Package ‘gpuR’

May 29, 2019

Type Package

Title GPU Functions for R Objects

Description Provides GPU enabled functions for R objects in a simple and approachable manner. New gpu* and vcl* classes have been provided to wrap typical R objects (e.g. vector, matrix), in both host and device spaces, to mirror typical R syntax without the need to know OpenCL.

Version 2.0.3

Date 2019-05-29

Author Charles Determan Jr.

Maintainer Charles Determan Jr. <cdetermanjr@gmail.com>

VignetteBuilder knitr

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 3.0.2), methods, utils

Imports Rcpp (>= 0.12.15), assertive

LinkingTo Rcpp (>= 0.12.15), RcppEigen (>= 0.3.3.4.0), RViennaCL (>= 1.7.1.7), BH

NeedsCompilation yes

Suggests testthat, knitr

URL http://github.com/cdeterman/gpuR

BugReports http://github.com/cdeterman/gpuR/issues/new

SystemRequirements C++11 (supporting at least std=c++0x), OpenCL shared library (provided by an SDK such as AMD/NVIDIA) and OpenCL headers including the C++ header file (provided by Khronos if not by SDK)

RoxygenNote 6.1.1

Repository CRAN

Date/Publication 2019-05-29 21:10:12 UTC
R topics documented:

gpuR-package ................................................................. 4
Arith,gpuVector,gpuVector-method ................................. 4
as.gpuMatrix ................................................................. 6
as.ggpuVector ......................................................... 7
as.vclVector ................................................................. 8
assert_has_double ...................................................... 8
block ................................................................. 9
cgpuMatrix-class ....................................................... 10
chol.vclMatrix ............................................................. 10
colnames ................................................................. 11
colSums,gpuMatrix-method ............................................ 12
colSums.vclMatrix-method ............................................. 13
Compare.vector,gpuVector-method .................................... 13
cov.vclMatrix,missing,missing,missing-method .................. 14
crossprod,gpuMatrix,missing-method .................................. 15
currentContext ......................................................... 16
currentDevice ............................................................. 17
currentPlatform .......................................................... 17
custom_opencl ............................................................ 18
cvclMatrix-class ......................................................... 18
deprecated ................................................................. 19
det,vclMatrix-method ................................................... 20
detectCPUs ................................................................. 20
detectGPUs ................................................................. 21
detectPlatforms ........................................................... 21
deviceHasDouble .......................................................... 22
deviceType ................................................................. 22
dgpuMatrix-class .......................................................... 23
dgpuVector-class .......................................................... 23
diag.vclMatrix-method .................................................. 24
dim,vclMatrix-method .................................................... 25
distance ................................................................. 25
dvclMatrix-class .......................................................... 26
dvclVector-class .......................................................... 27
eigen,gpuMatrix-method ................................................. 27
fgpuMatrix-class .......................................................... 28
fgpuVector-class .......................................................... 29
fvclMatrix-class .......................................................... 29
fvclVector-class .......................................................... 30
gpuInfo ................................................................. 30
gpuMatrix ................................................................. 31
gpuMatrix-class .......................................................... 32
gpuVector ................................................................. 33
gpuVector-class .......................................................... 34
has_cpu_skip ................................................................. 34
has_double_skip ............................................................. 35
**gpuR-package**

*GPU functions for R Objects*

**Description**

This package was developed to provide simple to use R functions that leverage the power of GPU’s but also retain a format familiar to the R user. There are a hand full of other R packages that provide some GPU functionality but nearly all rely on a CUDA backend thereby restricting the user to NVIDIA GPU hardware. In the spirit of being as broadly applicable as possible, this GPU code herein relies upon OpenCL via the ViennaCL library.

OpenCL, in contrast to CUDA, is open source and can be used across different graphics cards (e.g. NVIDIA, AMD, Intel). This package removes the complex code needed for GPU computing and provides easier to use functions to apply on R objects.

<table>
<thead>
<tr>
<th>Package:</th>
<th>gpuR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0.0</td>
</tr>
<tr>
<td>Date:</td>
<td>2015-03-31</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-3</td>
</tr>
<tr>
<td>Copyright:</td>
<td>(c) 2015 Charles E. Determan Jr.</td>
</tr>
<tr>
<td>URL:</td>
<td><a href="http://www.github.com/cdeterman/gpuR">http://www.github.com/cdeterman/gpuR</a></td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
</tbody>
</table>

**Note**

There are other packages that also provide wrappers for OpenCL programming including OpenCL by Simon Urbanek and ROpenCL at Open Analytics by Willem Ligtenberg. Both of these packages provide the R user an interface to directly call OpenCL functions. This package, however, hides these functions so the user does not require any knowledge of OpenCL to begin using their GPU. The idea behind this package is to provide a means to begin using existing algorithms without the need to write extensive amounts of C/C++/OpenCL code.

