

# Package ‘jbb’

May 8, 2026

**Type** Package

**Title** Balamuta Miscellaneous

**Version** 0.1.1

**License** GPL (>= 2)

**Description** Set of common functions used for manipulating colors, detecting and interacting with 'RStudio', modeling, formatting, determining users' operating system, feature scaling, and more!

**URL** <https://github.com/coatless/jbb>

**BugReports** <https://github.com/coatless/jbb/issues>

**RoxygenNote** 7.0.2

**Encoding** UTF-8

**NeedsCompilation** no

**Author** James Balamuta [aut, cre, cph] (ORCID:  
<<https://orcid.org/0000-0003-2826-8458>>)

**Maintainer** James Balamuta <balamut2@illinois.edu>

**Repository** CRAN

**Date/Publication** 2020-01-08 16:10:07 UTC

## Contents

jbb-package . . . . .	2
acc . . . . .	3
celsius_to_fahrenheit . . . . .	3
celsius_to_kelvin . . . . .	4
char_at . . . . .	5
circle_matrix . . . . .	5
convert_cols . . . . .	6
external_graphs . . . . .	7
fahrenheit_to_celsius . . . . .	8
fahrenheit_to_kelvin . . . . .	9
feature_scaling . . . . .	9

floor_and_cap . . . . .	12
int_to_hex . . . . .	13
is_rstudio . . . . .	13
is_whole . . . . .	14
is_windows . . . . .	14
kelvin_to_celsius . . . . .	15
kelvin_to_fahrenheit . . . . .	15
lagged . . . . .	16
max_n . . . . .	17
mkdir . . . . .	18
mse . . . . .	19
pad_number . . . . .	20
require_linux . . . . .	20
rgb_to_hex . . . . .	21
rmse . . . . .	22
shade . . . . .	23
system_arch . . . . .	23
system_graphic_driver . . . . .	24
tint . . . . .	24
tr . . . . .	25
url_title . . . . .	26
<b>Index</b>	<b>27</b>

---

 jjb-package

*jjb: Balamuta Miscellaneous*


---

## Description

Set of common functions used for manipulating colors, detecting and interacting with 'RStudio', modeling, formatting, determining users' operating system, feature scaling, and more!

## Author(s)

**Maintainer:** James Balamuta <balamut2@illinois.edu> ([ORCID](#)) [copyright holder]

## See Also

Useful links:

- <https://github.com/coatless/jjb>
- Report bugs at <https://github.com/coatless/jjb/issues>

---

acc                      *Accuracy of the Model*

---

**Description**

Calculates the accuracy of the model by taking the mean of the number of times the truth,  $y$ , equals the predicted,  $\hat{y}$ .

**Usage**

```
acc(y, yhat)
```

**Arguments**

$y$                       A vector of the true  $y$  values  
 $yhat$                     A vector of predicted  $\hat{y}$  values.

**Value**

The accuracy of the classification in numeric form.

**Examples**

```
# Set seed for reproducibility
set.seed(100)

# Generate data
n = 1e2

y = round(runif(n))
yhat = round(runif(n))

# Compute
o = acc(y, yhat)
```

---

celsius\_to\_fahrenheit    *Celsius to Fahrenheit Conversion*

---

**Description**

Converts temperature recorded in Celsius to Fahrenheit.

**Usage**

```
celsius_to_fahrenheit(t_celsius)
```

**Arguments**

t\_celsius      Temperature recorded in Celsius.

**Value**

A numeric vector.

**Examples**

```
celsius_to_fahrenheit(33)
```

```
celsius_to_fahrenheit(0)
```

---

celsius\_to\_kelvin      *Celsius to Kelvin Conversion*

---

**Description**

Converts temperature recorded in Celsius to Kelvin.

**Usage**

```
celsius_to_kelvin(t_celsius)
```

**Arguments**

t\_celsius      Temperature recorded in Celsius.

**Value**

A numeric vector.

