell condition, and group information
protein_names	A vector of column names of protein to use in the analysis
response	The column name of the condition variable
group	The column name of the group variable
covariate_names	The column names of covariates
num_cores	Number of computing cores

**Value**

`mbest` object

---

is_unpaired	<i>Check if samples match or paired on condition</i>
-------------	--

---

**Description**

Check if samples match or paired on condition

**Usage**

```
is_unpaired(df_samples_subset, condition, group)
```

**Arguments**

df_samples_subset	Data frame or tibble with proteins counts, cell condition, and group information
condition	The column name of the condition variable
group	The column name of the group variable

**Value**

A boolean

---

```
plot.cytoflexmix      Plot all components of mixture regression
```

---

**Description**

Plot all components of mixture regression

**Usage**

```
## S3 method for class 'cytoflexmix'
plot(x, k = NULL, separate = FALSE, ...)
```

**Arguments**

x	A cytoflexmix class
k	Number of clusters
separate	create two separate <a href="#">ggplot2</a> objects
...	Other parameters

**Value**

[ggplot2](#) object

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
mix_fit <- CytoGLMM::cytoflexmix(df,
                                protein_names = protein_names,
                                condition = "condition",
                                group = "donor",
                                ks = 2)

plot(mix_fit)
```

---

```
plot.cytoglm      Plot bootstrapped coefficients
```

---

**Description**

Plot bootstrapped coefficients

**Usage**

```
## S3 method for class 'cytoglm'
plot(x, order = FALSE, separate = FALSE, ...)
```

**Arguments**

x	A cytoglm class
order	Order the markers according to the mangintute of the coefficients
separate	create two separate <code>ggplot2</code> objects
...	Other parameters

**Value**

`ggplot2` object

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
glm_fit <- CytoGLMM::cytoglm(df,
                             protein_names = protein_names,
                             condition = "condition",
                             group = "donor",
                             num_boot = 10) # in practice >=1000
plot(glm_fit)
```

---

plot.cytoglm	<i>Plot fixed coefficients of random effects model</i>
--------------	--

---

**Description**

Plot fixed coefficients of random effects model

**Usage**

```
## S3 method for class 'cytoglm'
plot(x, order = FALSE, separate = FALSE, ...)
```

**Arguments**

x	A cytoglm class
order	Order the markers according to the mangintute of the coefficients
separate	create two separate <code>ggplot2</code> objects
...	Other parameters

**Value**

`ggplot2` object



---

plot_coeff	<i>Helper function to plot regression coefficient</i>
------------	---

---

**Description**

Helper function to plot regression coefficient

**Usage**

```
plot_coeff(
  tb,
  title_str,
  title_str_right,
  xlab_str,
  redline = 0,
  order = FALSE,
  separate = FALSE
)
```

**Arguments**

tb	A data frame
title_str	Title string for summary plot
title_str_right	Title for bootstrap sample plot
xlab_str	Label on x-axis
redline	Point on x-axis to draw the red line
order	Order the markers according to the magnitude of the coefficients
separate	Plot both summary and bootstrap samples

**Value**

[ggplot2](#) object or list of two objects if separate is true

---

plot_heatmap	<i>Heatmap of median marker expression</i>
--------------	--

---

**Description**

Heatmap of median marker expression

**Usage**

```
plot_heatmap(
  df_samples,
  sample_info_names,
  protein_names,
  arrange_by_1,
  arrange_by_2 = "",
  cluster_cols = FALSE,
  fun = median
)
```

**Arguments**

<code>df_samples</code>	Data frame or tibble with proteins counts, cell condition, and group information
<code>sample_info_names</code>	Column names that contain information about the cell, e.g. donor, condition, file name, or cell type
<code>protein_names</code>	A vector of column names of protein to use in the analysis
<code>arrange_by_1</code>	Column name
<code>arrange_by_2</code>	Column name
<code>cluster_cols</code>	Apply hierarchical cluster to columns
<code>fun</code>	Summary statistics of marker expression

**Value**

`pheatmap` object

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
CytoGLMM::plot_heatmap(df,
  protein_names = protein_names,
  sample_info_names = c("donor", "condition"),
  arrange_by_1 = "condition")
```

---

plot\_lda

*LDA on marker expression*

---

**Description**

LDA on marker expression



**Usage**

```
plot_lda(
  df_samples,
  protein_names,
  group,
  cor_scaling_factor = 1,
  arrow_color = "black",
  marker_color = "black",
  marker_size = 5
)
```

**Arguments**

df_samples	Data frame or tibble with proteins counts, cell condition, and group information
protein_names	A vector of column names of protein to use in the analysis
group	The column name of the group variable
cor_scaling_factor	Scaling factor of circle of correlations
arrow_color	Color of correlation circle
marker_color	Colors of marker names
marker_size	Size of markerr names

**Value**

`ggplot2` object

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
df$condition <- rep(c("A", "B", "C", "D"), each = length(df$condition)/4)
CytoGLMM::plot_lda(df,
  protein_names = protein_names,
  group = "condition",
  cor_scaling_factor = 2)
```

---

plot\_mds

*MDS on median marker expression*

---

**Description**

MDS on median marker expression

**Usage**

```
plot_mds(
  df_samples,
  protein_names,
  sample_info_names,
  color,
  sample_label = ""
)
```

**Arguments**

df_samples	Data frame or tibble with proteins counts, cell condition, and group information
protein_names	A vector of column names of protein to use in the analysis
sample_info_names	Column names that contain information about the cell, e.g. donor, condition, file name, or cell type
color	Column name
sample_label	Column name

