Package 'quantities'

May 9, 2026

Type Package

Title Quantity Calculus for R Vectors

Version 0.2.3

Description Integration of the 'units' and 'errors' packages for a complete quantity calculus system for R vectors, matrices and arrays, with automatic propagation, conversion, derivation and simplification of magnitudes and uncertainties. Documentation about 'units' and 'errors' is provided in the papers by Pebesma, Mailund & Hiebert (2016, <doi:10.32614/RJ-2016-061>) and by Ucar, Pebesma & Azcorra (2018, <doi:10.32614/RJ-2018-075>), included in those packages as vignettes; see 'citation(``quantities'')' for details.

License MIT + file LICENSE

Encoding UTF-8

URL https://r-quantities.github.io/quantities/, https://github.com/r-quantities/quantities

BugReports https://github.com/r-quantities/quantities/issues

Depends R (>= 3.1.0), units (>= 0.8-0), errors (>= 0.4.0)

Imports Rcpp

Suggests dplyr (>= 1.0.0), vctrs (>= 0.5.0), tidyr, pillar, ggplot2 (> 3.2.1), testthat, vdiffr, knitr, rmarkdown

LinkingTo Rcpp (>= 0.12.10)

ByteCompile yes

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation yes

Author Iñaki Ucar [aut, cph, cre] (ORCID: <https://orcid.org/0000-0001-6403-5550>)

Maintainer Iñaki Ucar <iucar@fedoraproject.org>

Repository CRAN

Date/Publication 2025-01-18 21:20:02 UTC

Contents

`quantities-package` **quantities***: Quantity Calculus for R Vectors*

Description

Support for painless automatic units and uncertainty propagation in numerical operations. Both **units** and **errors** are integrated into a complete quantity calculus system within the R language. R vectors, matrices and arrays automatically propagate those attributes when you operate with `quantities` objects.

Author(s)

Iñaki Ucar

References

Edzer Pebesma, Thomas Mailund and James Hiebert (2016). Measurement Units in R. *The R Journal*, 8(2), 486–494. doi:10.32614/RJ2016061

Iñaki Ucar, Edzer Pebesma and Arturo Azcorra (2018). Measurement Errors in R. *The R Journal*, 10(2), 549–557. doi:10.32614/RJ2018075

See Also

Useful links:

- https://r-quantities.github.io/quantities/
- https://github.com/r-quantities/quantities
- Report bugs at https://github.com/r-quantities/quantities/issues

---

`as.data.frame.quantities` *Coerce to a Data Frame*

Description

S3 method for `quantities` objects (see `as.data.frame`).

Usage

```
## S3 method for class 'quantities'
as.data.frame(x, ...)
```

Arguments

`x`	any R object.
`...`	additional arguments to be passed to or from methods.

Examples

```
x <- set_quantities(1:3, m/s, 0.1)
y <- set_quantities(4:6, m/s, 0.2)
(z <- cbind(x, y))
as.data.frame(z)
```

`as.list.quantities` *Coerce to a List*

Description

S3 method for `quantities` objects (see `as.list`).

Usage

```
## S3 method for class 'quantities'
as.list(x, ...)
```

Arguments

x	object to be coerced or tested.
...	objects, possibly named.

Examples

```
x <- set_quantities(1:3, m/s, 0.1)
as.list(x)
```

---

as.matrix.quantities *Coerce to a Matrix*

Description

S3 method for quantities objects (see as.matrix).

Usage

```
## S3 method for class 'quantities'
as.matrix(x, ...)
```

Arguments

x	an R object.
...	additional arguments to be passed to or from methods.

Examples

```
as.matrix(set_quantities(1:3, m/s, 0.1))
```

---

c.quantities *Combine Values into a Vector or List*

Description

S3 method for quantities objects (see c).

Usage

```
## S3 method for class 'quantities'
c(...)
```

Arguments

`...`	objects to be concatenated. All `NULL` entries are dropped before method dispatch unless at the very beginning of the argument list.

Examples

```
c(set_quantities(1, m/s, 0.2), set_quantities(30, km/h, 0.1))
```

---

cbind.quantities *Combine R Objects by Rows or Columns*

Description

S3 methods for `quantities` objects (see `cbind`).

