

# Package: nima (via r-universe)

September 4, 2024

**Title** Nima Hejazi's R Toolbox

**Version** 0.6.2

**Description** Miscellaneous R functions developed as collateral damage over the course of work in statistical and scientific computing for research. These include, for example, utilities that supplement existing idiosyncrasies of the R language, extend existing plotting functionality and aesthetics, help prepare data objects for imputation, and extend access to command line tools and systems-level information.

**Maintainer** Nima Hejazi <nh@nimahejazi.org>

**Depends** R (>= 3.2.3)

**Imports** utils, stats, assertthat, ggplot2, ggthemes, scales, gtools, dplyr, grid, gridExtra,

**Suggests** knitr, roxygen2, testthat, tibble, stringr

**License** MIT + file LICENSE

**URL** <https://github.com/nhejazi/nima>

**BugReports** <https://github.com/nhejazi/nima/issues>

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.0.2

**Repository** <https://nhejazi.r-universe.dev>

**RemoteUrl** <https://github.com/nhejazi/nima>

**RemoteRef** HEAD

**RemoteSha** b31aaef4f9a44e725cdd3004ab4b4d8e3b52f76

## Contents

absmax	2
attrnames	3
clear	3

commas . . . . .	4
discrete_by_quantile . . . . .	4
exit . . . . .	5
factor_to_num . . . . .	5
hweb . . . . .	6
lm_plot . . . . .	6
miss_ind . . . . .	7
mse . . . . .	7
nll . . . . .	8
openfile . . . . .	9
qq_plot . . . . .	9
scale_color_nima . . . . .	10
scale_fill_nima . . . . .	10
sim_plot . . . . .	11
summarize_sim . . . . .	12
theme_jetblack . . . . .	13
theme_nima . . . . .	13
uniqlen . . . . .	14
<b>Index</b>	<b>15</b>

---

absmax	<i>Maximum of Absolute Values of Vector</i>
--------	---

---

### Description

Take the maximum of the absolute values of an input vector.

### Usage

```
absmax(x, na.rm = FALSE)
```

### Arguments

x	A numeric vector or array.
na.rm	A logical indicating whether missing values should be removed.

### Value

The maximum of the absolute values of elements of the input vector.

### Examples

```
x <- c(5, 3, -9, -100, 3.14159, 7.5)
absmax(x)
```

---

attrnames	<i>Get Names of Attributes</i>
-----------	--------------------------------

---

**Description**

Get the names of the attributes of an input object.

**Usage**

```
attrnames(obj)
```

**Arguments**

obj            Any object.

**Value**

Vector of character strings with the names of the attributes.

**Examples**

```
x <- matrix(1:100, ncol = 5)
colnames(x) <- LETTERS[1:5]
attrnames(x)
```

---

clear	<i>Clear the Current Screen/Buffer</i>
-------	--

---

**Description**

Clear the screen with a call to [system](#) and clear.

**Usage**

```
clear()
```

**Details**

This function is merely a call to `system("clear")`

**Examples**

```
system("clear")
```

commas *Add Commas to a Large Number*

---

**Description**

Convert a number to a string, with commas inserted at every 3rd digit.

**Usage**

```
commas(numbers)
```

**Arguments**

numbers            Vector of non-negative numbers (will be rounded to integers)

**Value**

Character string with numbers written like "5,771,009".

**Examples**

```
commas(c(2300, 9000, 21456, 987654890, 1256787, 345765, 1432))
```

---

discrete\_by\_quantile *Discretize a Vector by Quantiles*

---

**Description**

Discretizes a non-factor input vector and returns the result as numeric.

**Usage**

```
discrete_by_quantile(x, ...)
```

**Arguments**

x                    A vector containing arbitrary data.  
...                  Additional arguments passed to [quantcut](#).

**Value**

A numeric vector with the data re-coded to based on the quantiles.

**Examples**

```
x <- rnorm(1000)  
discrete_by_quantile(x)
```

---

exit	<i>Exit R Without Saving</i>
------	------------------------------

---

**Description**

Exit R without saving workspace, using the ubiquitous UNIX syntax.

**Usage**

```
exit()
```

**Details**

This function is merely a call to `q("no")`.

---

factor_to_num	<i>Convert a Factor to Numeric</i>
---------------	------------------------------------

---

**Description**

Convert a factor with numeric levels to a non-factor (numeric).

**Usage**

```
factor_to_num(x)
```

**Arguments**

`x` A vector containing a factor with numeric levels.

**Value**

The input factor made into a numeric vector.

**Examples**

```
x <- factor(c(3, 4, 9, 4, 9), levels = c(3, 4, 9))
factor_to_num(x)
```

---

`hweb`*View HTML Version of Help Files*

---

**Description**

View the HTML version of a help file while running R from the terminal.

**Usage**

```
hweb(...)
```

**Arguments**

... Help topics.

**Details**

Calls function [help](#) using argument `htmlhelp=TRUE`.