**Examples**

```
celsius_to_kelvin(92)
```

```
celsius_to_kelvin(32)
```

---

char_at	<i>Character at Position i</i>
---------	--------------------------------

---

**Description**

Returns the character at location *i* inside the string.

**Usage**

```
char_at(x, index)
```

**Arguments**

x	A character vector to extract position from.
index	An integer between 1 and length <i>n</i> .

**Value**

A character vector of length index.

**Author(s)**

James J Balamuta

**Examples**

```
# Example string
s = "statistics"

# Single character
char_at(s, 1)

# Vectorized position
char_at(s, c(2, 3))
```

---

circle_matrix	<i>Create a circle pattern within a matrix</i>
---------------	--

---

**Description**

Takes a default matrix and embeds circles within the matrix.

**Usage**

```
circle_matrix(m, n, x.center, y.center, r, f = 1)
```

**Arguments**

m	A int that is the number of rows of the matrix
n	A int that is the number of the columns of the matrix.
x.center	A vector of x coordinate center position of the circle.
y.center	A vector of y coordinate center position of the circle.
r	A vector of integers denoting the different circle radii.
f	A vector of values that specify what the inside of the circles should be.

**Value**

A matrix with circles imprinted within its dimensions.

**Author(s)**

James Balamuta

**Examples**

```
# Generate a basic circle matrix
circle_matrix(10, 10, 3, 4, 2)

# Generate two circles within the matrix
circle_matrix(10, 20, c(3,6), c(4,6), c(2,2))

# Different fills
circle_matrix(10, 20, c(3,6), c(4,6), c(2,2), f = c(1,2))
```

---

convert_cols	<i>Convert Multiple Columns of a data.frame All at once conversion of a data.frame from current column types to alternates.</i>
--------------	---

---

**Description**

Convert Multiple Columns of a data.frame

All at once conversion of a data.frame from current column types to alternates.

**Usage**

```
convert_cols(d, cast)
```

**Arguments**

d	A data.frame that needs to have specific columns converted.
cast	A string vector containing either: "n" (numeric), "c" (character), or "f" (factor).

**Value**

A data.frame with converted column types.

**Examples**

```
n = 100

st = sample(LETTERS, n, replace = TRUE)
sr = sample(letters, n, replace = TRUE)
num = rnorm(n)

d = data.frame(x = st, y = num, z = sr, stringsAsFactors = FALSE)

# Convert all columns

o = convert_cols(d, c("f", "c", "f"))

# Convert a subset
d[, c(1, 3)] = convert_cols(d[, c(1, 3)], c("f", "f"))
```

---

external\_graphs

*Change Default Graphing Device from RStudio*

---

**Description**

Checks to see if the user is in RStudio. If so, then it changes the device to a popup window.

**Usage**

```
external_graphs(ext = TRUE)
```

**Arguments**

**ext** A logical indicating whether the graph should be done externally or internally in RStudio.

**Details**

Depending on the operating system, the default drivers attempted to be used are:

- OS X: quartz()
- Linux: x11()
- Windows: windows()

Note, this setting is not permanent. Thus, the behavioral change will last until the end of the session.

Also, the active graphing environment will be killed. As a result, any graphs that are open will be deleted. You will have to regraph them.

**Value**

There is no return value. Instead, once finished, the function will cause a side effect to occur. See details for more.

**Author(s)**

James Balamuta

**Examples**

```
# Turn on external graphs
external_graphs()

# Turn off external graphs
external_graphs(FALSE)
```

---

fahrenheit\_to\_celsius *Fahrenheit to Celsius Conversion*

---

**Description**

Converts temperature recorded in Fahrenheit to Celsius.

**Usage**

```
fahrenheit_to_celsius(t_fahrenheit)
```

**Arguments**

t\_fahrenheit    Temperature recorded in Fahrenheit.

**Value**

A numeric vector.

**Examples**

```
fahrenheit_to_celsius(92)

fahrenheit_to_celsius(32)
```

---

fahrenheit\_to\_kelvin    *Fahrenheit to Kelvin to Conversion*

---

**Description**

Converts temperature recorded in Fahrenheit to Kelvin.

