

Package: dual (via r-universe)

August 28, 2024

Type Package

Title Automatic Differentiation with Dual Numbers

Version 0.0.5

Date 2023-10-02

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 yes

Encoding UTF-8

RoxygenNote 7.2.3

Author Luca Sartore [aut, cre]
(<<https://orcid.org/0000-0002-0446-1328>>)

Date/Publication 2023-10-03 01:20:02 UTC

Repository <https://drwolf85.r-universe.dev>

RemoteUrl <https://github.com/cran/dual>

RemoteRef HEAD

RemoteSha d60260aabefb627273d4c867a60a2f353fe1ad71

Contents

dual-package	2
Arithmetic	3
dual-class	5
Error	6

Hyperbolic	7
log	8
Logic	9
MathFun	11
Special	12
Trig	14
Index	16

dual-package	<i>Automatic Differentiation with Dual Numbers</i>
--------------	--

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>.

Details

```
Package: dual
Type: Package
Version: 0.0.5
Date: 2023-10-02
License: GPL-3
```

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

Author(s)

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 Machine 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)

# Initializing 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 returns 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

<code>f</code>	a single numeric value denoting the "Real" component of the dual number.
<code>grad</code>	a numeric vector representing the "Dual" components of the dual number.
<code>.Object</code>	an object of class <code>dual</code> to be initialized
<code>x</code>	an object of class <code>dual</code> .
<code>object</code>	an object of class <code>dual</code> 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 representing 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 function.

The functions `erfcinv(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)
```

