

# Package ‘blosc’

September 11, 2025

**Title** Compress and Decompress Data Using the 'BLOSC' Library

**Version** 0.1.1

**Description** Arrays of structured data types can require large volumes of disk space to store. 'Blosc' is a library that provides a fast and efficient way to compress such data. It is often applied in storage of n-dimensional arrays, such as in the case of the geo-spatial 'zarr' file format. This package can be used to compress and decompress data using 'Blosc'.

**License** GPL (>= 3)

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**Depends** R (>= 4.3)

**Suggests** dplyr, knitr, reticulate, rmarkdown, spelling, testthat (>= 3.0.0)

**LinkingTo** cpp11

**URL** <https://pepijn-devries.github.io/blosc/>,  
<https://github.com/pepijn-devries/blosc/>

**BugReports** <https://github.com/pepijn-devries/blosc/issues>

**SystemRequirements** blosc: blosc-devel (rpm) or libblosc-dev (deb)

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**Language** en-US

**NeedsCompilation** yes

**Author** Pepijn de Vries [aut, cre] (ORCID:  
<<https://orcid.org/0000-0002-7961-6646>>),  
Chris Maiwald [cph],  
Alexander Gessler [cph]

**Maintainer** Pepijn de Vries <pepijn.devries@outlook.com>

**Repository** CRAN

**Date/Publication** 2025-09-11 10:00:19 UTC

Contents

|                          |          |
|--------------------------|----------|
| blosc_compress . . . . . | 2        |
| blosc_info . . . . .     | 3        |
| r_to_dtype . . . . .     | 4        |
| <b>Index</b>             | <b>6</b> |

---

|                |                                           |
|----------------|-------------------------------------------|
| blosc_compress | <i>Compress and decompress with Blosc</i> |
|----------------|-------------------------------------------|

---

Description

Use the Blosc library to compress or decompress data.

Usage

```
blosc_compress(  
    x,  
    compressor = "blosclz",  
    level = 7L,  
    shuffle = "noshuffle",  
    typesize = 4L,  
    ...  
)  
  
blosc_decompress(x, ...)
```

Arguments

|            |                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| x          | <p>In case of blosc_decompress(), x should always be raw data to be decompressed. Use ... arguments to convert decompressed data to a specific data type.</p> <p>In case of blosc_compress(), x should either be raw data or a vector of data to be compressed. In the latter case, you need to specify dtype (see r_to_dtype()) in order to convert the data to raw information first. See vignette("blosc-compression") for more details.</p> |
| compressor | The compression algorithm to be used. Can be any of "blosclz", "lz4", "lz4hc", "zlib", or "zstd".                                                                                                                                                                                                                                                                                                                                               |
| level      | An integer indicating the required level of compression. Needs to be between 0 (no compression) and 9 (maximum compression).                                                                                                                                                                                                                                                                                                                    |
| shuffle    | A shuffle filter to be activated before compression. Should be one of "noshuffle", "shuffle", or "bitshuffle".                                                                                                                                                                                                                                                                                                                                  |
| typesize   | BLOSC compresses arrays of structured data. This argument specifies the size (integer) of the data structure / type in bytes. Default is 4L bytes (i.e. 32 bits), which would be suitable for compressing 32 bit integers.                                                                                                                                                                                                                      |
| ...        | Arguments passed to r_to_dtype().                                                                                                                                                                                                                                                                                                                                                                                                               |

**Value**

In case of `blosc_compress()` a vector of compressed raw data is returned. In case of `blosc_decompress()` returns a vector of decompressed raw data. Or in in case `dtype` (see `dtype_to_r()`) is specified, a vector of the specified type is returned.

**Examples**

```
my_dat      <- as.raw(sample.int(2L, 10L*1024L, replace = TRUE) - 1L)
my_dat_out  <- blosc_compress(my_dat, typesize = 1L)
my_dat_decomp <- blosc_decompress(my_dat_out)

## After compressing and decompressing the data is the same as the original:
all(my_dat == my_dat_decomp)
```

---

|            |                                          |
|------------|------------------------------------------|
| blosc_info | <i>Information about compressed data</i> |
|------------|------------------------------------------|

---

**Description**

Obtain information about raw data compressed with blosc.