**Usage**

```
fahrenheit_to_kelvin(t_fahrenheit)
```

**Arguments**

t\_fahrenheit    Temperature recorded in Fahrenheit.

**Value**

A numeric vector.

**Examples**

```
fahrenheit_to_kelvin(92)
```

```
fahrenheit_to_kelvin(32)
```

---

feature\_scaling    *Feature Scaling*

---

**Description**

Scale features in a datasets.

**Usage**

```
feature_rescale(x, x_min = NULL, x_max = NULL)
```

```
feature_derescale(x_rescaled, x_min, x_max)
```

```
feature_norm(x, x_norm = NULL)
```

```
feature_denorm(x_norm_std, x_norm = NULL)
```

```
feature_standardize(x, x_mean = NULL, x_sd = NULL)
```

```
feature_destandardize(x_std, x_mean = NULL, x_sd = NULL)
```

**Arguments**

<code>x</code>	Numeric values
<code>x_min</code>	Minimum non-normalized numeric value
<code>x_max</code>	Maximum non-normalized numeric value
<code>x_rescaled</code>	Rescaled values of <code>x</code> .
<code>x_norm</code>	Euclidean norm of <code>x</code>
<code>x_norm_std</code>	Euclidean vector of normalized <code>x</code> values.
<code>x_mean</code>	Mean of <code>x</code> values
<code>x_sd</code>	Standard Deviation of <code>x</code> values
<code>x_std</code>	Z-transformed <code>x</code> values

**Details**

The following functions provide a means to either scale features or to descale the features and return them to normal. These functions are ideal for working with optimizers.

	Feature Scale	Feature Descale
	<code>feature_rescale</code>	<code>feature_derescale</code>
	<code>feature_norm</code>	<code>feature_denorm</code>
	<code>feature_standardize</code>	<code>feature_destandardize</code>

**Value**

A numeric vector.

**Feature Rescaling**

Convert the original data  $x$  to  $x_{scaled}$ :

$$x[scaled] = (x - x[min]) / (x[max] - x[min])$$

To move from the rescaled value  $x_{scaled}$  to the original value  $x$  use:

$$x = x[scaled] * (x[max] - x[min]) + x[min]$$

**Feature Standardization**

Convert the original data  $x$  to  $x_{std}$ :

$$x[std] = (x - avg[x]) / (sigma[x])$$

To move from the standardized value  $x_{std}$  to the original value  $x$  use:

$$x = x[std] * sigma[x] + avg[x]$$

## Feature Normalization

Convert the original data  $x$  to  $x_{norm}$ :

$$x[norm] = (x)/||x||$$

To move from the normalized value  $x_{norm}$  to the original value  $x$  use:

$$x = x[norm] * ||x||$$

## Author(s)

James Balamuta

## Examples

```
# Rescaling Features
temperatures = c(94.2, 88.1, 32, 0)

temp_min = min(temperatures)
temp_max = max(temperatures)

temperatures_norm = feature_rescale(temp_min, temp_max)
temperatures_denorm = feature_derescale(temperatures_norm, temp_min, temp_max)

all.equal(temperatures, temperatures_denorm)

# Norming Features
x = 1:10

x_norm = sqrt(sum(x^2))

x_norm_std = feature_norm(x, x_norm)

x_recover = feature_denorm(x_norm_std, x_norm)
all.equal(x, x_recover)

# Standardizing Features
x = 1:10

x_mean = mean(x)
x_sd = sd(x)

x_std = feature_standardize(x, x_mean, x_sd)
x_recovery = feature_destandardize(x, x_mean, x_sd)

all.equal(x, x_recovery)
```

---

`floor_and_cap`*Floor and Cap a Numeric Variable*

---

### Description

Determine the floor and cap of a numeric variable by taking quantiles. Using the quantiles, values in the data found to be *lower* or *higher* than the floor or cap are replaced.

### Usage

```
floor_and_cap(x, probs = c(0.025, 0.975))
```

### Arguments

<code>x</code>	A vector that has length $N$ .
<code>probs</code>	A vector containing two values between 0 and 1, with the first being less than the second.

