Package 'assertive.matrices'

November 20, 2018

Type Package
Title Assertions to Check Properties of Matrices
Version 0.0-2
Date 2018-11-20
Author Richard Cotton [aut, cre]
Maintainer Richard Cotton <richierocks@gmail.com>
Description A set of predicates and assertions for checking the properties of matrices. This is mainly for use by other package developers who want to include run-time testing features in their own packages. End-users will usually want to use assertive directly.
URL https://bitbucket.org/richierocks/assertive.matrices
BugReports https://bitbucket.org/richierocks/assertive.matrices/issues
Depends R (>= 3.0.0)
Imports assertive.base (>= 0.0-2)
Suggests testthat
License GPL (>= 3)
LazyLoad yes
LazyData yes
Acknowledgments Development of this package was partially funded by the Proteomics Core at Weill Cornell Medical College in Qatar <http://qatar-weill.cornell.edu>. The Core is supported by 'Biomedical Research Program' funds, a program funded by Qatar Foundation.
Collate 'imports.R' 'assert-is-matrix.R' 'is-matrix.R'
RoxygenNote 6.1.0
NeedsCompilation no
Repository CRAN
Date/Publication 2018-11-20 18:10:03 UTC
assert_is_diagonal_matrix

Is the input a diagonal matrix?

Description
Checks that the input is a diagonal matrix.

Usage

assert_is_diagonal_matrix(x, tol = 100 * .Machine$double.eps,
                         severity = getOption("assertive.severity", "stop"))

is_diagonal_matrix(x, tol = 100 * .Machine$double.eps,
                    .xname = get_name_in_parent(x))

Arguments

x
  Input to check.

tol
  Absolute values smaller than tol are not considered.

severity
  How severe should the consequences of the assertion be? Either "stop", "warning",
  "message", or "none".

.xname
  Not intended to be used directly.

Value
TRUE if the input is all zeroes (after coercion to be a matrix).

Examples

x <- diag(3)
is_diagonal_matrix(x)  # TRUE
x[1, 2] <- 100 * .Machine$double.eps
is_diagonal_matrix(x)  # TRUE
is_diagonal_matrix(x)  # FALSE
assert_is_identity_matrix

Is the matrix an identity matrix?

Description

Checks that the input is an identity matrix.

Usage

assert_is_identity_matrix(x, tol = 100 * .Machine$double.eps, severity = getOption("assertive.severity", "stop"))

is_identity_matrix(x, tol = 100 * .Machine$double.eps, .xname = get_name_in_parent(x))

Arguments

x          Input to check.
tol        Absolute deviations from the expected values smaller than tol are not considered.
severity    How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
.xname      Not intended to be used directly.

Value

TRUE if the input is all zeroes (after coercion to be a matrix).

Examples

x <- diag(3)
is_identity_matrix(x)
x[, 2] <- 100 * .Machine$double.eps
is_identity_matrix(x)
is_identity_matrix(x)
assert_is_lower_triangular_matrix

**Is the matrix upper/lower triangular?**

**Description**
Checks that the input is an upper or lower triangular matrix.

**Usage**

```r
assert_is_lower_triangular_matrix(x, strictly = FALSE, tol = 100 * .Machine$double.eps, severity = getOption("assertive.severity", "stop"))
```

```r
assert_is_upper_triangular_matrix(x, strictly = FALSE, tol = 100 * .Machine$double.eps, severity = getOption("assertive.severity", "stop"))
```

```r
is_lower_triangular_matrix(x, strictly = FALSE, tol = 100 * .Machine$double.eps, .xname = get_name_in_parent(x))
```

```r
is_upper_triangular_matrix(x, strictly = FALSE, tol = 100 * .Machine$double.eps, .xname = get_name_in_parent(x))
```

**Arguments**

- **x** Input to check.
- **strictly** Logical. If TRUE, the diagonal must consist of zeroes.
- **tol** Absolute deviations from the expected values smaller than tol are not considered.
- **severity** How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
- **.xname** Not intended to be used directly.

