Package 'omu'

July 22, 2025

Title A Metabolomics Analysis Tool for Intuitive Figures and Convenient Metadata Collection

Version 1.1.2

Description Facilitates the creation of intuitive figures to describe metabolomics data by utilizing Kyoto Encyclopedia of Genes and Genomes (KEGG) hierarchy data, and gathers functional orthology and gene data from the KEGG-REST API.

Depends R (>= 3.3.0)

Imports plyr, dplyr, stringr, httr, ggfortify, ggplot2, magrittr, tidyr, broom, FSA, rstatix, randomForest, caret

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests knitr, rmarkdown

VignetteBuilder knitr

URL https://github.com/connor-reid-tiffany/Omu, https://www.kegg.jp/kegg/rest/keggapi.html

BugReports https://github.com/connor-reid-tiffany/Omu/issues

NeedsCompilation no

Author Connor Tiffany [aut, cre]

Maintainer Connor Tiffany <tiffanyc1@chop.edu>

Repository CRAN

Date/Publication 2024-03-06 23:40:02 UTC

Contents

assign_hierarchy 2
c57_nos2KO_mouse_countDF 3
c57_nos2KO_mouse_metadata 3

check_zeros 4
count_fold_changes 5
get_seqs 5
KEGG_gather 6
make_omelette 7
omu_anova 7
omu_summary 8
PCA_plot 9
pie_chart 10
plate_omelette 11
plate_omelette_rxnko 12
plot_bar 12
plot_boxplot 13
plot_heatmap 14
plot_rf_PCA 15
plot_variable_importance 15
plot_volcano 16
random_forest 17
ra_table 18
read_metabo 18
transform_metabolites 19
transform_samples 19

Index **20**

`assign_hierarchy` *Assign hierarchy metadata*

Description

Assigns hierarchy metadata to a metabolomics count matrix using identifier values. It can assign KEGG compound hierarchy, orthology hierarchy, or organism hierarchy data.

Usage

```
assign_hierarchy(count_data, keep_unknowns, identifier)
```

Arguments

`count_data`	a metabolomics count data frame with either a KEGG compound, orthology, or a gene identifier column
`keep_unknowns`	a boolean of either TRUE or FALSE. TRUE keeps unannotated compounds, FALSE removes them
`identifier`	a string that is either "KEGG" for metabolite, "KO" for orthology, "Prokaryote" for organism, or "Eukaryote" for organism

Examples

```
assign_hierarchy(count_data = c57_nos2KO_mouse_countDF, keep_unknowns = TRUE, identifier = "KEGG")
```

c57_nos2KO_mouse_countDF

c57b6J nos2KO metabolomics count matrix

Description

A dataset containing metabolomics counts for an experiment done using c57b6J wild type and c57b6J nos2 knockout mice

Usage

```
c57_nos2KO_mouse_countDF
```

Format

A data frame with 668 rows and 36 variables:

c57_nos2KO_mouse_metadata

c57b6J nos2KO meta data

Description

A a meta data file for the c57b6J metabolomics count matrix

Usage

```
c57_nos2KO_mouse_metadata
```

Format

A data frame with 29 rows and 4 variables:

check_zeros *Check data for zeros across samples within factor levels. Will determine if there are more zeros than a user specified threshold within any given factor level(s). Returns a vector of Metabolites that are 0 above the threshold in any given factor level.*

Description

Check data for zeros across samples within factor levels. Will determine if there are more zeros than a user specified threshold within any given factor level(s). Returns a vector of Metabolites that are 0 above the threshold in any given factor level.

Usage

```
check_zeros(
  count_data,
  metadata,
  numerator = NULL,
  denominator = NULL,
  threshold = 25,
  response_variable = "Metabolite",
  Factor
)
```

Arguments

count_data	A metabolomics count data frame
metadata	Metadata dataframe for the metabolomics count data frame
numerator	String of the first independent variable you wish to test. Defualt is NULL
denominator	String of the second independent variable you wish to test. Default is NULL.
threshold	Integer. A percentage threshold for the number of zeros in a Metabolite. Default is 25.
response_variable	String of the column header for the response variables, usually "Metabolite"
Factor	A factor with levels to test for zeros.