**Author(s)**

Charles Determan <cdetermanjr@gmail.com>

Maintainer: Charles Determan <cdetermanjr@gmail.com>

**Description**

Methods for the base Arith methods S4groupGeneric
Usage

```r
## S4 method for signature 'gpuVector,gpuVector'
Arith(e1, e2)

## S4 method for signature 'numeric,gpuVector'
Arith(e1, e2)

## S4 method for signature 'gpuVector,numeric'
Arith(e1, e2)

## S4 method for signature 'gpuVector,missing'
Arith(e1, e2)

## S4 method for signature 'gpuVector,gpuMatrix'
Arith(e1, e2)

## S4 method for signature 'vclMatrix,vclMatrix'
Arith(e1, e2)

## S4 method for signature 'vclMatrix,matrix'
Arith(e1, e2)

## S4 method for signature 'matrix,vclMatrix'
Arith(e1, e2)

## S4 method for signature 'vclMatrix,numeric'
Arith(e1, e2)

## S4 method for signature 'numeric,vclMatrix'
Arith(e1, e2)

## S4 method for signature 'vclMatrix,missing'
Arith(e1, e2)

## S4 method for signature 'vclMatrix,vclVector'
Arith(e1, e2)

## S4 method for signature 'vclVector,vclVector'
Arith(e1, e2)

## S4 method for signature 'numeric,vclVector'
Arith(e1, e2)

## S4 method for signature 'vclVector,numeric'
Arith(e1, e2)

## S4 method for signature 'vclVector,missing'
Arith(e1, e2)
```
as.gpuMatrix

Arguments

- `e1`: A gpuR object
- `e2`: A gpuR object

Value

- A gpuR object

Author(s)

Charles Determan Jr.

Description

Convert object to a gpuMatrix

Usage

as.gpuMatrix(object, type)
as.gpuVector

Arguments

object       An object that is or can be converted to a matrix object

A character string specifying the type of gpuMatrix. Default is NULL where

Value

A gpuMatrix object

Author(s)

Charles Determan Jr.

as.gpuVector  

Convert object to a gpuVector

Description

Construct a gpuVector of a class that inherits from gpuVector

Usage

as.gpuVector(object, type)

## S4 method for signature 'vector'
as.gpuVector(object, type = NULL)

Arguments

object       An object that is or can be converted to a vector object

A character string specifying the type of gpuVector. Default is NULL where
ty type is inherited from the source data type.

Value

A gpuVector object

Author(s)

Charles Determan Jr.
as.vclVector  

*Convert object to a vclVector*

**Description**

Construct a vclVector of a class that inherits from vclVector

**Usage**

```r
as.vclVector(object, type = NULL, ...)
```

```r
## S4 method for signature 'vector'
as.vclVector(object, type = NULL)
```

```r
## S4 method for signature 'vclMatrix'
as.vclVector(object, type = NULL, shared = FALSE)
```

**Arguments**

- **object**: An object that is or can be converted to a vector object
- **type**: A character string specifying the type of vclVector. Default is NULL where type is inherited from the source data type.
- **...**: Additional arguments to as.vclVector methods
- **shared**: Logical indicating if memory should be shared with x

**Value**

A vclVector object

**Author(s)**

Charles Determan Jr.

---

assert_has_double  

*Does device have 'double' support?*

**Description**

Function to query if device (identified by index) supports double precision

**Usage**

```r
assert_has_double(device_idx, context_idx,
    severity = getOption("assertive.severity", "stop"))
```
**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>device_idx</td>
<td>An integer indicating which device to query</td>
</tr>
<tr>
<td>context_idx</td>
<td>An integer indicating which context to query</td>
</tr>
<tr>
<td>severity</td>
<td>How severe should the consequences of the assertion be?</td>
</tr>
</tbody>
</table>

**Value**

Returns nothing but throws an error if device does not support double precision.

**Author(s)**

Charles Determan Jr.

