Package 'dual'

May 28, 2025

Type Package

Title Automatic Differentiation with Dual Numbers

Version 0.0.6

Date 2025-05-27

Maintainer Luca Sartore <drwolf85@gmail.com>

Description Automatic differentiation is achieved by using dual numbers without providing hand-coded gradient functions. The output value of a mathematical function is returned with the values of its exact first derivative (or gradient). For more details see Baydin, Pearlmutter, Radul, and Siskind (2018) <https://jmlr.org/papers/volume18/17-468/17-468.pdf>.

License GPL-3

Depends R (>= 3.2.0), base, stats, methods

NeedsCompilation no

Encoding UTF-8

RoxygenNote 7.3.1

Author Luca Sartore [aut] (ORCID = ``0000-0002-0446-1328''),
Luca Sartore [cre] (ORCID = ``0000-0002-0446-1328'')

Repository CRAN

Date/Publication 2025-05-28 04:00:02 UTC

Contents

`dual-package` *dual: Automatic Differentiation with Dual Numbers*

Description

Automatic differentiation is achieved by using dual numbers without providing hand-coded gradient functions. The output value of a mathematical function is returned with the values of its exact first derivative (or gradient). For more details see Baydin, Pearlmutter, Radul, and Siskind (2018) https://jmlr.org/papers/volume18/17-468/17-468.pdf.

Automatic differentiation is achieved by using dual numbers without providing hand-coded gradient functions. The output value of a mathematical function is returned with the values of its exact first derivative (or gradient). For more details see Baydin, Pearlmutter, Radul, and Siskind (2018) https://jmlr.org/papers/volume18/17-468/17-468.pdf.

Details

Package:	dual
Type:	Package
Version:	0.0.6
Date:	2025-05-27
License:	GPL-3

For a complete list of exported functions, use `library(help = "dual")`.

Author(s)

Maintainer: Luca Sartore <drwolf85@gmail.com> (ORCID = "0000-0002-0446-1328")

Authors:

- Luca Sartore <drwolf85@gmail.com> (ORCID = "0000-0002-0446-1328")

Luca Sartore <drwolf85@gmail.com>

Maintainer: Luca Sartore <drwolf85@gmail.com>

References

Baydin, A. G., Pearlmutter, B. A., Radul, A. A., & Siskind, J. M. (2018). Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, **18**, 1-43.

Cheng, H. H. (1994). Programming with dual numbers and its applications in mechanisms design. *Engineering with Computers*, **10**(4), 212-229.

Examples

```
library(dual)

# Initilizing variables of the function
x <- dual(f = 1.5, grad = c(1, 0, 0))
y <- dual(f = 0.5, grad = c(0, 1, 0))
z <- dual(f = 1.0, grad = c(0, 0, 1))
# Computing the function and its gradient
exp(z - x) * sin(x)^y / x

# General use for computations with dual numbers
a <- dual(1.1, grad = c(1.2, 2.3, 3.4, 4.5, 5.6))
0.5 * a^2 - 0.1

# Johann Heinrich Lambert's W-function
lambertW <- function(x) {
  w0 <- 1
  w1 <- w0 - (w0*exp(w0)-x)/((w0+1)*exp(w0)-(w0+2)*(w0*exp(w0)-x)/(2*w0+2))
  while(abs(w1-w0) > 1e-15) {
    w0 <- w1
    w1 <- w0 - (w0*exp(w0)-x)/((w0+1)*exp(w0)-(w0+2)*(w0*exp(w0)-x)/(2*w0+2))
  }
  return(w1)
}
lambertW(dual(1, 1))
```

`Arithmetic` *Arithmetic Operators*

Description

These unary and binary operators perform arithmetic on dual objects.

Usage

```
## S4 method for signature 'dual,missing'
e1 + e2

## S4 method for signature 'dual,numeric'
e1 + e2

## S4 method for signature 'numeric,dual'
e1 + e2

## S4 method for signature 'dual,dual'
e1 + e2
```

```
## S4 method for signature 'dual,missing'
e1 - e2

## S4 method for signature 'dual,numeric'
e1 - e2

## S4 method for signature 'numeric,dual'
e1 - e2

## S4 method for signature 'dual,dual'
e1 - e2

## S4 method for signature 'dual,numeric'
e1 * e2

## S4 method for signature 'numeric,dual'
e1 * e2

## S4 method for signature 'dual,dual'
e1 * e2

## S4 method for signature 'dual,numeric'
e1 / e2

## S4 method for signature 'numeric,dual'
e1 / e2

## S4 method for signature 'dual,dual'
e1 / e2

## S4 method for signature 'dual,numeric'
e1 ^ e2

## S4 method for signature 'numeric,dual'
e1 ^ e2

## S4 method for signature 'dual,dual'
e1 ^ e2
```

Arguments

e1	dual object or numeric value.
e2	dual object or numeric value.

Value

The correspondent values of the arithmetic operation on e1 and e2 is returned.

Examples

```
x <- dual(1.5, 1:0)
y <- dual(2.6, 0:1)
+x
-x
x - y
x * y
x / y
x ^ y
x + y
```

`dual-class` *Dual object class An S4 Class for dual numbers*

Description

The method `initialize` sets the initial values of a new object of the class `dual`.