Usage

```
## S3 method for class 'quantities'
cbind(..., deparse.level = 1)

## S3 method for class 'quantities'
rbind(..., deparse.level = 1)
```

Arguments

`...`	(generalized) vectors or matrices. These can be given as named arguments. Other R objects may be coerced as appropriate, or S4 methods may be used: see sections 'Details' and 'Value'. (For the `"data.frame"` method of `cbind` these can be further arguments to `data.frame` such as `stringsAsFactors`.)
`deparse.level`	integer controlling the construction of labels in the case of non-matrix-like arguments (for the default method): `deparse.level = 0` constructs no labels; the default `deparse.level = 1` typically and `deparse.level = 2` always construct labels from the argument names, see the 'Value' section below.

See Also

`c.quantities`

Examples

```
x <- set_quantities(1, m/s, 0.1)
y <- set_quantities(1:3, m/s, 0.2)
z <- set_quantities(8:10, m/s, 0.1)
(m <- cbind(x, y)) # the '1' (= shorter vector) is recycled
(m <- cbind(m, z)[, c(1, 3, 2)]) # insert a column
(m <- rbind(m, z)) # insert a row
```

correl *Handle Correlations Between* quantities *Objects*

Description

Methods to set or retrieve correlations or covariances between quantities objects.

Usage

```
## S3 method for class 'quantities'
correl(x, y)

## S3 replacement method for class 'quantities'
correl(x, y) <- value

## S3 method for class 'quantities'
covar(x, y)

## S3 replacement method for class 'quantities'
covar(x, y) <- value
```

Arguments

x	an object of class quantities.
y	an object of class quantities of the same length as x.
value	a compatible object of class units of length 1 or the same length as x. For correlations, this means a unitless vector (a numeric vector is also accepted in this case). For covariances, this means the same magnitude as x*y.

See Also

correl.

Examples

```
x <- set_quantities(1:10, m/s, 0.1)
y <- set_quantities(10:1, km/h, 0.2)
correl(x, y) <- 0.1 # accepted
correl(x, y) <- set_units(0.1) # recommended
correl(x, y)
covar(x, y)
```

diff.quantities *Lagged Differences*

Description

S3 method for quantities objects (see diff).

Usage

```
## S3 method for class 'quantities'
diff(x, lag = 1L, differences = 1L, ...)
```

Arguments

x	a numeric vector or matrix containing the values to be differenced.
lag	an integer indicating which lag to use.
differences	an integer indicating the order of the difference.
...	further arguments to be passed to or from methods.

Examples

```
diff(set_quantities(1:10, m/s, 0.1), 2)
diff(set_quantities(1:10, m/s, 0.1), 2, 2)
x <- cumsum(cumsum(set_quantities(1:10, m/s, 0.1)))
diff(x, lag = 2)
diff(x, differences = 2)
```

drop_quantities *Drop Units and Errors*

Description

Drop Units and Errors

Usage

```
drop_quantities(x)

## S3 method for class 'quantities'
drop_units(x)

## S3 method for class 'quantities'
drop_errors(x)

## S3 method for class 'data.frame'
drop_quantities(x)
```

Arguments

x	a quantities object.

Details

drop_quantities is equivalent to quantities(x) <- NULL or set_quantities(x, NULL, NULL). drop_units is equivalent to units(x) <- NULL or set_units(x, NULL). drop_errors is equivalent to errors(x) <- NULL or set_errors(x, NULL).

Value

the numeric without any units or errors attributes, while preserving other attributes like dimensions or other classes.

---

errors *Handle Measurement Uncertainty on a Numeric Vector*

Description

Set or retrieve measurement uncertainty to/from numeric vectors (extensions to the **errors** package for quantities and units objects).

Usage

```
## S3 method for class 'units'
errors(x)

## S3 method for class 'mixed_units'
errors(x)

## S3 replacement method for class 'units'
errors(x) <- value

## S3 replacement method for class 'mixed_units'
errors(x) <- value

## S3 method for class 'units'
set_errors(x, value = 0)

## S3 method for class 'mixed_units'
set_errors(x, value = 0)

## S3 method for class 'units'
errors_max(x)

## S3 method for class 'units'
errors_min(x)
```

Arguments

x	a numeric object, or object of class quantities, units or errors.
value	a numeric vector or units object of length 1, or the same length as x (see details).