### Value

A vector with the values floored and capped.

### Examples

```
# One case version
n = 100

x = rnorm(n)

x[n - 1] = -99999
x[n] = 10000

y = floor_and_cap(x)

# Dataset example

d = data.frame(x, y = rnorm(n))

o = sapply(d, floor_and_cap)
```

---

int_to_hex	<i>Convert 0-255 to a Hex number</i>
------------	--------------------------------------

---

**Description**

This is a helper function for [rgb\\_to\\_hex](#). This function takes a single R, G, or B numeric value and converts it to hex.

**Usage**

```
int_to_hex(n)
```

**Arguments**

n                    An int

**Value**

A string of length 2.

**Examples**

```
int_to_hex(22)
```

---

is_rstudio	<i>Is R Open in RStudio?</i>
------------	------------------------------

---

**Description**

Detects whether R is open in RStudio.

**Usage**

```
is_rstudio()
```

**Value**

A logical value that indicates whether R is open in RStudio.

**Author(s)**

James Balamuta

**Examples**

```
is_rstudio()
```

is\_whole

*Integer Check*

---

**Description**

Checks whether the submitted value is an integer

**Usage**

```
is_whole(x)
```

**Arguments**

x                    A numeric value to check to see if it is an integer.

**Value**

A boolean value indicating whether the value is an integer or not.

**Author(s)**

James Balamuta

**Examples**

```
is_whole(2.3)
is_whole(4)
is_whole(c(1,2,3))
is_whole(c(.4,.5,.6))
is_whole(c(7,.8,9))
```

---

is\_windows*Check for an Operating System*

---

**Description**

Performs a check to determine the OS

**Usage**

```
is_windows()
```

```
is_macos()
```

```
is_linux()
```

```
is_sun()
```

**Value**

Either TRUE or FALSE

**Author(s)**

James Joseph Balamuta

---

kelvin\_to\_celsius      *Kelvin to Celsius Conversion*

---

**Description**

Converts temperature recorded in Kelvin to Celsius.

**Usage**

```
kelvin_to_celsius(t_kelvin)
```

**Arguments**

t\_kelvin      Temperature recorded in Kelvin.

**Value**

A numeric vector.

**Examples**

```
kelvin_to_celsius(92)
```

```
kelvin_to_celsius(32)
```

---

kelvin\_to\_fahrenheit      *Kelvin to Fahrenheit Conversion*

---

**Description**

Converts temperature recorded in Celsius to Kelvin.

**Usage**

```
kelvin_to_fahrenheit(t_kelvin)
```

**Arguments**

t\_kelvin      Temperature recorded in Kelvin.

**Value**

A numeric vector.

**Examples**

```
kelvin_to_fahrenheit(92)
```

```
kelvin_to_fahrenheit(32)
```

---

lagged

*Lag Vector Values*

---

**Description**

Provides a lagging mechanism for vector data.

**Usage**

```
lagged(x, lag = 1)
```

**Arguments**

x                   A vec of data.

lag                  An integer value.

**Value**

A vector with lagged values and NAs.

**Author(s)**

James Balamuta

**Examples**

```
x = rnorm(10)
```

```
lagged(x, 2)
```

---

max_n	<i>Maxima and Minima n elements</i>
-------	-------------------------------------

---

**Description**

Obtain the Maximum or Minimum  $n$  elements from a vector.

**Usage**

```
max_n(x, n = 1L)
```

```
min_n(x, n = 1)
```

**Arguments**

x	Data vector
n	Number of observations to select

**Details**

The underlying function sorts the data using `base::sort()` and then extracts out the appropriate n-back or n-forward values.

As a result of the sorting procedure, this is an inefficient function.

**Value**

A vector containing the maximum/minimum of  $n$  elements.