**Usage**

```
blosc_info(x, ...)
```

**Arguments**

|                  |                                                          |
|------------------|----------------------------------------------------------|
| <code>x</code>   | Raw data compressed with <code>blosc_compress()</code> . |
| <code>...</code> | Ignored                                                  |

**Value**

Returns a named list with information about blosc compressed data `x`.

**Examples**

```
data_compressed <-
  blosc_compress(volcano, typesize = 2, dtype = "<i2", compressor = "lz4",
    shuffle = "bitshuffle")

blosc_info(data_compressed)
```

r\_to\_dtype

*Convert from or to ZARR data types***Description**

Use **ZARR V2.0** data types to convert between R native types and raw data.

**Usage**

```
r_to_dtype(x, dtype, na_value = NA, ...)
```

```
dtype_to_r(x, dtype, na_value = NA, ...)
```

**Arguments**

|          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| x        | Object to be converted                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| dtype    | <p>The data type used for encoding/decoding raw data. The dtype is a code consisting of at least 3 characters. The first character indicates the <b>endianness</b> of the data: '&lt;' (little-endian), '&gt;' (big-endian), or ' ' (endianness not relevant).</p> <p>The second character represents the main data type ('b' boolean (logical), 'i' signed integer, 'u' unsigned integer, 'f' floating point number, 'c' complex number). 'M' is used for date-time objects and 'm' for delta time (see <code>difftime()</code>). 'S' for UTF8 encoded character strings and 'U' for UTF32 encoded character strings.</p> <p>The following characters are numerical indicating the byte size of the data type. For example: dtype = "&lt;f4" means a 32 bit floating point number; dtype = " b1" means an 8 bit logical value. An exception is the main type 'U', where the number indicates the number of characters, where each character is represented by 4 bytes.</p> <p>The main types 'M' and 'm' should always be ended with the time unit between square brackets for storing the date time (difference). A valid code would be "&lt;M8[h].</p> <p>For more details about dtypes see <b>ZARR V2.0</b> or <code>vignette("dtypes")</code>.</p> |
| na_value | <p>When storing raw data, you may want to reserve a value to represent missing values. This is also what R does for NA values. Other software may use different values to represent missing values. Also, some data types have insufficient storage capacity to store R NA values.</p> <p>Therefore, you can use this argument to indicate which value should represent missing values. By default it uses R NA. When set to NULL, missing values are just processed as is, without any further notice or warning.</p> <p>For more details see <code>vignette("dtypes")</code>.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| ...      | Ignored                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

## Details

One of the applications of BLOSC compression is in ZARR, which is used to store n-dimensional structured data. `r_to_dtype()` and `dtype_to_r()` are convenience functions that allows you to convert most common data types to R native types.

R natively only supports `logical()` (actually stored as 32 bit integer in memory), `integer()` (signed 32 bit integers), `numeric()` (64 bit floating points) and `complex()` (real and imaginary component both represented by a 64 bit floating point). R also has some more complex classes, but those are generally derivatives of the aforementioned types.

The functions documented here will attempt to convert raw data to R types (or vice versa). As not all 'dtypes' have an appropriate R type counterpart, some conversions will not be possible directly and will result in an error.

For more details see `vignette("dtypes")`.

## Value

In case of `r_to_dtype()` a vector of encoded raw data is returned. In case of `dtype_to_r()` a vector of an R type (appropriate for the specified dtype) is returned if possible.

## Author(s)

Pepijn de Vries

## Examples

```
## Encode volcano data to 16 bit floating point values
volcano_encoded <-
  r_to_dtype(volcano, dtype = "<f2")

## Decode the volcano format to its original
volcano_reconstructed <-
  dtype_to_r(volcano_encoded, dtype = "<f2")

## The reconstruction is the same as its original:
all(volcano_reconstructed == volcano)

## Encode a numeric sequence with a missing value represented by -999
r_to_dtype(c(1, 2, 3, NA, 4), dtype = "<i2", na_value = -999)
```

# Index

`blosc_compress`, [2](#)  
`blosc_decompress` (`blosc_compress`), [2](#)  
`blosc_info`, [3](#)  
  
`complex()`, [5](#)  
  
`difftime()`, [4](#)  
`dtype_to_r` (`r_to_dtype`), [4](#)  
  
`integer()`, [5](#)  
  
`logical()`, [5](#)  
  
`numeric()`, [5](#)  
  
`r_to_dtype`, [4](#)