Package 'ldat'

October 13, 2022

Title Large Data Sets

Version 0.3.3

Date 2020-03-11

Author Jan van der Laan

Maintainer Jan van der Laan <r@eoos.dds.nl>

Description Tools for working with vectors and data sets that are too large to keep in memory. Extends the basic functionality provided in the 'lvec' package. Provides basis statistical functionality of 'lvec' objects, such as arithmetic operations and calculating means and sums. Also implements 'data.frame'-like objects storing its data in 'lvec' objects.

URL https://github.com/djvanderlaan/ldat

Depends R (>= 3.2.0), lvec (>= 0.2.0), stats, Rcpp

Imports utils, methods

Suggests testthat

LinkingTo lvec, BH, Rcpp

SystemRequirements C++11

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-03-13 15:30:05 UTC

R topics documented:

all.lvec 2
append 3
as_ldat 4

chunk.ldat 4
chunkwise 5
clone.ldat 6
duplicated.lvec 6
elementwise 7
generate 8
is.na.lvec 8
ldat 9
lget.ldat 10
lset.ldat 10
match 11
Math.lvec 12
median.lvec 12
Ops.lvec 13
order.ldat 13
partial_sort 14
quantile.lvec 15
slice_range 15
sort.ldat 16
table 17
table.lvec 18
[.lvec 18

Index **20**

`all.lvec` *Basic summary functions for lvec objects*

Description

These functions should behave as their regular counterparts.

Usage

```
## S3 method for class 'lvec'
all(x, ..., na.rm = FALSE)

## S3 method for class 'lvec'
any(x, ..., na.rm = FALSE)

## S3 method for class 'lvec'
prod(x, ..., na.rm = FALSE)

## S3 method for class 'lvec'
sum(x, ..., na.rm = FALSE)

## S3 method for class 'lvec'
mean(x, ..., na.rm = FALSE)
```

```
## S3 method for class 'lvec'
max(x, ..., na.rm = FALSE)

## S3 method for class 'lvec'
min(x, ..., na.rm = FALSE)

## S3 method for class 'lvec'
range(x, ..., na.rm = FALSE)
```

Arguments

x	an lvec object
...	ignored.
na.rm	logical indicating whether missing values should be ignored

---

append *Append a vector to an lvec*

Description

Append a vector to an lvec

Usage

```
append(x, y, ...)

## S3 method for class 'lvec'
append(x, y, clone = TRUE, ...)

## S3 method for class 'ldat'
append(x, y, clone = TRUE, ...)
```

Arguments

x	lvec to append to.
y	vector to append to x. Is converted to lvec using as_lvec.
...	ignored; used to pass additional arguments to other methods.
clone	should x be cloned first. If not, the input x is modified.

Value

Returns an lvec combining both x and y. When x is NULL a clone of y is returned.

as_ldat *Convert r-objects to ldat's*

Description

Convert r-objects to ldat's

Usage

```
as_ldat(x, ...)

## S3 method for class 'data.frame'
as_ldat(x, ...)

## Default S3 method:
as_ldat(x, ...)

## S3 method for class 'ldat'
as_ldat(x, ...)
```

Arguments

x	object to convert
...	further arguments passed to or from other methods

Value

Returns a ldat with columns corresponding to the columns in x. When x is not a data.frame it is first converted to a data.frame using a call to as.data.frame.

Examples

```
a <- data.frame(a = 1:10, b = rnorm(10))
b <- as_ldat(a)
```

chunk.ldat *Generate a number of index ranges from an ldat object*

Description

The ranges have a maximum length.

Usage

```
## S3 method for class 'ldat'
chunk(x, chunk_size = 5e+06, ...)
```

Arguments

x	an object of type ldat for which the index ranges should be calculated.
chunk_size	a numeric vector of length 1 giving the maximum length of the chunks. When not given it uses the value of the option 'chunk_size' (see options) otherwise the default value.
...	ignored; used to pass additional arguments to other methods.

Details

The default chunk size can be changes by setting the option 'chunk_size', ('options(chunk_size = <new default chunk size>)').