**See Also**

[help](#), [help.start](#)

**Examples**

```
hweb(read.table)
```

---

`lm_plot`*Linear Model Diagnostic Plots*

---

**Description**

Produce standard diagnostic plots for linear models using `ggplot2`.

**Usage**

```
lm_plot(x, ...)
```

**Arguments**

`x` A linear model object produced by `lm()`.  
... Extra arguments, currently ignored.

**Examples**

```
n <- 100
x1 <- rnorm(n)
y1 <- rnorm(n)
linmod <- lm(y1 ~ x1)
plot(linmod)
```

---

miss_ind	<i>Add missingness indicators to existing data object</i>
----------	---

---

**Description**

Add indicator columns to a data.frame showing the pattern of missingness.

**Usage**

```
miss_ind(data, prefix = "miss_")
```

**Arguments**

data	A numeric vector or array.
prefix	A string used to name the indicator variables..

**Value**

An augmented data.frame with indicators for missingness patterns.

**Examples**

```
data <- data.frame(cbind(rnorm(10), runif(10)))
data[sample(nrow(data), 3), 1] <- NA
data[sample(nrow(data), 4), 2] <- NA
data <- miss_ind(data)
```

---

mse	<i>Mean Squared Error</i>
-----	---------------------------

---

**Description**

Compute the mean squared error (risk under L2 loss).

**Usage**

```
mse(prediction, outcome)
```

**Arguments**

prediction      A numeric vector of predictions.  
outcome         A numeric vector of outcomes actually observed.

**Examples**

```
x <- rnorm(100)
y <- x^2
test_x <- rnorm(100)
test_y <- test_x^2
mod <- glm(y ~ x)
pred <- predict(mod, newx = as.data.frame(test_x))
error <- mse(prediction = pred, outcome = test_y)
```

---

nll

*Risk for Cross-Entropy Loss*

---

**Description**

Compute the empirical risk under cross-entropy loss for binary predictions.

**Usage**

```
nll(prediction, outcome)
```

**Arguments**

prediction      A numeric vector of predicted probabilities.  
outcome         A numeric vector of binary outcomes actually observed.

**Examples**

```
n_obs <- 100
x <- rnorm(n_obs)
y <- rbinom(n_obs, 1, plogis(x^2))
test_x <- rnorm(n_obs)
test_y <- rbinom(n_obs, 1, plogis(test_x^2))
mod <- glm(y ~ x, family = "binomial")
pred <- predict(mod, newx = as.data.frame(test_x), type = "response")
error <- nll(prediction = unname(pred), outcome = test_y)
```

---

openfile	<i>Open a File</i>
----------	--------------------

---

**Description**

Open a file using `system` and `open`.

**Usage**

```
openfile(file)
```

**Arguments**

`file` File name (as character string).

**Details**

Open files from R by using the default operating system program.

**Examples**

```
## Not run:  
openfile("myplot.pdf")  
  
## End(Not run)
```

---

qq_plot	<i>Quantile-Quantile Plots</i>
---------	--------------------------------

---

**Description**

Produce standard quantile-quantile plots for modeling using `ggplot2`.

**Usage**

```
qq_plot(  
  x,  
  distribution = "norm",  
  ...,  
  line.estimate = NULL,  
  conf = 0.95,  
  labels = names(x)  
)
```

**Arguments**

x	A numeric vector of residuals from a generalized linear model.
distribution	The reference probability distribution for residuals.
...	Any additional parameters to be passed to distribution functions.
line.estimate	Should quantiles be estimated, if so which quantiles?
conf	The confidence level to be used with confidence intervals.
labels	The names to be used when identifying points on the Q-Q plot.

**Examples**

```
n <- 100
x1 <- rnorm(n)
y1 <- rnorm(n)
linmod <- lm(y1 ~ x1)
x <- linmod$residuals
qq_plot(x)
```

---

scale\_color\_nima      *Nima's ggplot2 theme - supplement: scale\_color*

---

**Description**

Nima's ggplot2 theme scale\_color supplement: colors optimized via ColorBrewer

**Usage**

```
scale_color_nima(...)
```

**Arguments**

...      Passed to [ggplot](#)

---

scale\_fill\_nima      *Nima's ggplot2 theme - supplement: scale\_fill*

---

**Description**

Nima's ggplot2 theme scale\_fill supplement: colors optimized via ColorBrewer

**Usage**

```
scale_fill_nima(...)
```

**Arguments**

...      Passed to [ggplot](#)

---

 sim\_plot

*Visualize Summaries of Simulation Results*


---

**Description**

Visualize Summaries of Simulation Results

**Usage**

```
sim_plot(x, ..., sample_sizes, stat = c("bias", "mc_var", "mse"))
```

**Arguments**

x	A list of several simulation summary objects, of class <code>simulation_stats</code> .
...	Extra arguments currently ignored.
sample_sizes	A numeric vector giving the sample sizes at which each of the simulations in the input <code>x</code> was performed. There should be one unique sample size corresponding to each element of <code>x</code> .
stat	A character indicating which of three simulation summary statistics for which to generate a plot. Options are currently limited to bias ( <code>"bias"</code> ), variance ( <code>"mc_var"</code> ), and mean-squared error ( <code>"mse"</code> ).