**See Also**

- `deviceHasDouble`

---

**Description**

This doesn’t create a copy, it provides a child class that points to a contiguous submatrix of a `gpuMatrix` or `vclMatrix`. Non-contiguous blocks are currently not supported.

**Usage**

```r
block(object, rowStart, rowEnd, colStart, colEnd)
```

```r
## S4 method for signature 'vclMatrix,integer,integer,integer,integer'
block(object, rowStart, rowEnd, colStart, colEnd)
```

```r
## S4 method for signature 'gpuMatrix,integer,integer,integer,integer'
block(object, rowStart, rowEnd, colStart, colEnd)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>A <code>gpuMatrix</code> or <code>vclMatrix</code> object</td>
</tr>
<tr>
<td>rowStart</td>
<td>An integer indicating the first row of block</td>
</tr>
<tr>
<td>rowEnd</td>
<td>An integer indicating the last row of block</td>
</tr>
<tr>
<td>colStart</td>
<td>An integer indicating the first column of block</td>
</tr>
<tr>
<td>colEnd</td>
<td>An integer indicating the last column of block</td>
</tr>
</tbody>
</table>
chol.vclMatrix

Details
This function allows a user to create a gpuR matrix object that references a continuous subset of columns and rows of another gpuR matrix object without a copy.

NOTE - this means that altering values in a matrix block object will alter values in the source matrix.

Value
A gpuMatrixBlock or vclMatrixBlock object

Author(s)
Charles Determan Jr.

cgpuMatrix-class  cgpuMatrix Class

Description
An complex float type matrix in the S4 gpuMatrix representation.

Slots
address: Pointer to a complex float matrix.

Author(s)
Charles Determan Jr.

See Also
gpuMatrix-class, igpuMatrix-class, dgpuMatrix-class

chol.vclMatrix  Cholesky Decomposition of a gpuR matrix

Description
Compute the Choleski factorization of a real symmetric positive-definite square matrix.

Usage
## S3 method for class 'vclMatrix'
chol(x, ...)

## S3 method for class 'gpuMatrix'
chol(x, ...)
Arguments
   x       A symmetric, positive-definite gpuR matrix object.
   ...    arguments to be passed to or from methods

Value
   Default - the upper triangular factor of the Choleski decomposition, i.e. the matrix R such that R'R = x.

Note
   This an S3 generic of chol. The default continues to point to the default base function.
   No pivoting is used.
   The argument upper is additionally accepted representing a boolean which will indicate if the upper or lower (FALSE) triangle should be solved.

Author(s)
   Charles Determan Jr.

See Also
   chol

---

colnames    Row and Column Names

Description
   Retrieve or set the row or column names of a gpuR matrix object

Usage
   colnames(x, do.NULL, prefix)

   ## Default S3 method:
colnames(x, do.NULL = TRUE, prefix = "col")

   ## S3 method for class 'gpuMatrix'
colnames(x, ...)

   ## S4 replacement method for signature 'gpuMatrix'
colnames(x) <- value

   ## S3 method for class 'vclMatrix'
colnames(x, ...)

## S4 replacement method for signature 'vclMatrix'

`colnames(x) <- value`

### Arguments

- `x`: A gpuR matrix object
- `do.NULL`: logical. If FALSE names are NULL, names are created. (not currently used)
- `prefix`: for create names. (not currently used)
- `...`: Additional arguments
- `value`: A character vector to assign as row/column names

### Description

Row and column sums and of gpuMatrix objects

### Usage

```r
# S4 method for signature 'gpuMatrix'
colSums(x)
```

```r
# S4 method for signature 'gpuMatrix'
rowSums(x)
```

```r
# S4 method for signature 'gpuMatrix'
colMeans(x)
```

```r
# S4 method for signature 'gpuMatrix'
rowMeans(x)
```

### Arguments

- `x`: A gpuMatrix object

### Value

A gpuVector object

### Author(s)

Charles Determan Jr.
**colSums,vclMatrix-method**

*Row and Column Sums and Means of vclMatrix*

**Description**

Row and column sums and of vclMatrix objects

**Usage**

```r
## S4 method for signature 'vclMatrix'
colSums(x)