---

chunkwise *Process an lvec in chunks*

Description

Process an lvec in chunks

Usage

```
chunkwise(x, init, update, final, ...)
```

Arguments

x	the lvec.
init	initialisation function. This function should accept an lvec as its first argument and return an initial value for the state.
update	update function. Called for each chunk of data. Receives the current value of the state as its first argument and the next chunk of data as its second argument. Should return an updated state. This function can be called multiple times.
final	finaliser function. Is called after processing the complete lvec. Receives the final state as its first argument. Should return the end result.
...	optional arguments passed on to the supplied functions.

Details

For examples of its use see mean.lvec and sum.lvec.

clone.ldat *Clone an ldat object*

Description

Clone an ldat object

Usage

```
## S3 method for class 'ldat'
clone(x, ...)
```

Arguments

x	ldat object to be cloned
...	ignored.

Details

Clones each of the vectors in the ldat object.

duplicated.lvec *Find duplicates in a vector*

Description

Find duplicates in a vector

Usage

```
## S3 method for class 'lvec'
duplicated(x, incomparables = FALSE, fromLast = FALSE, ...)

## S3 method for class 'lvec'
unique(x, incomparables = FALSE, ...)
```

Arguments

x	an object of type lvec.
incomparables	passed on to link{duplicated}.
fromLast	not supported.
...	passed on to duplicated.

Details

Because this function works on chunks of data the data first needs to be sorted. Since a non stable sort is used, which of the duplicates is marked as duplicate is undefined. This is unlike the regular `duplicated` in which `fromLast` determines which records are marked as duplicates.

The function processes the data in chunks. The size of the chunks can be controlled using the option 'chunk_size' (see `chunk`).

`elementwise` *Apply a function to each element of an lvec*

Description

Apply a function to each element of an lvec

Usage

```
elementwise(x, fun, ...)
```

Arguments

`x`	an object of type `lvec`.
`fun`	the function to apply to the `lvec`. This function receives chunks of the lvec (which are regular R-vector) and should return a (R) vector of the same length as its input.
`...`	passed on to `fun`.

Value

Returns a `link{lvec}` of the same length as the input. The type is determined by the output of `fun`.

Examples

```
# Calculate square root of lvec
x <- as_lvec(1:10)
y <- elementwise(x, sqrt)
# of course, this is already implemented
sqrt(x)
```

generate *Generate an lvec with (random) values*

Description

Generate an lvec with (random) values

Usage

```
generate(n, fun, ..., chunk_size = 5e+06)
```

Arguments

n	number of elements in result vector
fun	function that generates values. Should accept a number of elements to generate as its first argument.
...	additional arguments are passed on to fun.
chunk_size	the size of the chunks of values with which to fill the resulting lvec. When not given it uses the value of the option 'chunk_size' (see options) otherwise the default value.

Value

Returns an lvec with length n. The type is determined by the type of values returned by fun.

Examples

```
# generate an lvec with random normally distributed values with sd of 10
x <- generate(2E6, rnorm, sd = 10)
# generate lvec with random letters; use sample; expects n as its second
# argument, but we work around that by explicitly naming first argument x
y <- generate(2E6, sample, x = letters, replace = TRUE)
```

is.na.lvec *Simple elementwise functions*

Description

These are implementations for lvec object for their regular R counterparts.

Usage

```
## S3 method for class 'lvec'
is.na(x)
```

Arguments

x	an object f type lvec

Value

Returns an lvec of the same length as the input.

---

ldat *Create an ldat object*

Description

This function creates an ldat object, which behaves similar to a data.frame except that its columns are lvec. This allows and ldat to have an arbitrary large number of rows without running into memory problems.

Usage

```
ldat(...)

is_ldat(x)
```

Arguments

...	these arguments are of either the form 'tag = value' or 'value'. Each argument becomes a column in the ldat. All columns are required to have the same length.
x	object for which to check if it is of type ldat

Details

Each of the arguments of ldat is converted to an lvec when it isn't already and lvec using calls to as_lvec. The arguments are required to all have the same length (unlike data.frame).