Examples

```
check_zeros(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
Factor = "Treatment")

check_zeros(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
Factor = "Treatment",numerator = "Strep", denominator = "Mock", threshold = 10)
```

count_fold_changes *Get counts for significant fold changes by metabolite class.*

Description

Takes an input data frame from the output of omu_summary and creates a data frame of counts for significantly changed metabolites by class hierarchy data.

Usage

```
count_fold_changes(count_data, column, sig_threshold, keep_unknowns)
```

Arguments

count_data	Output dataframe from the omu_summary function or omu_anova.
column	Metabolite metadata you want to group by, i.e. "Class", "Subclass_1".
sig_threshold	Significance threshold for compounds that go towars the count, sig_threshold = 0.05
keep_unknowns	TRUE or FALSE for whether to drop compounds that weren't assigned hierarchy metadata

Examples

```
c57_nos2KO_mouse_countDF <- assign_hierarchy(c57_nos2KO_mouse_countDF, TRUE, "KEGG")

t_test_df <- omu_summary(count_data = c57_nos2KO_mouse_countDF,
metadata = c57_nos2KO_mouse_metadata,
numerator = "Strep", denominator = "Mock", response_variable = "Metabolite",
Factor = "Treatment", log_transform = TRUE, p_adjust = "BH", test_type = "welch")

fold_change_counts <- count_fold_changes(count_data = t_test_df,
column = "Class", sig_threshold = 0.05, keep_unknowns = "FALSE")
```

get_seqs *Get nucleotide and amino acid sequences for genes*

Description

Function that gets nt and aa seqs for gene data from KEGG_gather

Usage

```
get_seqs(gene_data)
```

Arguments

gene_data	A dataframe with genes from KEGG_gather, with class seqs

Examples

```
## Not run:
gene_data <- c57_nos2KO_mouse_countDF[(1:2),]

gene_data <- KEGG_gather(gene_data)

gene_data <- KEGG_gather(gene_data)
gene_data <- gene_data[1:2,]

gene_data <- get_seqs(gene_data)

## End(Not run)
```

KEGG_gather *Gather metadata from KEGG for metabolites*

Description

Method for gathering metadata from the KEGG API.

Usage

```
KEGG_gather(count_data)

## S3 method for class 'cpd'
KEGG_gather(count_data)

## S3 method for class 'rxn'
KEGG_gather(count_data)

## S3 method for class 'KO'
KEGG_gather(count_data)
```

Arguments

count_data	A metabolomics count dataframe with a KEGG identifier columns

Examples

```
## Not run:
count_data <- assign_hierarchy(count_data = c57_nos2KO_mouse_countDF,
keep_unknowns = TRUE, identifier = "KEGG")

count_data <- subset(count_data, Subclass_2=="Aldoses")
```

```
count_data <- KEGG_gather(count_data = count_data)

## End(Not run)
```

`make_omelette` *Get metadata from KEGG API*

Description

Internal function for KEGG_Gather

Usage

```
make_omelette(count_data, column, first_char)
```

Arguments

`count_data`	The metabolomics count data
`column`	The name of the KEGG identifier being sent to the KEGG API
`first_char`	firct character in number being fed to KEGG database

`omu_anova` *Perform anova*

Description

Performs an anova across all response variables, followed by a Tukeys test on every possible contrast in your model and calculates group means and fold changes for each contrast. Returns a list of data frames for each contrast, and includes a dataframe of model residuals

Usage

```
omu_anova(
  count_data,
  metadata,
  response_variable = "Metabolite",
  model,
  log_transform = FALSE,
  method = "anova"
)
```

Arguments

count_data	A metabolomics count data frame
metadata	Metadata dataframe for the metabolomics count data frame
response_variable	String of the column header for the response variables, usually "Metabolite"
model	A formual class object, see ?formula for more info on formulas in R. an interaction between independent variables. Optional parameter
log_transform	Boolean of TRUE or FALSE for whether or not you wish to log transform your metabolite counts
method	A string of 'anova', 'kruskal', or 'welch'. anova performs an anova with a post hoc tukeys test, kruskal performs a kruskal wallis with a post hoc dunn test, welch performs a welch's anova with a post hoc games howell test