Details

For objects of class quantities or units, the errors() method returns a units object that matches the units of x. Methods `errors<-`() and set_errors() assume that the provided uncertainty (value) has the same units as x. However, it is a best practice to provide a value with explicit units. In this way, uncertainty can be provided in different (but compatible) units, and it will be automatically converted to the units of x (see examples below).

See Also

errors.

Examples

```
x <- set_units(1:5, m)
errors(x) <- 0.01 # implicit units, same as x
errors(x)
errors(x) <- set_units(1, cm) # explicit units
errors(x)
```

Extract.quantities *Extract or Replace Parts of an Object*

Description

S3 operators to extract or replace parts of quantities objects.

Usage

```
## S3 method for class 'quantities'
x[...]

## S3 method for class 'quantities'
x[[...]]

## S3 replacement method for class 'quantities'
x[...] <- value

## S3 replacement method for class 'quantities'
x[[...]] <- value
```

Arguments

x	object from which to extract element(s) or in which to replace element(s).
...	additional arguments to be passed to base methods (see Extract).
value	typically an array-like R object of a similar class as x.

Examples

```
x <- set_quantities(1:3, m/s, 0.1)
y <- set_quantities(4:6, m/s, 0.2)
(z <- rbind(x, y))
z[2, 2]
z[2, 2] <- -1
errors(z[[1, 2]]) <- 0.8 # assumes same unit
errors(z[[2, 2]]) <- set_units(80, cm/s)
z[, 2]
```

---

groupGeneric.quantities

S3 Group Generic Functions

Description

Math, Ops and Summary group generic methods for quantities objects (see groupGeneric for a comprehensive list of available methods).

Usage

```
## S3 method for class 'quantities'
Math(x, ...)

## S3 method for class 'quantities'
Ops(e1, e2)

## S3 method for class 'quantities'
Summary(..., na.rm = FALSE)
```

Arguments

x, e1, e2	objects.
...	further arguments passed to methods.
na.rm	logical: should missing values be removed?

Details

See groupGeneric.errors, Ops.units, Math.units, for further details.

Examples

```
x <- set_quantities(1:3, m/s, 0.1)
log(x)
cumsum(x)
cumprod(x)

a <- set_quantities(1:3, m/s, 0.1)
b <- set_quantities(1:3, m/s, 0.1)
a + b
a * b
a / b
a = set_quantities(1:5, m, 0.1)
a %/% a
a %/% set_quantities(2)
set_quantities(1:5, m^2, 0.1) %/% set_quantities(2, m, 0.1)
a %% a
a %% set_quantities(2)
c(min(x), max(x))
range(x)
sum(x)
```

---

parse_quantities *Parse Units and Errors*

Description

Functions to parse character vectors into quantities.

Usage

```
parse_quantities(x, decimal_mark)

parse_units(x, decimal_mark)

parse_errors(x, decimal_mark)
```

Arguments

x	a character vector to parse.
decimal_mark	the dot (.) if not provided.

Details

Each parse_*() function returns an object of the corresponding type, no matter what it is found. This means that, for parse_units, if errors are found, they are dropped with a warning. Similarly for parse_errors, if units are found, they are dropped with a warning. On the other hand, parse_quantities always returns a valid quantities object, even if no errors or units are found (then, zero error and dimensionless units are applied).

Value

A quantities, units or errors object respectively.

Examples

```
parse_quantities("(1.6021766208 +/- .0000000098) e-19 C")
parse_quantities("1.6021766208(98) e-19 C")
parse_units("1.6021766208 e-19 C")
parse_errors("1.6021766208(98) e-19")

# quantities are converted to the first unit
parse_quantities(c("12.34(2) m/s", "36.5(1) km/h"))

# or kept as a list of mixed units
parse_quantities(c("1.02(5) g", "2.51(0.01) V", "(3.23 +/- 0.12) m"))
```

---

quantities *Handle Measurement Units and Uncertainty on a Numeric Vector*

Description

Set or retrieve measurement units and uncertainty to/from numeric vectors.