**Examples**

```
x = 1:10

# Defaults to traditional max
# This is more costly to compute than using the regular max function.
max_n(x)

# Retrieve top two observations (highest first)
max_n(x, 2)

# Missing values have no effect on the sorting procedure
x[9] = NA
max_n(x, 3)

# Defaults to traditional min.
# This is more costly to compute than using the regular min function.
min_n(x)
min(x)

# Retrieve bottom two observations (lowest first)
```

```
min_n(x, 2)

# Missing values have no effect on the sorting procedure
x[2] = NA
min_n(x, 3)
```

---

mkdir

*Make Directory*

---

## Description

Create a directory using either a relative path or an absolute path.

## Usage

```
mkdir(dir, r = TRUE)
```

## Arguments

`dir` A string indicating the directory to make.

`r` A boolean that indicates whether the directories should be made recursively

## Value

New directory on file system

## Author(s)

James Balamuta

## Examples

```
# Make directory from working directory
mkdir("toad")

## This assumes the computer is on Windows and the C drive exists.
# Make directory from absolute path
mkdir("C:/path/to/dir/toad")
```

---

mse

*Mean Squared Error (MSE)*

---

### Description

Calculates the mean square of the model by taking the mean of the sum of squares between the truth,  $y$ , and the predicted,  $\hat{y}$  at each observation  $i$ .

### Usage

```
mse(y, yhat)
```

### Arguments

<code>y</code>	A vector of the true $y$ values
<code>yhat</code>	A vector of predicted $\hat{y}$ values.

### Details

The equation for MSE is:

$$\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

### Value

The MSE in numeric form.

### Examples

```
# Set seed for reproducibility
set.seed(100)

# Generate data
n = 1e2

y = rnorm(n)
yhat = rnorm(n, 0.5)

# Compute
o = mse(y, yhat)
```

---

pad_number	<i>Pad Numeric Numbers</i>
------------	----------------------------

---

**Description**

Add zeros before start of the number

**Usage**

```
pad_number(x)
```

**Arguments**

x	A vector
---	----------

**Value**

A character vector that is padded to the length of the maximum entry.

**Author(s)**

James Balamuta

**Examples**

```
# Padding applied
pad_number(8:10)

# No padding applied
pad_number(2:3)

# Pads non-negative number with 0.
# This needs to be improved slightly...
pad_number(-1:1)
```

---

require_linux	<i>Require a Specific Operating System</i>
---------------	--

---

**Description**

Mandates the presence of an operating system

**Usage**

```
require_linux()

require_windows()

require_macos()

require_sun()
```

**Details**

If any of these functions are called on the wrong operating system. A stop error is triggered and the function will fail.

**Author(s)**

James Joseph Balamuta

---

rgb\_to\_hex                      *Convert RGB Value to Hexadecimal*

---

**Description**

This function converts an RGB value to the hexadecimal numbering system.

**Usage**

```
rgb_to_hex(R, G, B, pound = TRUE)
```

**Arguments**

R	A int that is between 0 and 255 for the Red value.
G	A int that is between 0 and 255 for the Green value.
B	A int that is between 0 and 255 for the Blue value.
pound	A bool that indicates whether a pound sign should be prepended to the hexadecimal.

**Value**

A string containing the hexadecimal information.

**Examples**

```
# Hexadecimal with pound sign
rgb_to_hex(255,255,255)

# Hexadecimal without pound sign
rgb_to_hex(255,255,255,FALSE)
```

---

rmse

*Root Mean Squared Error (RMSE)*

---

### Description

Calculates the root mean square of the model by taking the square root of mean of the sum of squares between the truth,  $y$ , and the predicted,  $\hat{y}$  at each observation  $i$ .

### Usage

```
rmse(y, yhat)
```

### Arguments

`y`                    A vector of the true  $y$  values  
`yhat`                A vector of predicted  $\hat{y}$  values.

### Details

The formula for RMSE is:

$$\sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

### Value

The RMSE in numeric form

### Examples

```
# Set seed for reproducibility
set.seed(100)

# Generate data
n = 1e2

y = rnorm(n)
yhat = rnorm(n, 0.5)

# Compute
o = mse(y, yhat)
```

---

shade	<i>Shade an RGB value</i>
-------	---------------------------

---

**Description**

The function shades or darkens an RGB value by adding black to the values.