Examples

```
anova_df <- omu_anova(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
response_variable = "Metabolite", model = ~ Treatment, log_transform = TRUE)

anova_df <- omu_anova(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
response_variable = "Metabolite", model = ~ Treatment + Background, log_transform = TRUE)

anova_df <- omu_anova(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
response_variable = "Metabolite", model = ~ Treatment + Background + Treatment*Background,
log_transform = TRUE)
```

---

omu_summary *omu_summary Performs comparison of means between two independent variables, standard deviation, standard error, FDR correction, fold change, log2FoldChange. The order effects the fold change values*

Description

omu_summary Performs comparison of means between two independent variables, standard deviation, standard error, FDR correction, fold change, log2FoldChange. The order effects the fold change values

Usage

```
omu_summary(
  count_data,
  metadata,
  numerator,
  denominator,
  response_variable = "Metabolite",
```

```
  Factor,
  log_transform = FALSE,
  p_adjust = "BH",
  test_type = "welch",
  paired = FALSE
)
```

Arguments

count_data	should be a metabolomics count data frame
metadata	is meta data
numerator	is the variable you wish to compare against the denominator, in quotes
denominator	see above, in quotes
response_variable	the name of the column with your response variables
Factor	the column name for your independent variables
log_transform	TRUE or FALSE value for whether or not log transformation of data is performed before the t test
p_adjust	Method for adjusting the p value, i.e. "BH"
test_type	One of "mwu", "students", or "welch" to determine which model to use
paired	A boolean of TRUE or FALSE. If TRUE, performs a paired sample test. To perform a paired sample test, metadata must have a column named 'ID' containing the subject IDs.

Examples

```
omu_summary(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
numerator = "Strep", denominator = "Mock", response_variable = "Metabolite", Factor = "Treatment",
log_transform = TRUE, p_adjust = "BH", test_type = "welch")
```

PCA_plot *Create a PCA plot*

Description

Performs an ordination and outputs a PCA plot using a metabolomics count data frame and metabolomics metadata

Usage

```
PCA_plot(
  count_data,
  metadata,
  variable,
  color,
```

```
  response_variable = "Metabolite",
  label = FALSE,
  size = 2,
  ellipse = FALSE
)
```

Arguments

count_data	Metabolomics count data
metadata	Metabolomics metadata
variable	The independent variable you wish to compare and contrast
color	String of what you want to color by. Usually should be the same as variable.
response_variable	String of the response_variable, usually should be "Metabolite"
label	TRUE or FALSE, whether to add point labels or not
size	An integer for point size.
ellipse	TRUE or FALSE, whether to add confidence interval ellipses or not.

Examples

```
PCA_plot(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
variable = "Treatment", color = "Treatment", response_variable = "Metabolite")
```

---

pie_chart *Create a pie chart*

Description

Creates a pie chart as ggplot2 object using the output from ra_table.

Usage

```
pie_chart(ratio_data, variable, column, color)
```

Arguments

ratio_data	a dataframe object of percents. output from ra_table function
variable	The metadata variable you are measuring, i.e. "Class"
column	either "Increase", "Decrease", or "Significant_Changes"
color	string denoting color for outline. use NA for no outline

Examples

```
c57_nos2KO_mouse_countDF <- assign_hierarchy(c57_nos2KO_mouse_countDF, TRUE, "KEGG")

t_test_df <- omu_summary(count_data = c57_nos2KO_mouse_countDF,
metadata = c57_nos2KO_mouse_metadata,
numerator = "Strep", denominator = "Mock", response_variable = "Metabolite",
Factor = "Treatment",
log_transform = TRUE, p_adjust = "BH", test_type = "welch")

fold_change_counts <- count_fold_changes(count_data = t_test_df,
column = "Class", sig_threshold = 0.05, keep_unknowns = FALSE)

ra_table <- ra_table(fc_data = fold_change_counts, variable = "Class")

pie_chart(ratio_data = ra_table, variable = "Class", column = "Decrease", color = "black")
```

---

plate_omelette *plate_omelette Internal method for KEGG_Gather which parses flat text files*