**Value**

cowplot object

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
CytoGLMM::plot_mds(df,
  protein_names = protein_names,
  sample_info_names = c("donor", "condition"),
  color = "condition")
```

---

plot\_model\_selection *Plot model selection to choose number optimal number of clusters*

---

**Description**

Plot model selection to choose number optimal number of clusters

**Usage**

```
plot_model_selection(fit, k = NULL)
```

**Arguments**

fit                    A cytoflexmix class  
 k                     Number of clusters

**Value**

cowplot object

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
mix_fit <- CytoGLMM::cytoflexmix(df,
                                protein_names = protein_names,
                                condition = "condition",
                                group = "donor",
                                ks = 1:2)
plot_model_selection(mix_fit)
```

---

 plot\_prcomp

---

*Plot PCA of subsampled data using ggplot*


---

**Description**

Plot PCA of subsampled data using ggplot

**Usage**

```
plot_prcomp(
  df_samples,
  protein_names,
  color_var = "treatment",
  subsample_size = 10000,
  repel = TRUE
)
```

**Arguments**

df\_samples            Data frame or tibble with proteins counts, cell condition, and group information  
 protein\_names        A vector of column names of protein to use in the analysis  
 color\_var            A column name  
 subsample\_size      Subsample per color\_var variable  
 repel                Repel labels

**Value**

cowplot object

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
CytoGLMM::plot_prcomp(df,
                      protein_names = protein_names,
                      color_var = "condition")
```

---

```
print.cytoglm
```

```
Extract and print bootstrap GLM fit
```

---

**Description**

Extract and print bootstrap GLM fit

**Usage**

```
## S3 method for class 'cytoglm'
print(x, ...)
```

**Arguments**

x	A cytoglm class
...	Other parameters

**Value**

NULL.

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
glm_fit <- CytoGLMM::cytoglm(df,
                             protein_names = protein_names,
                             condition = "condition",
                             group = "donor",
                             num_boot = 10) # in practice >=1000

print(glm_fit)
```

---

print.cytoglmm	<i>Extact and print GLMM fit</i>
----------------	----------------------------------

---

**Description**

Extact and print GLMM fit

**Usage**

```
## S3 method for class 'cytoglmm'  
print(x, ...)
```

**Arguments**

x	A cytoglmm class
...	Other parameters

**Value**

NULL.

**Examples**

```
set.seed(23)  
df <- generate_data()  
protein_names <- names(df)[3:12]  
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))  
glmm_fit <- CytoGLMM::cytoglmm(df,  
                               protein_names = protein_names,  
                               condition = "condition",  
                               group = "donor")  
  
print(glmm_fit)
```

---

remove_samples	<i>Remove samples based on low cell counts</i>
----------------	--

---

**Description**

Remove samples based on low cell counts

**Usage**

```
remove_samples(df_samples_subset, condition, group, unpaired, cell_n_min)
```

**Arguments**

df_samples_subset	Data frame or tibble with proteins counts, cell condition, and group information
condition	The column name of the condition variable
group	The column name of the group variable
unpaired	true if unpaired samples were provided as input
cell_n_min	Remove samples that are below this cell counts threshold

**Value**

NULL.

---

summary.cytoglm	<i>Extract and calculate p-values of bootstrap GLM fit</i>
-----------------	--

---

**Description**

Extract and calculate p-values of bootstrap GLM fit

**Usage**

```
## S3 method for class 'cytoglm'
summary(object, method = "BH", ...)
```

**Arguments**

object	A cytoglm class
method	Multiple comparison adjustment method
...	Other parameters

**Value**

[tibble](#) data frame

**Examples**

```
set.seed(23)
df <- generate_data()
protein_names <- names(df)[3:12]
df <- dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))
glm_fit <- CytoGLMM::cytoglm(df,
                             protein_names = protein_names,
                             condition = "condition",
                             group = "donor",
                             num_boot = 10) # in practice >=1000

summary(glm_fit)
```

---

summary.cytoglmm	<i>Extract and calculate p-values of GLMM fit</i>
------------------	---

---

### Description

Extract and calculate p-values of GLMM fit

### Usage

```
## S3 method for class 'cytoglmm'  
summary(object, method = "BH", ...)
```

### Arguments

object	A cytoglmm class
method	Multiple comparison adjustment method
...	Other parameters

### Value

[tibble](#) data frame

### Examples

```
set.seed(23)  
df = generate_data()  
protein_names = names(df)[3:12]  
df = dplyr::mutate_at(df, protein_names, function(x) asinh(x/5))  
glmm_fit = CytoGLMM::cytoglmm(df,  
                               protein_names = protein_names,  
                               condition = "condition",  
                               group = "donor")  
  
summary(glmm_fit)
```

# Index

cowplot, [18–20](#)  
cyto\_check, [9](#)  
cytoflexmix, [3](#)  
cytoglm, [4](#)  
cytoglmm, [6](#)  
cytogroup, [7](#)  
cytostab, [8](#)

flexmix, [3](#)

generate\_data, [10](#)  
ggplot2, [12–15, 17](#)  
glm, [8](#)  
glmm\_moment, [10](#)

is\_unpaired, [11](#)

mbest, [6, 11](#)

pheatmap, [16](#)  
plot.cytoflexmix, [12](#)  
plot.cytoglm, [12](#)  
plot.cytoglmm, [13](#)  
plot.cytogroup, [14](#)  
plot\_coeff, [15](#)  
plot\_heatmap, [15](#)  
plot\_lda, [16](#)  
plot\_mds, [17](#)  
plot\_model\_selection, [18](#)  
plot\_prcomp, [19](#)  
print.cytoglm, [20](#)  
print.cytoglmm, [21](#)

remove\_samples, [21](#)

summary.cytoglm, [22](#)  
summary.cytoglmm, [23](#)

tibble, [10, 22, 23](#)