**Value**

TRUE if the input is all zeroes (after coercion to be a matrix).

**Examples**

```r
x <- matrix(c(1, 2, 3, 0, 5, 6, 0, 0, 9), nrow = 3)
is_lower_triangular_matrix(x)
is_lower_triangular_matrix(x, strictly = TRUE)
is_upper_triangular_matrix(t(x))
is_upper_triangular_matrix(t(x), strictly = TRUE)
x[1, 2] <- 100 * .Machine$double.eps
is_lower_triangular_matrix(x)
```
assert_is_square_matrix

```r
is_lower_triangular_matrix(x)
```

assert_is_square_matrix

*Is the matrix a square matrix?*

**Description**

Checks that the input is a square matrix.

**Usage**

```r
assert_is_square_matrix(x, severity = getOption("assertive.severity", 
"stop"))
```

```r
is_square_matrix(x, .xname = get_name_in_parent(x))
```

**Arguments**

- `x`: Input to check.
- `severity`: How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
- `.xname`: Not intended to be used directly.

**Value**

TRUE if the input is all zeroes (after coercion to be a matrix).

**Examples**

```r
is_square_matrix(matrix(1:9L, nrow = 3))
is_square_matrix(matrix(1:12, nrow = 3))
```

assert_is_symmetric_matrix

*Is the input a symmetric matrix?*

**Description**

Checks that the input is a symmetric matrix.
assert_is_symmetric_matrix

Usage

```
assert_is_symmetric_matrix(x, tol = 100 * .Machine$double.eps, ...
  severity = getOption("assertive.severity", "stop"))
```

```
is_symmetric_matrix(x, tol = 100 * .Machine$double.eps,
  .xname = get_name_in_parent(x), ...)
```

Arguments

- **x**: Input to check.
- **tol**: Differences smaller than tol are not considered. Passed to `all.equal`.
- **...**: Passed to `all.equal`.
- **severity**: How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
- **.xname**: Not intended to be used directly.

Value

TRUE if the input is symmetric (after coercion to be a matrix).

Examples

```
m <- diag(3); m[3, 1] <- 1e-100
assert_is_symmetric_matrix(m)
# These examples should fail.
assertive.base::dont_stop(assert_is_symmetric_matrix(m, tol = 0))
```

assert_is_zero_matrix

Is the input a zero matrix

Description

Checks that the input is a matrix of zeroes.

Usage

```
assert_is_zero_matrix(x, tol = 100 * .Machine$double.eps,
  severity = getOption("assertive.severity", "stop"))
```

```
is_zero_matrix(x, tol = 100 * .Machine$double.eps,
  .xname = get_name_in_parent(x))
```
assert_is_zero_matrix

Arguments

- **x**: Input to check.
- **tol**: Absolute values smaller than tol are not considered.
- **severity**: How severe should the consequences of the assertion be? Either "stop", "warning", "message", or "none".
- **.xname**: Not intended to be used directly.

Value

TRUE if the input is all zeroes (after coercion to be a matrix).

Examples

```r
x <- matrix(numeric(9), 3)
is_zero_matrix(x)
x[1, 1] <- 100 * .Machine$double.eps
is_zero_matrix(x)
is_zero_matrix(x)
```
Index

assert_is_diagonal_matrix, 2
assert_is_identity_matrix, 3
assert_is_lower_triangular_matrix, 4
assert_is_square_matrix, 5
assert_is_symmetric_matrix, 5
assert_is_upper_triangular_matrix
   (assert_is_lower_triangular_matrix), 4
assert_is_zero_matrix, 6

is_diagonal_matrix
   (assert_is_diagonal_matrix), 2
is_identity_matrix
   (assert_is_identity_matrix), 3
is_lower_triangular_matrix
   (assert_is_lower_triangular_matrix), 4
is_square_matrix
   (assert_is_square_matrix), 5
is_symmetric_matrix
   (assert_is_symmetric_matrix), 5
is_upper_triangular_matrix
   (assert_is_lower_triangular_matrix), 4
is_zero_matrix (assert_is_zero_matrix), 6