Description

plate_omelette Internal method for KEGG_Gather which parses flat text files

Usage

```
plate_omelette(output)

## S3 method for class 'rxn'
plate_omelette(output)

## S3 method for class 'genes'
plate_omelette(output)

## S3 method for class 'KO'
plate_omelette(output)
```

Arguments

output	The metabolomics count dataframe

plate_omelette_rxnko *Clean up orthology metadata*

Description

Internal function for KEGG_Gather.rxn method KEGG_Gather.rxn requires dispatch on multiple elements, so There was no way to incorporate as a method

Usage

```
plate_omelette_rxnko(output)
```

Arguments

output	output from plate_omelette

plot_bar *Create a bar plot*

Description

Creates a ggplot2 object using the output file from the count_fold_changes function

Usage

```
plot_bar(fc_data, fill, size = c(1, 1), outline_color = c("black", "black"))
```

Arguments

fc_data	The output file from Count_Fold_Changes
fill	A character vector of length 2 containing colors for filling the bars, the first color is for the "Decrease" bar while the second is for "Increase"
size	A numeric vector of 2 numbers for the size of the bar outlines.
outline_color	A character vector of length 2 containing colors for the bar outlines

Examples

```
c57_nos2KO_mouse_countDF <- assign_hierarchy(c57_nos2KO_mouse_countDF, TRUE, "KEGG")

t_test_df <- omu_summary(count_data = c57_nos2KO_mouse_countDF,
metadata = c57_nos2KO_mouse_metadata, numerator = "Strep", denominator = "Mock",
response_variable = "Metabolite", Factor = "Treatment",
log_transform = TRUE, p_adjust = "BH", test_type = "welch")

fold_change_counts <- count_fold_changes(count_data = t_test_df,
column = "Class", sig_threshold = 0.05, keep_unknowns = FALSE)
```

```
plot_bar(fc_data = fold_change_counts, fill = c("firebrick2", "dodgerblue2"),
outline_color = c("black", "black"), size = c(1,1))
```

`plot_boxplot` *Create a box plot*

Description

Takes a metabolomics count data frame and creates boxplots. It is recommended to either subset, truncate, or agglomerate by hierarchical metadata.

Usage

```
plot_boxplot(
  count_data,
  metadata,
  aggregate_by,
  log_transform = FALSE,
  Factor,
  response_variable = "Metabolite",
  fill_list
)
```

Arguments

`count_data`	A metabolomics count data frame, either from read_metabo or omu_summary
`metadata`	The descriptive meta data for the samples
`aggregate_by`	Hierarchical metadata value to sum metabolite values by, i.e. "Class"
`log_transform`	TRUE or FALSE. Recommended for visualization purposes. If true data is transformed by the natural log
`Factor`	The column name for the experimental variable
`response_variable`	The response variable for the data, i.e. "Metabolite"
`fill_list`	Colors for the plot which is colored by Factor, in the form of c("")

Examples

```
c57_nos2KO_mouse_countDF <- c57_nos2KO_mouse_countDF[1:5,]
c57_nos2KO_mouse_countDF <- assign_hierarchy(c57_nos2KO_mouse_countDF, TRUE, "KEGG")

plot_boxplot(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
log_transform = TRUE, Factor = "Treatment", response_variable = "Metabolite",
aggregate_by = "Subclass_2", fill_list = c("darkgoldenrod1", "dodgerblue2"))
```

plot_heatmap *Create a heatmap*

Description

Takes a metabolomics count data frame and creates a heatmap. It is recommended to either subset, truncate, or agglomerate by metabolite metadata to improve legibility.