Usage

```
quantities(x)

quantities(x) <- value

set_quantities(x, unit, errors = 0, ...,
  mode = units_options("set_units_mode"))
```

Arguments

x	a numeric object, or object of class quantities, units or errors.
value	a list of two components: an object of class units or symbolic_units (see units), and a numeric vector of length 1 or the same length as x (see errors).
unit	a units object, or something coercible to one with as_units (see set_units).
errors	a numeric vector of length 1 or the same length as x (see set_errors).
...	passed on to other methods.
mode	if "symbols" (the default), then unit is constructed from the expression supplied. Otherwise, ifmode = "standard", standard evaluation is used for the supplied value This argument can be set via a global option units_options(set_units_mode = "standard")

Details

quantities returns a named list with the units and errors attributes.

`quantities<-` sets the units and error values (and converts x into an object of class quantities). set_quantities is a pipe-friendly version of `quantities<-` and returns an object of class quantities.

See Also

errors, units, groupGeneric.quantities. Extract.quantities, c.quantities, rep.quantities, cbind.quantities. as.data.frame.quantities, as.matrix.quantities, t.quantities.

Examples

```
x = 1:3
class(x)
x
quantities(x) <- list("m/s", 0.1)
class(x)
x

(x <- set_quantities(x, m/s, seq(0.1, 0.3, 0.1)))
```

---

rep.quantities *Replicate Elements of Vectors and Lists*

Description

S3 method for quantities objects (see rep).

Usage

```
## S3 method for class 'quantities'
rep(x, ...)
```

Arguments

x	a vector (of any mode including a list) or a factor or (for rep only) a POSIXct or POSIXlt or Date object; or an S4 object containing such an object.
...	further arguments to be passed to or from other methods. For the internal default method these can include:
	times an integer-valued vector giving the (non-negative) number of times to repeat each element if of length length(x), or to repeat the whole vector if of length 1. Negative or NA values are an error. A double vector is accepted, other inputs being coerced to an integer or double vector.

`length.out` non-negative integer. The desired length of the output vector. Other inputs will be coerced to a double vector and the first element taken. Ignored if `NA` or invalid.

`each` non-negative integer. Each element of `x` is repeated `each` times. Other inputs will be coerced to an integer or double vector and the first element taken. Treated as `1` if `NA` or invalid.

Examples

```
rep(set_quantities(1, m/s, 0.1), 4)
```

`t.quantities` *Matrix Transpose*

Description

S3 method for `quantities` objects (see `t`).

Usage

```
## S3 method for class 'quantities'
t(x)
```

Arguments

`x`	a matrix or data frame, typically.

Examples

```
a <- matrix(1:30, 5, 6)
quantities(a) <- list("m/s", 1:30)
t(a)
```

`units` *Handle Measurement Units on a Numeric Vector*

Description

Set or retrieve measurement units to/from numeric vectors and convert units (extensions to the **units** package for `quantities` and `errors` objects).

Usage

```
## S3 replacement method for class 'quantities'
units(x) <- value

## S3 replacement method for class 'errors'
units(x) <- value

## S3 method for class 'errors'
set_units(x, value, ...,
  mode = units_options("set_units_mode"))

## S3 method for class 'quantities'
mixed_units(x, values, ...)

## S3 method for class 'errors'
mixed_units(x, values, ...)
```

Arguments

x	a numeric object, or object of class quantities, units or errors.
value	object of class units or symbolic_units, or in the case of set_units expression with symbols (see examples).
...	passed on to other methods.
mode	if "symbols" (the default), then unit is constructed from the expression supplied. Otherwise, ifmode = "standard", standard evaluation is used for the supplied value This argument can be set via a global option units_options(set_units_mode = "standard")
values	character vector with units encodings, or list with symbolic units of class mixed_symbolic_units

Details

For objects of class quantities, methods `units<-`() and set_units() automatically convert the associated uncertainty to the new unit (see examples below).

See Also

units, set_units.

Examples

```
(x <- set_quantities(1:5, m, 0.01))
set_units(x, cm)
```

Index