Value

An object of type ldat. This object is basically a list with lvec objects.

Examples

```
# Create ldat object from r-objects
a <- ldat(id = 1:20, x = letters[1:20], y = rnorm(20))
# this is identical to
a <- ldat(id = as_lvec(1:20), x = as_lvec(letters[1:20]),
    y = as_lvec(rnorm(20)))
```

lget.ldat *Read elements from an ldat object*

Description

Read elements from an ldat object

Usage

```
## S3 method for class 'ldat'
lget(x, ...)
```

Arguments

x	the ldat to read from
...	passed on to lget.lvec.

Details

Indexing using index should follow the same rules as indexing a regular data.frame using a logical or numeric index. The range given by range includes both end elements. So, a range of c(1,3) selects the first three elements.

Value

Returns an ldat with the selected elements. In order to convert the selection to an R-vector as_rvec can e used.

lset.ldat *Set values is an ldat object*

Description

Set values is an ldat object

Usage

```
## S3 method for class 'ldat'
lset(x, index = NULL, values, range = NULL, ...)
```

Arguments

x	an object of type ldat
index	a numeric of logical vector with indices at which the values should be set.
values	a vector with new values.
range	a numeric vector of length 2 specifying a range of elements to select. Specify either index or range.
...	ignored.

Details

When values is a vector the values are assigned to each column in x. Otherwise, vectors is assumed to be a list or data.frame of the same length as x. Each element of values is assigned to the corresponding element of x.

---

match *Value matching*

Description

Value matching

Usage

```
match(x, table, ...)

## Default S3 method:
match(x, table, ...)

## S3 method for class 'lvec'
match(x, table, na_incomparable = FALSE, ...)
```

Arguments

x	lvec of values to be matched
table	vector of values in which to look for matches.
...	optional arguments passed to and from other methods.
na_incomparable	can NA's and NaN's be matched.

Value

Returns a numeric lvec of the same length as x with the corresponding indices of records in table with the same value. When no match in table is found, NA is returned for the corresponding record.

Math.lvec *Implementation of Math group generics for lvec*

Description

Implementation of Math group generics for lvec

Usage

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

Arguments

x	an object of type lvec.
...	passed on to the corresponding R functions

Details

Math is group generic implementing the following functions: abs, sign, sqrt, floor, ceiling, trunc, round, signif exp, log, expm1, log1p, cos, sin, tan, cospi, sinpi, tanpi, acos, asin, atan, cosh, sinh, tanh, acosh, asinh, atanh, lgamma, gamma, digamma, trigamma, cumsum, cumprod, cummax, cummin. For more information see Math.

Value

Returns an link{lvec} of the same length as the input.

median.lvec *Calculate the median of an lvec*

Description

Calculate the median of an lvec

Usage

```
## S3 method for class 'lvec'
median(x, na.rm = TRUE, ...)
```

Arguments

x	an object of type lvec.
na.rm	remove missing values before calculating the quantiles
...	ignored.

See Also

For more details see quantile.lvec.

`Ops.lvec` *Implementation of Ops group generics for lvec*

Description

Implementation of Ops group generics for lvec

Usage

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

Arguments

e1	an object of type lvec.
e2	an object of type lvec.

Details

Math is group generic implementing the following functions: '"+"', '"-"', '"*"', '"/"', '"^"', '" '"|"', '"!"' '"=="', '"!="', '"<"', '"<="', '">="', '">"'. For more information see Ops.

Value

Returns an link{lvec} of the same length as the input.

`order.ldat` *Order an ldat*

Description

Order an ldat

Usage

```
## S3 method for class 'ldat'
order(x, ...)
```

Arguments

x	ldat to sort
...	unused.

Value

Returns the order of `x`. Unlike the default `order` function in R, the sort used is not stable (e.g. in case there are multiple records with the same value in `x`, there relative order after sorting is not defined).