Usage

```
plot_heatmap(
  count_data,
  metadata,
  Factor,
  response_variable,
  log_transform = FALSE,
  high_color,
  low_color,
  aggregate_by
)
```

Arguments

count_data	A metabolomics count data frame.
metadata	The descriptive meta data for the samples.
Factor	The column name for the independent variable in your metadata.
response_variable	The response variable for the data, i.e. "Metabolite"
log_transform	TRUE or FALSE. Recommended for visualization purposes. If true data is transformed by the natural log.
high_color	Color for high abundance values
low_color	Color for low abundance values
aggregate_by	Hierarchical metadata value to sum metabolite values by, i.e. "Class"

Examples

```
c57_nos2KO_mouse_countDF <- assign_hierarchy(c57_nos2KO_mouse_countDF, TRUE, "KEGG")

plot_heatmap(count_data = c57_nos2KO_mouse_countDF, metadata = c57_nos2KO_mouse_metadata,
log_transform = TRUE, Factor = "Treatment", response_variable = "Metabolite",
aggregate_by = "Subclass_2", high_color = "darkgoldenrod1", low_color = "dodgerblue2")
```

plot_rf_PCA *plot_rf_PCA*

Description

PCA plot of the proximity matrix from a random forest classification model

Usage

```
plot_rf_PCA(rf_list, color, size, ellipse = FALSE, label = FALSE)
```

Arguments

rf_list	The output from the random_forest function. This only works on classification models.
color	A grouping factor. Use the one that was the LHS of your model parameter in the random_forest funciton
size	The number for point size in the plot
ellipse	TRUE or FALSE. Whether to plot with confidence interval ellipses or not.
label	TRUE or FALSE. Whether to include point labels or not.

Examples

```
rf_list <- random_forest(c57_nos2KO_mouse_countDF,c57_nos2KO_mouse_metadata,
Treatment ~.,c(60,40),500)
plot_rf_PCA(rf_list = rf_list, color = "Treatment", size = 1.5)
```

---

plot_variable_importance

plot_variable_importance

Description

Plot the variable importance from a random forest model. Mean Decrease Gini for Classification and

Usage

```
plot_variable_importance(rf_list, color = "Class", n_metabolites = 10)
```

Arguments

rf_list	The output from the random_forest function
color	Metabolite metadata to color by
n_metabolites	The number of metabolites to include. Metabolites are sorted by decreasing importance.

Examples

```
rf_list <- random_forest(c57_nos2KO_mouse_countDF,c57_nos2KO_mouse_metadata,
Treatment ~.,c(60,40),500)
plot_variable_importance(rf_list = rf_list, color = "Class", n_metabolites = 10)
```

`plot_volcano` *Create a volcano plot*

Description

Creates a volcano plot as ggplot2 object using the output of omu_summary

Usage

```
plot_volcano(
  count_data,
  column,
  size,
  strpattern,
  fill,
  sig_threshold,
  alpha,
  shape,
  color
)
```

Arguments

`count_data`	The output file from the omu_summary function.
`column`	The column with metadata you want to highlight points in the plot with, i.e. "Class"
`size`	Size of the points in the plot
`strpattern`	A character vector of levels of the column you want the plot to focus on, i.e. strpattern = c("Carbohydrates", "Organicacids")
`fill`	A character vector of colors you want your points to be. Must be of length 1 + length(strpattern) to account for points not in strpattern. Levels of a factor are organzed alphabetically. All levels not in the strpattern argument will be set to NA.
`sig_threshold`	An integer. Creates a horizontal dashed line for a significance threshold. i.e. sig_threshold = 0.05. Defaut value is 0.05
`alpha`	A character vector for setting transparency of factor levels.Must be of length 1 + length(strpattern) to account for points not in strpattern.
`shape`	A character vector for setting the shapes for your column levels. Must be of length 1 + length(strpattern) to account for points not in strpattern. See ggplot2 for an index of shape values.

color A character vector of colors for the column levels. Must be of length 1 + length(strpattern) to account for points not in strpattern. If you choose to use shapes with outlines, this list will set the outline colors.

Examples

```
c57_nos2KO_mouse_countDF <- assign_hierarchy(c57_nos2KO_mouse_countDF, TRUE, "KEGG")

t_test_df <-  omu_summary(count_data = c57_nos2KO_mouse_countDF,
metadata = c57_nos2KO_mouse_metadata, numerator = "Strep", denominator = "Mock",
response_variable = "Metabolite", Factor = "Treatment",
log_transform = TRUE, p_adjust = "BH", test_type = "welch")

plot_volcano(count_data = t_test_df, column = "Class", strpattern = c("Carbohydrates"),
fill = c("firebrick2", "white"), sig_threshold = 0.05, alpha = c(1,1),
shape = c(1,24), color = c("black", "black"), size = 2)

plot_volcano(count_data = t_test_df, sig_threshold = 0.05, size = 2)
```

---

random_forest *random_forest Perform a classification or regression random forest model*

Description

a wrapper built around the randomForest function from package randomForest. Returns a list with a randomForest object list, training data set, testing data set, metabolite metadata, and confusion matrices for training and testing data (if type was classification).