The function `dual` generates an object of class `dual` for the representation of dual numbers.

The function `is.dual` returns `TRUE` if `x` is of the class `dual`. It retuns `FALSE` otherwise.

The method `show` shows the content of a `dual` object.

Usage

```
dual(f, grad)

## S4 method for signature 'dual'
initialize(.Object, f = numeric(0), grad = numeric(0))

dual(f, grad)

is.dual(x)

## S4 method for signature 'dual'
show(object)
```

Arguments

`f`	a single numeric value denoting the "Real" component of the dual number.
`grad`	a numeric vector rappresenting the "Dual" components of the dual number.
`.Object`	an object of class `dual` to be initialized
`x`	an object of class `dual`.
`object`	an object of class `dual` to be shown

Value

an object of the class `dual`.

a logical value indicating if the object is of the class `dual` or not.

Slots

`f` a single numeric value denoting the "Real" component of the dual number

`grad` a numeric vector rappresenting the "Dual" components of the dual number

Examples

```
x <- dual(3, 0:1)
library(dual)
x <- new("dual", f = 1, grad = 1)
is.dual(3)
is.dual(x)
```

`Error` *Special Functions of Mathematics*

Description

Special mathematical functions related to the error function.

The function `erfc(x)` is a variant of the cumulative normal (or Gaussian) distribution funciton.

The functions `erfinv(x)` and `erfcinv(x)` respectively implement the inverse functions of `erf(x)` and `erfc(x)`.

Usage

```
erf(x)

## S4 method for signature 'dual'
erf(x)

erfinv(x)

## S4 method for signature 'dual'
erfinv(x)

erfc(x)

## S4 method for signature 'dual'
erfc(x)

erfcinv(x)

## S4 method for signature 'dual'
erfcinv(x)
```

Arguments

x	dual object.

Value

A dual object containing the transformed values according to the chosen function.

Examples

```
x <- dual(0.5, 1)
erf(x)
erfc(x)
erfinv(x)
erfcinv(x)
```

Hyperbolic *Hyperbolic Functions*

Description

These functions provide the obvious hyperbolic functions. They respectively compute the hyperbolic cosine, sine, tangent, and their inverses, arc-cosine, arc-sine, arc-tangent.

Usage

```
## S4 method for signature 'dual'
cosh(x)

## S4 method for signature 'dual'
sinh(x)

## S4 method for signature 'dual'
tanh(x)

## S4 method for signature 'dual'
acosh(x)

## S4 method for signature 'dual'
asinh(x)

## S4 method for signature 'dual'
atanh(x)
```

Arguments

x	a dual object

Value

A dual object containing the transformed values according to the chosen function.

Examples

```
x <- dual(0.5, 1)
cosh(x)
sinh(x)
tanh(x)
acosh(1 + x)
asinh(x)
atanh(x)
```

---

log *Logarithms and Exponentials*

Description

Logarithms and Exponentials

Usage

```
## S4 method for signature 'dual'
log(x)

## S4 method for signature 'dual,numeric'
logb(x, base = exp(1))

## S4 method for signature 'numeric,dual'
logb(x, base = exp(1))

## S4 method for signature 'dual,dual'
logb(x, base = exp(1))

## S4 method for signature 'dual'
log10(x)

## S4 method for signature 'dual'
log2(x)

## S4 method for signature 'dual'
log1p(x)

## S4 method for signature 'dual'
exp(x)

## S4 method for signature 'dual'
expm1(x)
```

Arguments

x	a dual object or numeric value.
base	a dual object or a positive number. Defaults to e=exp(1).

Value

A dual object containing the transformed values according to the chosen function.

Examples

```
x <- dual(sqrt(pi), 1:0)
y <- dual(pi * .75, 0:1)
log(x)
logb(x, base = 1.1)
logb(3.1, base = x)

logb(x, y)
log10(x)
log2(x)

log1p(x)

exp(2*x)
expm1(2*x)
```

Logic *Logic Operators for Comparing Dual Numbers*

Description

These functions provide the operators for logical comparisons between dual numbers. These operators are designed to test equality and inequalities (such as "not equal", "greater", "less", "greater or equal", "less or equal").

Usage

```
## S4 method for signature 'dual,numeric'
e1 == e2

## S4 method for signature 'numeric,dual'
e1 == e2

## S4 method for signature 'dual,dual'
e1 == e2

## S4 method for signature 'dual,numeric'
e1 != e2
```

```
## S4 method for signature 'numeric,dual'
e1 != e2

## S4 method for signature 'dual,dual'
e1 != e2

## S4 method for signature 'dual,numeric'
e1 > e2

## S4 method for signature 'numeric,dual'
e1 > e2

## S4 method for signature 'dual,dual'
e1 > e2

## S4 method for signature 'dual,numeric'
e1 >= e2

## S4 method for signature 'numeric,dual'
e1 >= e2

## S4 method for signature 'dual,dual'
e1 >= e2

## S4 method for signature 'dual,numeric'
e1 < e2

## S4 method for signature 'numeric,dual'
e1 < e2

## S4 method for signature 'dual,dual'
e1 < e2

## S4 method for signature 'dual,numeric'
e1 <= e2

## S4 method for signature 'numeric,dual'
e1 <= e2

## S4 method for signature 'dual,dual'
e1 <= e2
```

Arguments

e1	dual object or numeric value.
e2	dual object or numeric value.