**Usage**

```
shade(rgb_value, shade_factor = 0.1)
```

**Arguments**

`rgb_value` A vector with length  $3 \times 1$ .  
`shade_factor` A double that ranges between  $[0, 1]$ .

**Value**

A matrix with dimensions  $3 \times 1$ .

**Examples**

```
shade(c(22, 150, 230), shade_factor = 0.5)
```

---

system_arch	<i>System Architecture</i>
-------------	----------------------------

---

**Description**

System Architecture

**Usage**

```
system_arch()
```

**Value**

Either "x64" or "x32"

---

`system_graphic_driver` *Natural Graphics Driver for Operating System*

---

**Description**

Provides the default operating system graphics utility

**Usage**

```
system_graphic_driver()
```

**Value**

A string that is either:

- "quartz": if on MacOS
- "windows": if on Windows
- "x11": if on Linux or Solaris

**Author(s)**

James Balamuta

**See Also**

[is\\_rstudio](#)

**Examples**

```
# Returns a string depending on test platform
system_graphic_driver()
```

---

`tint` *Tint an RGB value*

---

**Description**

The function tints or lightens an RGB value by adding white to the values.

**Usage**

```
tint(rgb_value, tint_factor = 0.2)
```

**Arguments**

`rgb_value` A vector with length  $3 \times 1$ .  
`tint_factor` A double that ranges between  $[0, 1]$ .

**Value**

A matrix with dimensions  $3 \times 1$ .

**Examples**

```
tint(c(22, 150, 230), tint_factor = 0.5)
```

---

tr

*Obtain the Trace of a Square Matrix*

---

**Description**

Calculates and returns the trace of a square matrix.

**Usage**

```
tr(x)
```

**Arguments**

x                    A matrix that is square e.g.  $N \times N$

**Value**

A matrix with circles imprinted within its dimensions.

**Author(s)**

James Balamuta

**Examples**

```
# I_2 matrix  
tr(diag(2))
```

---

`url_title`*Create a "safe" url title*

---

**Description**

Takes a string, forces characters to lower case, then removes punctuation and switch spaces to - instead of \_

**Usage**

```
url_title(st)
```

**Arguments**

`st`                    A string that needs to be a title in a url

**Value**

A string with the aforementioned modifications.

**Author(s)**

James Balamuta

**Examples**

```
url_title("My Name is Jaime!")
```

# Index

acc, 3

base::sort(), 17

celsius\_to\_fahrenheit, 3

celsius\_to\_kelvin, 4

char\_at, 5

circle\_matrix, 5

convert\_cols, 6

external\_graphs, 7

fahrenheit\_to\_celsius, 8

fahrenheit\_to\_kelvin, 9

feature\_denorm (feature\_scaling), 9

feature\_derescale (feature\_scaling), 9

feature\_destandardize  
(feature\_scaling), 9

feature\_norm (feature\_scaling), 9

feature\_rescale (feature\_scaling), 9

feature\_scaling, 9

feature\_standardize (feature\_scaling), 9

floor\_and\_cap, 12

int\_to\_hex, 13

is\_linux (is\_windows), 14

is\_macos (is\_windows), 14

is\_rstudio, 13, 24

is\_sun (is\_windows), 14

is\_whole, 14

is\_windows, 14

jjb (jjb-package), 2

jjb-package, 2

kelvin\_to\_celsius, 15

kelvin\_to\_fahrenheit, 15

lagged, 16

max\_n, 17

min\_n (max\_n), 17

mkdir, 18

mse, 19

pad\_number, 20

require\_linux, 20

require\_macos (require\_linux), 20

require\_sun (require\_linux), 20

require\_windows (require\_linux), 20

rgb\_to\_hex, 13, 21

rmse, 22

shade, 23

system\_arch, 23

system\_graphic\_driver, 24

tint, 24

tr, 25

url\_title, 26