Usage

```
random_forest(
  count_data,
  metadata,
  model,
  training_proportion = c(80, 20),
  n_tree = 500
)
```

Arguments

count_data	Metabolomics data
metadata	sample data
model	a model of format variable ~.
training_proportion	a numeric vector of length 2, first element is the percent of samples to use for training the model, second element is the percent of samples used to test the models accuracy
n_tree	number of decision trees to create

Examples

```
rf_list <- random_forest(count_data = c57_nos2KO_mouse_countDF,metadata = c57_nos2KO_mouse_metadata,
model = Treatment ~.,training_proportion = c(60,40),n_tree = 500)
```

---

ra_table *Creates a ratio table from the count_fold_changes function output.*

Description

Create a ratio table

Usage

```
ra_table(fc_data, variable)
```

Arguments

fc_data	data frame output from the count_fold_changes function
variable	metadata from count_fold_changes, i.e. "Class"

Examples

```
c57_nos2KO_mouse_countDF <- assign_hierarchy(c57_nos2KO_mouse_countDF, TRUE, "KEGG")

t_test_df <- omu_summary(count_data = c57_nos2KO_mouse_countDF,
metadata = c57_nos2KO_mouse_metadata, numerator = "Strep", denominator = "Mock",
response_variable = "Metabolite", Factor = "Treatment",
log_transform = TRUE, p_adjust = "BH", test_type = "welch")

fold_change_counts <- count_fold_changes(count_data = t_test_df,
column = "Class", sig_threshold = 0.05, keep_unknowns = FALSE)

ra_table(fc_data = fold_change_counts, variable = "Class")
```

---

read_metabo *Import a metabolomics count data frame*

Description

Wrapper for read.csv that appends the "cpd" class and sets blank cells to NA. Used to import metabolomics count data into R.

Usage

```
read_metabo(filepath)
```

Arguments

filepath	a file path to your metabolomics count data

Examples

```
filepath_to_yourdata = paste0(system.file(package = "omu"), "/extdata/read_metabo_test.csv")
count_data <- read_metabo(filepath_to_yourdata)
```

---

transform_metabolites *transform_metabolites*

Description

A functional to transform metabolomics data across metabolites.

Usage

```
transform_metabolites(count_data, func)
```

Arguments

count_data	Metabolomics data
func	a function to transform metabolites by. can be an anonymous function

Examples

```
data_pareto_scaled <- transform_samples(count_data = c57_nos2KO_mouse_countDF,
function(x) x/sqrt(sd(x)))
```

---

transform_samples *transform_samples*

Description

A functional to transform metabolomics data across samples.

Usage

```
transform_samples(count_data, func)
```

Arguments

count_data	Metabolomics data
func	a function to transform samples by. can be an anonymous function

Examples

```
data_ln <- transform_samples(count_data = c57_nos2KO_mouse_countDF, log)
```

Index

∗ **datasets**
c57_nos2KO_mouse_countDF, 3
c57_nos2KO_mouse_metadata, 3

assign_hierarchy, 2

c57_nos2KO_mouse_countDF, 3
c57_nos2KO_mouse_metadata, 3
check_zeros, 4
count_fold_changes, 5

get_seqs, 5

KEGG_gather, 6

make_omelette, 7

omu_anova, 7
omu_summary, 8

PCA_plot, 9
pie_chart, 10
plate_omelette, 11
plate_omelette_rxnko, 12
plot_bar, 12
plot_boxplot, 13
plot_heatmap, 14
plot_rf_PCA, 15
plot_variable_importance, 15
plot_volcano, 16

ra_table, 18
random_forest, 17
read_metabo, 18

transform_metabolites, 19
transform_samples, 19