Examples

```
x <- as_ldat(iris)
o <- order(x[c("Sepal.Width", "Sepal.Length")])
```

`partial_sort` *Partial sort an lvec*

Description

Partial sort an lvec

Usage

```
partial_sort(x, pivots, clone = TRUE)

partial_order(x, pivots)
```

Arguments

`x`	an object of type `lvec`
`pivots`	a numeric vector with indices at which the vector will be sorted. See details for more information.
`clone`	clone the vector first before sorting; or sort (and therefore modify) the input vector directly.

Details

After partial sorting the vector values at the pivots are the same as the vector values of a completely sorted vector. Furthermore, for each pivot `i` all elements `x[j]; j < i` are smaller or equal to than `x[i]` and all elements `x[j]; j > i` are larger than or equal to `x[i]`.

The speed of this operation should be O(n, k) with n the size of the `lvec` and k the number of pivots.

Examples

```
x <- as_lvec(rnorm(100))
y <- partial_sort(x, c(10, 50, 90))
x_sorted <- sort(x)
stopifnot(all(y[c(10, 50, 90)] == x_sorted[c(10, 50, 90)]))
stopifnot(max(y[1:9]) <= min(y[11:100]))
stopifnot(max(y[1:49]) <= min(y[51:100]))
stopifnot(max(y[1:89]) <= min(y[91:100]))
```

quantile.lvec *Calculate the quantiles of an lvec*

Description

Calculate the quantiles of an lvec

Usage

```
## S3 method for class 'lvec'
quantile(
  x,
  probs = seq(0, 1, 0.25),
  names = TRUE,
  na.rm = TRUE,
  true_probs = FALSE,
  ...
)
```

Arguments

x	an object of type lvec.
probs	a numeric vector with probabilities ([0,1]).
names	add names to the result vector.
na.rm	remove missing values before calculating the quantiles
true_probs	add an attribute with the probabilities at the chosen pivots.
...	ignored.

Details

This function uses a more simple method than that used by the regular quantile method. It sorts the vector (using partial_sort for speed) and selects elements from x that correspond to the given probabilities. For example, when x has length of 11 and prob equal to 0.5, it selects the 6th element from the (partially) sorted x. For large enough vectors this is a reasonable approach.

slice_range *Select a range of records from an object*

Description

Select a range of records from an object

Usage

```
slice_range(x, range, begin = range[1], end = range[2], ...)

## S3 method for class 'lvec'
slice_range(x, range, begin = range[1], end = range[2], as_r = FALSE, ...)

## S3 method for class 'ldat'
slice_range(x, range, begin = range[1], end = range[2], as_r = FALSE, ...)

## Default S3 method:
slice_range(x, range, begin = range[1], end = range[2], ...)

## S3 method for class 'data.frame'
slice_range(x, range, begin = range[1], end = range[2], ...)
```

Arguments

x	the object to select items from
range	a numeric vector with two elements specifying the range to select.
begin	the first element to select.
end	the last element to select.
...	ignored; used to pass additional arguments to other methods.
as_r	convert the result to an R-object.

Examples

```
x <- as_lvec(1:20)
# Select elements 5:7
slice_range(x, range = c(5, 7))
slice_range(x, begin = 5, end = 7)
slice_range(x, range = c(5, 10), end = 7)
# also works for R-vectors
slice_range(1:20, range = c(5,7))
# convert lvec to rvec
slice_range(x, range = c(5,7), as_r = TRUE)
```

---

sort.ldat *Sort an ldat*

Description

Sort an ldat

Usage

```
## S3 method for class 'ldat'
sort(x, decreasing = FALSE, ...)
```

Arguments

x	ldat to sort
decreasing	unused (a value unequal to FALSE will generate an error).
...	unused.

Value

Sorts x and returns a sorted copy of x.

Examples

```
x <- as_ldat(iris)
sort(x)
```

---

table *Give the 'TRUE' indices of an lvec*

Description

Give the 'TRUE' indices of an lvec

Usage

```
table(...)

## Default S3 method:
table(...)

which(x, ...)

## Default S3 method:
which(x, ...)

## S3 method for class 'lvec'
which(x, ...)
```

Arguments

...	not used
x	logical lvec to get the indices from

Value

Returns a numeric lvec with the indices of the elements of x that are TRUE.

