

# Package ‘beachmat.hdf5’

April 9, 2025

**Version** 1.5.1

**Date** 2024-11-08

**Title** beachmat bindings for HDF5-backed matrices

**Description** Extends beachmat to support initialization of tatami matrices from HDF5-backed arrays. This allows C++ code in downstream packages to directly call the HDF5 C/C++ library to access array data, without the need for block processing via DelayedArray. Some utilities are also provided for direct creation of an in-memory tatami matrix from a HDF5 file.

**Encoding** UTF-8

**Imports** methods, beachmat, HDF5Array, DelayedArray, Rcpp

**Suggests** testthat, BiocStyle, knitr, rmarkdown, rhdf5, Matrix

**LinkingTo** Rcpp, assorthead, beachmat, Rhdf5lib

**biocViews** DataRepresentation, DataImport, Infrastructure

**License** GPL-3

**NeedsCompilation** yes

**VignetteBuilder** knitr

**SystemRequirements** C++17, GNU make

**RoxygenNote** 7.3.2

**git\_url** <https://git.bioconductor.org/packages/beachmat.hdf5>

**git\_branch** devel

**git\_last\_commit** cd7e813

**git\_last\_commit\_date** 2024-11-08

**Repository** Bioconductor 3.21

**Date/Publication** 2025-04-09

**Author** Aaron Lun [aut, cre]

**Maintainer** Aaron Lun <[infinite.monkeys.with.keyboards@gmail.com](mailto:infinite.monkeys.with.keyboards@gmail.com)>

## Contents

initializeCpp . . . . .	2
initializeOptions . . . . .	3
loadIntoMemory . . . . .	4

<b>Index</b>	<b>6</b>
--------------	----------

---

initializeCpp	<i>Initialize HDF5-backed matrices.</i>
---------------	---

---

## Description

Initialize C++ representations of HDF5-backed matrices based on their **HDF5Array** representations.

## Usage

```
## S4 method for signature 'H5SparseMatrixSeed'
initializeCpp(
  x,
  ...,
  hdf5.cache.size = getAutoBlockSize(),
  hdf5.realize = initializeOptions("realize"),
  memorize = hdf5.realize,
  hdf5.realize.force.integer = initializeOptions("realize.force.integer")
)
```

```
## S4 method for signature 'HDF5ArraySeed'
initializeCpp(
  x,
  ...,
  hdf5.cache.size = getAutoBlockSize(),
  hdf5.realize = initializeOptions("realize"),
  memorize = hdf5.realize,
  hdf5.realize.force.integer = initializeOptions("realize.force.integer")
)
```

## Arguments

x	A <b>HDF5Array</b> seed object.
...	Further arguments, ignored.
hdf5.cache.size	Integer scalar specifying the size of the cache in bytes during data extraction from a <b>HDF5</b> matrix. Larger values reduce disk I/O during random access to the matrix, at the cost of increased memory usage.
hdf5.realize	See the realize option in <a href="#">initializeOptions</a> .

memorize           Deprecated, use `hdf5.realize` instead.  
hdf5.realize.force.integer  
                  See the `force.integer` option in [initializeOptions](#).

### Value

An external pointer that can be used in any **tatami**-compatible function.

### Author(s)

Aaron Lun

### Examples

```
library(HDF5Array)
y <- matrix(runif(1000), ncol=20, nrow=50)
z <- as(y, "HDF5Array")
ptr <- initializeCpp(z)
```

---

initializeOptions           *Options for HDF5 matrices*

---

### Description

Options for initializing HDF5 matrices in [initializeCpp](#).

### Usage

```
initializeOptions(option, value)
```

### Arguments

option           String specifying the name of the option.  
value            Value of the option.

### Details

The following options are supported:

- `realize`, a logical scalar specifying whether to load the matrix data from HDF5 into memory with [loadIntoMemory](#), and then cache it for future calls with [checkMemoryCache](#). This avoids time-consuming disk I/O when performing multiple passes through the matrix, at the expense of increased memory usage.
- `realize.force.integer`, a logical scalar indicating whether values should be coerced into integers when loading the matrix into memory with [loadIntoMemory](#).

**Value**

If value is missing, the current setting of option is returned.

If value is supplied, it is used to set the option, and the previous value of the option is invisibly returned.

**Author(s)**

Aaron Lun

**Examples**

```
initializeOptions("realize.force.integer")
old <- initializeOptions("realize.force.integer", TRUE) # setting to a new value
initializeOptions("realize.force.integer") # new option takes affect
initializeOptions("realize.force.integer", old) # setting it back
```

---

loadIntoMemory	<i>Load a HDF5 matrix into memory</i>
----------------	---------------------------------------

---

**Description**

Load a HDF5-backed matrix into memory as an external pointer to a **tatami**-compatible representation. This differs from the (default) behavior of `initializeCpp`, which only loads slices of the matrix on request.

**Usage**

```
loadIntoMemory(x, force.integer = FALSE)
```

**Arguments**

<code>x</code>	A <b>HDF5Array</b> -derived matrix or seed object.
<code>force.integer</code>	Whether to force floating-point values to be integers to reduce memory consumption.

**Value**

An external pointer that can be used in **tatami**-based functions.

**Author(s)**

Aaron Lun

**Examples**

```
library(HDF5Array)
y <- matrix(runif(1000), ncol=20, nrow=50)
z <- as(y, "HDF5Array")
ptr <- loadIntoMemory(z)
```

# Index

checkMemoryCache, [3](#)

initializeCpp, [2](#), [3](#), [4](#)

initializeCpp,H5SparseMatrixSeed-method  
(initializeCpp), [2](#)

initializeCpp,HDF5ArraySeed-method  
(initializeCpp), [2](#)

initializeOptions, [2](#), [3](#), [3](#)

loadIntoMemory, [3](#), [4](#)