```
erfcinv(x)
```

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

`!=` (Logic), 9
`!=`, dual, dual-method (Logic), 9
`!=`, dual, numeric-method (Logic), 9
`!=`, numeric, dual-method (Logic), 9
*** autodiff**
 dual-package, 2
*** differentiation**
 dual-package, 2
*** dual**
 dual-package, 2
*** numeric**
 dual-package, 2
`*` (Arithmetic), 3
`*`, dual, dual-method (Arithmetic), 3
`*`, dual, numeric-method (Arithmetic), 3
`*`, numeric, dual-method (Arithmetic), 3
`+` (Arithmetic), 3
`+`, dual, dual-method (Arithmetic), 3
`+`, dual, missing-method (Arithmetic), 3
`+`, dual, numeric-method (Arithmetic), 3
`+`, numeric, dual-method (Arithmetic), 3
`-` (Arithmetic), 3
`-`, dual, dual-method (Arithmetic), 3
`-`, dual, missing-method (Arithmetic), 3
`-`, dual, numeric-method (Arithmetic), 3
`-`, numeric, dual-method (Arithmetic), 3
`/` (Arithmetic), 3
`/`, dual, dual-method (Arithmetic), 3
`/`, dual, numeric-method (Arithmetic), 3
`/`, numeric, dual-method (Arithmetic), 3
`<` (Logic), 9
`<`, dual, dual-method (Logic), 9
`<`, dual, numeric-method (Logic), 9
`<`, numeric, dual-method (Logic), 9
`<=` (Logic), 9
`<=`, dual, dual-method (Logic), 9
`<=`, dual, numeric-method (Logic), 9
`<=`, numeric, dual-method (Logic), 9
`==` (Logic), 9
`==`, dual, dual-method (Logic), 9
`==`, dual, numeric-method (Logic), 9
`==`, numeric, dual-method (Logic), 9
`>` (Logic), 9
`>`, dual, dual-method (Logic), 9
`>`, dual, numeric-method (Logic), 9
`>`, numeric, dual-method (Logic), 9
`>=` (Logic), 9
`>=`, dual, dual-method (Logic), 9
`>=`, dual, numeric-method (Logic), 9
`>=`, numeric, dual-method (Logic), 9
`^` (Arithmetic), 3
`^`, dual, dual-method (Arithmetic), 3
`^`, dual, numeric-method (Arithmetic), 3
`^`, numeric, dual-method (Arithmetic), 3
`abs`, dual-method (MathFun), 11
`acos`, dual-method (Trig), 14
`acosh`, dual-method (Hyperbolic), 7
`arccos`, dual-method (Trig), 14
`arcsin`, dual-method (Trig), 14
`arctan`, dual-method (Trig), 14
Arithmetic, 3
`asin`, dual-method (Trig), 14
`asinh`, dual-method (Hyperbolic), 7
`atan`, dual-method (Trig), 14
`atan2`, ANY, dual-method (Trig), 14
`atan2`, dual, ANY-method (Trig), 14
`atan2`, dual, dual-method (Trig), 14
`atan2`, dual, numeric-method (Trig), 14
`atan2`, numeric, dual-method (Trig), 14
`atanh`, dual-method (Hyperbolic), 7
`beta`, dual, dual-method (Special), 12
`beta`, dual, numeric-method (Special), 12
`beta`, numeric, dual-method (Special), 12
`choose`, dual, dual-method (Special), 12
`choose`, dual, numeric-method (Special), 12
`choose`, numeric, dual-method (Special), 12

- cos, dual-method (Trig), 14
- cosh, dual-method (Hyperbolic), 7
- cospi, dual-method (Trig), 14
- digamma, dual-mehtod (Special), 12
- digamma, dual-method (Special), 12
- dual (dual-class), 5
- dual-class, 5
- dual-package, 2
- erf (Error), 6
- erf, dual-mehtod (Error), 6
- erf, dual-method (Error), 6
- erfc (Error), 6
- erfc, dual-mehtod (Error), 6
- erfc, dual-method (Error), 6
- erfcinv (Error), 6
- erfcinv, dual-mehtod (Error), 6
- erfcinv, dual-method (Error), 6
- erfinv (Error), 6
- erfinv, dual-mehtod (Error), 6
- erfinv, dual-method (Error), 6
- Error, 6
- exp (log), 8
- exp, dual-method (log), 8
- expm1 (log), 8
- expm1, dual-method (log), 8
- factorial, dual-mehtod (Special), 12
- factorial, dual-method (Special), 12
- gamma, dual-mehtod (Special), 12
- gamma, dual-method (Special), 12
- Hyperbolic, 7
- initialize, dual-method (dual-class), 5
- is.dual (dual-class), 5
- is.dual, ANY, dual-method (dual-class), 5
- lbeta, dual, dual-method (Special), 12
- lbeta, dual, numeric-method (Special), 12
- lbeta, numeric, dual-method (Special), 12
- lchoose, dual, dual-method (Special), 12
- lchoose, dual, numeric-method (Special), 12
- lchoose, numeric, dual-method (Special), 12
- lfactorial, dual-mehtod (Special), 12
- lfactorial, dual-method (Special), 12
- lgamma, dual-mehtod (Special), 12
- lgamma, dual-method (Special), 12
- log, 8
- log, dual-method (log), 8
- log10 (log), 8
- log10, dual-method (log), 8
- log1p (log), 8
- log1p, dual-method (log), 8
- log2 (log), 8
- log2, dual-method (log), 8
- logb (log), 8
- logb, dual, dual-method (log), 8
- logb, dual, numeric-method (log), 8
- logb, numeric, dual-method (log), 8
- Logic, 9
- MathFun, 11
- psigamma, dual, NULL-mehtod (Special), 12
- psigamma, dual-method (Special), 12
- show, dual-method (dual-class), 5
- sin, dual-method (Trig), 14
- sinh, dual-method (Hyperbolic), 7
- sinpi, dual-method (Trig), 14
- Special, 12
- sqrt, dual-method (MathFun), 11
- tan, dual-method (Trig), 14
- tanh, dual-method (Hyperbolic), 7
- tanpi, dual-method (Trig), 14
- Trig, 14
- trigamma, dual-mehtod (Special), 12
- trigamma, dual-method (Special), 12