Examples

```
x <- as_lvec(runif(1E6) > 0.1)
which(x)
```

---

table.lvec *Create cross tables from lvec objects*

Description

Create cross tables from lvec objects

Usage

```
## S3 method for class 'lvec'
table(..., useNA = c("ifany", "no", "always"))

## S3 method for class 'ldat'
table(..., useNA = c("ifany", "no", "always"))
```

Arguments

...	an object of type lvec
useNA	what to do with missing values. See table.

Details

The function processes the data in chunks. The size of the chunks can be controlled using the option 'chunk_size' (see chunk).

See Also

This function duplicates the functionality of the table function.

[.lvec *Indexing of lvec objects*

Description

Indexing of lvec objects

Usage

```
## S3 method for class 'lvec'
x[i = NULL, range = NULL]

## S3 replacement method for class 'lvec'
x[i, range] <- value

## S3 method for class 'ldat'
x[i, j, drop = FALSE, range = NULL, clone = TRUE]

## S3 replacement method for class 'ldat'
x[i, range] <- value
```

Arguments

x	an object of type lvec
i	an index vector. See lget.
range	an range of indices. See lget.
value	new values. See lget.
j	a selection of columns (a character, numeric or logical vector).
drop	ignored; included for compatability with data.frame.
clone	clone columns when selecting only columns.

Details

These functions are a wrapper around lget and lset.

Index

[.ldat *([.lvec)*, 18
[.lvec, 18
[<-.ldat *([.lvec)*, 18
[<-.lvec *([.lvec)*, 18

abs, *12*
acos, *12*
acosh, *12*
all.lvec, 2
any.lvec *(*all.lvec*)*, 2
append, 3
as.data.frame, *4*
as_ldat, 4
as_lvec, *3*, *9*
as_rvec, *10*
asin, *12*
asinh, *12*
atan, *12*
atanh, *12*

ceiling, *12*
chunk, *7*, *18*
chunk.ldat, 4
chunkwise, 5
clone, *19*
clone.ldat, 6
cos, *12*
cosh, *12*
cospi, *12*
cummax, *12*
cummin, *12*
cumprod, *12*
cumsum, *12*

data.frame, *9*
digamma, *12*
duplicated, *6*, *7*
duplicated.lvec, 6

elementwise, 7
exp, *12*
expm1, *12*

floor, *12*

gamma, *12*
generate, 8

is.na.lvec, 8
is_ldat *(*ldat*)*, 9

ldat, *4–6*, 9, *10*, *11*, *13*, *17*
lgamma, *12*
lget, *19*
lget.ldat, 10
lget.lvec, *10*
log, *12*
log1p, *12*
lset, *19*
lset.ldat, 10
lvec, *3*, *5–9*, *11–15*, *17–19*

match, 11
Math, *12*
Math.lvec, 12
max.lvec *(*all.lvec*)*, 2
mean.lvec, *5*
mean.lvec *(*all.lvec*)*, 2
median.lvec, 12
min.lvec *(*all.lvec*)*, 2

Ops, *13*
Ops.lvec, 13
options, *5*, *8*
order, *14*
order.ldat, 13

partial_order *(*partial_sort*)*, 14
partial_sort, 14, *15*
prod.lvec *(*all.lvec*)*, 2

quantile, *15*
quantile.lvec, *13*, 15

range.lvec *(*all.lvec*)*, 2
round, *12*

sign, *12*
signif, *12*
sin, *12*
sinh, *12*
sinpi, *12*
slice_range, 15
sort.ldat, 16
sqrt, *12*
sum.lvec, *5*
sum.lvec *(*all.lvec*)*, 2

table, 17, *18*
table.ldat *(*table.lvec*)*, 18
table.lvec, 18
tan, *12*
tanh, *12*
tanpi, *12*
trigamma, *12*
trunc, *12*

unique.lvec *(*duplicated.lvec*)*, 6

which *(*table*)*, 17