Value

The correspondent boolean/logical value of the comparison between `e1` and `e2` is returned.

Examples

```
x <- dual(1.5, 1:0)
y <- dual(2.6, 0:1)
x == y
x != y
x > y
x >= y
x < y
x <= y
```

MathFun *Miscellaneous Mathematical Functions*

Description

The function `abs(x)` computes the absolute value of `x`, while `sqrt(x)` computes the square root of `x`.

Usage

```
## S4 method for signature 'dual'
sqrt(x)

## S4 method for signature 'dual'
abs(x)
```

Arguments

`x`	a dual object or numeric value.

Value

A dual object containing the transformed values according to the chosen function.

Examples

```
x <- dual(4.3, 1:0)
y <- dual(7.6, 0:1)
abs(-2.2 * x + 0.321 * y)
sqrt(y - x)
```

Special *Special Functions of Mathematics*

Description

Special mathematical functions related to the beta and gamma.

Usage

```
## S4 method for signature 'dual,dual'
beta(a, b)

## S4 method for signature 'dual,numeric'
beta(a, b)

## S4 method for signature 'numeric,dual'
beta(a, b)

## S4 method for signature 'dual,dual'
lbeta(a, b)

## S4 method for signature 'dual,numeric'
lbeta(a, b)

## S4 method for signature 'numeric,dual'
lbeta(a, b)

## S4 method for signature 'dual'
gamma(x)

## S4 method for signature 'dual'
lgamma(x)

## S4 method for signature 'dual'
psigamma(x, deriv = 0L)

## S4 method for signature 'dual'
digamma(x)

## S4 method for signature 'dual'
trigamma(x)

## S4 method for signature 'dual,dual'
choose(n, k)

## S4 method for signature 'numeric,dual'
choose(n, k)
```

```
## S4 method for signature 'dual,numeric'
choose(n, k)

## S4 method for signature 'dual,dual'
lchoose(n, k)

## S4 method for signature 'numeric,dual'
lchoose(n, k)

## S4 method for signature 'dual,numeric'
lchoose(n, k)

## S4 method for signature 'dual'
factorial(x)

## S4 method for signature 'dual'
lfactorial(x)
```

Arguments

a	non-negative numeric value or dual object with non-negative real part.
b	non-negative numeric value or dual object with non-negative real part.
x	dual object or numeric value.
deriv	integer value.
n	dual object or numeric value.
k	dual object or numeric value.

Value

A dual object containing the transformed values according to the chosen function.

Examples

```
x <- dual(0.5, 1)
a <- dual(1.2, 1:0)
b <- dual(2.1, 0:1)

beta(a, b)
beta(1, b)
beta(a, 1)
lbeta(a, b)
lbeta(1, b)
lbeta(a, 1)

gamma(x)
lgamma(x)
psigamma(x, deriv = 0)
digamma(x)
```

```
trigamma(x)
psigamma(x, 2)
psigamma(x, 3)


n <- 7.8 + a
k <- 5.6 + b
choose(n, k)
choose(5, k)
choose(n, 2)

lchoose(n, k)
lchoose(5, k)
lchoose(n, 2)

factorial(x)
lfactorial(x)
```

`Trig` *Trigonometric Functions*

Description

These functions give the obvious trigonometric functions. They respectively compute the cosine, sine, tangent, arc-cosine, arc-sine, arc-tangent, and the two-argument arc-tangent.

`cospi(x)`, `sinpi(x)`, and `tanpi(x)`, compute `cos(pi*x)`, `sin(pi*x)`, and `tan(pi*x)`.

Usage

```
## S4 method for signature 'dual'
cos(x)

## S4 method for signature 'dual'
sin(x)

## S4 method for signature 'dual'
tan(x)

## S4 method for signature 'dual'
acos(x)

## S4 method for signature 'dual'
asin(x)

## S4 method for signature 'dual'
atan(x)
```

```
## S4 method for signature 'dual,numeric'
atan2(y, x)

## S4 method for signature 'numeric,dual'
atan2(y, x)

## S4 method for signature 'dual,dual'
atan2(y, x)

## S4 method for signature 'dual'
cospi(x)

## S4 method for signature 'dual'
sinpi(x)

## S4 method for signature 'dual'
tanpi(x)
```

Arguments

x	dual object or numeric value.
y	dual object or numeric value.

Value

A dual object containing the transformed values according to the chosen function.

Examples

```
x <- dual(1, 1:0)
y <- dual(1, 0:1)

cos(x)
sin(x)
tan(x)
acos(x - 0.5)
asin(x - 0.5)
atan(x - 0.5)
atan2(x, y)
atan2(2.4, y)
atan2(x, 1.2)
cospi(1.2 * x)
sinpi(3.4 * x)
tanpi(5.6 * x)
```

Index