Package ‘CytoGLMM’

February 29, 2024

**Type**  Package

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

**Version**  1.10.0

**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://christofseiler.github.io/CytoGLMM),  [https://github.com/ChristofSeiler/CytoGLMM](https://github.com/ChristofSeiler/CytoGLMM)

**BugReports**  [https://github.com/ChristofSeiler/CytoGLMM/issues](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](https://git.bioconductor.org/packages/CytoGLMM)
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

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
ks
    A vector of cluster sizes
num_cores
    Number of computing cores

Value

A list of class cytoglm containing

flexmixfits
    list of flexmix objects
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
ks
    input ks
num_cores
    input num_cores
Examples

```r
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

```r
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

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

A list of class `cytoglm` containing

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

Examples

```r
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

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

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
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_cores: Number of computing cores

Value

A list of class cytoglm containing
glmfit: mbest object
df_samples_subset: possibly subsampled df_samples_subset table
protein_names: input protein names
condition: input condition variable
group: 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

```r
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
```

cytogroup

**Group-specific fixed effects model**

Description

Group-specific fixed effects model

Usage

```r
cytogroup(
    df_samples_subset, 
    protein_names, 
    condition, 
    group = "donor", 
    cell_n_min = Inf, 
    cell_n_subsample = 0
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df_samples_subset</td>
<td>Data frame or tibble with proteins counts, cell condition, and group information</td>
</tr>
<tr>
<td>protein_names</td>
<td>A vector of column names of protein to use in the analysis</td>
</tr>
<tr>
<td>condition</td>
<td>The column name of the condition variable</td>
</tr>
<tr>
<td>group</td>
<td>The column name of the group variable</td>
</tr>
<tr>
<td>cell_n_min</td>
<td>Remove samples that are below this cell counts threshold</td>
</tr>
<tr>
<td>cell_n_subsample</td>
<td>Subsample samples to have this maximum cell count</td>
</tr>
</tbody>
</table>

Value

A list of class cytoglm containing

- groupfit: glm object
- df_samples_subset: possibly subsampled df_samples_subset table
cytostab

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
## cyto_check

**Value**

A data frame

**Examples**

```r
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**

```r
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**

```r
generate_data()
```

**Value**

`tibble` data frame

**Examples**

```r
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**

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

**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

---

**Description**

Check if samples match or paired on condition

**Usage**

```r
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.cytoglm**

*Plot all components of mixture regression*

**Description**

Plot all components of mixture regression

**Usage**

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

**Arguments**

- `x`: A `cytoflexmix` class
- `k`: Number of clusters
- `separate`: create two separate `ggplot2` objects
- `...`: Other parameters

**Value**

`ggplot2` object

**Examples**

```r
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 bootstraped coefficients*

**Description**

Plot bootstraped coefficients

**Usage**

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

**Examples**

```r
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)
```
Arguments

- **x**
  A `cytoglm` class

- **order**
  Order the markers according to the magnitude of the coefficients

- **separate**
  Create two separate `ggplot2` objects

- **...**
  Other parameters

Value

- `ggplot2` object

Examples

```r
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)
```

Description

Plot fixed coefficients of random effects model

Usage

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

Arguments

- **x**
  A `cytoglm` class

- **order**
  Order the markers according to the magnitude of the coefficients

- **separate**
  Create two separate `ggplot2` objects

- **...**
  Other parameters

Value

- `ggplot2` object
Examples

```r
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")
plot(glmm_fit)
```

plot.cytogroup  Plot fixed coefficients of group-specific fixed effects model

Description

Plot fixed coefficients of group-specific fixed effects model

Usage

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

Arguments

- `x` A `cytoglmm` class
- `order` Order the markers according to the magnitude of the coefficients
- `separate` Create two separate `ggplot2` objects
- `...` Other parameters

Value

`ggplot2` object

Examples

```r
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")
plot(group_fit)
```
plot_coeff

Helper function to plot regression coefficient

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

Heatmap of median marker expression

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

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

Value

- `pheatmap` object

Examples

```r
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 marker 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

```r
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**

```r
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**

```r
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
print.cytoglm

Value

cowplot object

Examples

```r
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")

glm_fit <- CytoGLMM::cytoglm(df,
                             protein_names = protein_names,
                             condition = "condition",
                             group = "donor",
                             num_boot = 10) # in practice >=1000

print(glm_fit)
```

Description

Extract and print bootstrap GLM fit

Usage

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

Arguments

- `x` A cytoglm class
- `...` Other parameters

Value

NULL.

Examples

```r
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")

glm_fit <- CytoGLMM::cytoglm(df,
                             protein_names = protein_names,
                             condition = "condition",
                             group = "donor",
                             num_boot = 10) # in practice >=1000

print(glm_fit)
```
### print.cytoglmm

**Extract and print GLMM fit**

**Description**

Extract and print GLMM fit

**Usage**

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

**Arguments**

- `x`: A `cytoglmm` class
- `...`: Other parameters

**Value**

`NULL`.

**Examples**

```r
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

**Remove samples based on low cell counts**

**Description**

Remove samples based on low cell counts

**Usage**

```r
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.

Description

Extract and calculate p-values of bootstrap GLM fit

Usage

```r
## 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

```r
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)
```
**Description**

Extract and calculate p-values of GLMM fit

**Usage**

```r
## 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**

```r
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, 2

cytoglm, 4

cytoglmm, 5

cytogroup, 7

cytostab, 8

flexmix, 3

generate_data, 10

ggplot2, 12–15, 17

glm, 7

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