**Examples**

```
n_sim <- 100
n_obs <- c(100, 10000)
mu <- 2
sim_results <- lapply(n_obs, function(sample_size) {
  estimator_sim <- lapply(seq_len(n_sim), function(iter) {
    y_obs <- rnorm(sample_size, mu)
    est_param <- mean(y_obs)
    est_var <- var(y_obs)
    estimate <- tibble::as_tibble(list(
      param_est = est_param,
      param_var = est_var
    ))
    return(estimate)
  })
  estimates <- do.call(rbind, estimator_sim)
  return(estimates)
})
sim_summary <- lapply(sim_results, summarize_sim, truth = mu)
p_sim_summary <- sim_plot(sim_summary, sample_sizes = n_obs, stat = "mse")
p_sim_summary
```

---

summarize_sim	<i>Summarize Simulations Results</i>
---------------	--------------------------------------

---

**Description**

Summarize Simulations Results

**Usage**

```
summarize_sim(simulation_results, truth, ci_level = 0.95)
```

**Arguments**

simulation_results	A data.frame, tibble or similar with exactly two columns named "param_est" and "param_var" giving the estimate of a parameter of interest and estimate of its variance (based on a valid variance estimator specific to that parameter). Each row of this data structure corresponds to the parameter estimate and variance for a single iteration of several simulations.
truth	A numeric value giving the true value of the parameter of interest in the simulation setting.
ci_level	A numeric value giving the level of the confidence intervals to be generated around the parameter estimates and statistics computed to summarize the simulation.

**Examples**

```
n_sim <- 1000
n_obs <- c(100, 10000)
mu <- 2
sim_results <- lapply(n_obs, function(sample_size) {
  estimator_sim <- lapply(seq_len(n_sim), function(iter) {
    y_obs <- rnorm(sample_size, mu)
    est_param <- mean(y_obs)
    est_var <- var(y_obs) / sample_size
    estimate <- tibble::as_tibble(list(
      param_est = est_param,
      param_var = est_var
    ))
    return(estimate)
  })
  estimates <- do.call(rbind, estimator_sim)
  return(estimates)
})
sim_summary <- lapply(sim_results, summarize_sim, truth = mu)
```

---

theme_jetblack	<i>A jet black theme with inverted colors</i>
----------------	---

---

**Description**

A jet black theme with inverted colors

**Usage**

```
theme_jetblack(base_size = 12, base_family = "")
```

**Arguments**

base_size	Base font size
base_family	Base font family

**Value**

An object as returned by [theme](#)

**See Also**

[theme](#)

**Examples**

```
library(ggplot2)
p <- ggplot(mtcars, aes(y = mpg, x = disp, color = factor(cyl)))
p <- p + geom_point() + theme_jetblack()
p
```

---

theme_nima	<i>Nima's plotting theme</i>
------------	------------------------------

---

**Description**

Nima's ggplot2 theme: white background, colors optimized

**Usage**

```
theme_nima(base_size = 14, base_family = "Helvetica")

nima_theme(base_size = 14, base_family = "Helvetica")
```

**Arguments**

base\_size      Base font size  
base\_family    Base font family

**Value**

An object as returned by [theme](#)

**See Also**

[theme](#)

**Examples**

```
library(ggplot2)
p <- ggplot(mtcars, aes(y = mpg, x = disp, color = factor(cyl)))
p <- p + geom_point() + scale_fill_nima() + scale_color_nima()
p <- p + theme_nima()
p
```

---

uniqlen

*Find Number of Unique Values*

---

**Description**

Get the number of unique values in an input vector.

**Usage**

```
uniqlen(vec, na.rm = TRUE)
```

**Arguments**

vec              A vector of any type.  
na.rm            If TRUE, remove missing values.

**Value**

Number of unique values.

**Examples**

```
x <- c(1, 3, 1, 1, NA, 2, 2, 3, NA, NA, 1, 3, 1)
uniqlen(x)
uniqlen(x, na.rm = FALSE)
```

# Index

absmax, [2](#)  
attrnames, [3](#)

clear, [3](#)  
commas, [4](#)

discrete\_by\_quantile, [4](#)

exit, [5](#)

factor\_to\_num, [5](#)

ggplot, [10](#)

help, [6](#)  
help.start, [6](#)  
hweb, [6](#)

lm\_plot, [6](#)

miss\_ind, [7](#)  
mse, [7](#)

nima\_theme (theme\_nima), [13](#)  
nll, [8](#)

openfile, [9](#)

qq\_plot, [9](#)  
quantcut, [4](#)

scale\_color\_nima, [10](#)  
scale\_fill\_nima, [10](#)  
sim\_plot, [11](#)  
summarize\_sim, [12](#)  
system, [3, 9](#)

theme, [13, 14](#)  
theme\_jetblack, [13](#)  
theme\_nima, [13](#)

uniqlen, [14](#)