## S4 method for signature 'vclMatrix'
rowSums(x)

## S4 method for signature 'vclMatrix'
colMeans(x)

## S4 method for signature 'vclMatrix'
rowMeans(x)
```

**Arguments**

- `x` A vclMatrix object

**Value**

A gpuVector object

**Author(s)**

Charles Determan Jr.

---

**Compare,vector,gpuVector-method**

*Compare vector and gpuVector elements*

**Description**

Methods for comparison operators
Usage

## S4 method for signature 'vector, gpuVector'

Compare(e1, e2)

## S4 method for signature 'gpuVector, vector'

Compare(e1, e2)

Arguments

e1 A vector/gpuVector object
e2 A vector/gpuVector object

Value

A logical vector

Author(s)

Charles Determan Jr.

cov, vclMatrix, missing, missing, missing-method

Covariance (gpuR)

Description

Compute covariance values

Usage

## S4 method for signature 'vclMatrix, missing, missing, missing'

cov(x, y = NULL,
       use = NULL, method = "pearson")

## S4 method for signature 'vclMatrix, vclMatrix, missing, missing'

cov(x, y = NULL,
       use = NULL, method = "pearson")

## S4 method for signature 'vclMatrix, missing, missing, character'

cov(x, y = NULL,
       use = NULL, method = "pearson")

## S4 method for signature 'vclMatrix, vclMatrix, missing, character'

cov(x, y = NULL,
       use = NULL, method = "pearson")

## S4 method for signature 'gpuMatrix, missing, missing, missing'

Usage

## S4 method for signature 'vector, gpuVector'

Compare(e1, e2)

## S4 method for signature 'gpuVector, vector'

Compare(e1, e2)
cov(x, y = NULL,
    use = NULL, method = "pearson")

## S4 method for signature 'gpuMatrix, gpuMatrix, missing, missing'
cov(x, y = NULL,
    use = NULL, method = "pearson")

## S4 method for signature 'gpuMatrix, missing, missing, character'
cov(x, y = NULL,
    use = NULL, method = "pearson")

## S4 method for signature 'gpuMatrix, gpuMatrix, missing, character'
cov(x, y = NULL,
    use = NULL, method = "pearson")

Arguments

x A gpuR object
y A gpuR object
use Not used
method Character string indicating with covariance to be computed.

Value

A gpuMatrix/vclMatrix containing the symmetric covariance values.

Author(s)

Charles Determan Jr.

crossprod, gpuMatrix, missing-method


gpuMatrix Crossproduct

description

Return the matrix cross-product of two conformable matrices using a GPU. This is equivalent to t(x) or x device and host is required.

Usage

## S4 method for signature 'gpuMatrix, missing'
crossprod(x, y)

## S4 method for signature 'gpuMatrix, gpuMatrix'
crossprod(x, y)
currentContext

## S4 method for signature 'gpuMatrix,matrix'
crossprod(x, y)

## S4 method for signature 'matrix,gpuMatrix'
crossprod(x, y)

## S4 method for signature 'gpuMatrix,missing'
tcrossprod(x, y)

## S4 method for signature 'gpuMatrix,gpuMatrix'
tcrossprod(x, y)

## S4 method for signature 'matrix,gpuMatrix'
tcrossprod(x, y)

## S4 method for signature 'gpuMatrix,matrix'
tcrossprod(x, y)

### Arguments

- **x**: A `gpuMatrix`
- **y**: A `gpuMatrix`

### Value

A `gpuMatrix`

### Author(s)

Charles Determan Jr.

---

<table>
<thead>
<tr>
<th>currentContext</th>
<th>Current Context</th>
</tr>
</thead>
</table>

### Description

Get current context index

### Usage

`currentContext()`

### Value

An integer reflecting the context listed in `listContexts`
**currentDevice**

---

### Current Device Information

**Description**

Check current device information

**Usage**

`currentDevice()`

**Value**

- list containing
  - `device` Character string of device name
  - `device_index` Integer identifying device
  - `device_type` Character string identifying device type (e.g. gpu)

---

**currentPlatform**

---

### Return Current Platform

**Description**

Find out which platform is currently in use

**Usage**

`currentPlatform()`

**Value**

- `platform` Name of the current platform
- `platform_index` Index of current platform

**See Also**

`detectPlatforms`
custom_opencl | Custom OpenCL Kernels

**Description**

Compile a custom function using a user provided OpenCL kernel

**Usage**

custom_opencl(kernel, cl_args, type)

**Arguments**

- **kernel**: path to OpenCL kernel file
- **cl_args**: A data.frame that contains argument definitions. Provided by `setup_opencl`
- **type**: The precision on which the kernel is compiled. Options include "int", "float", and "double"

---

**cvclMatrix-class | cvclMatrix Class**

**Description**

An complex float type matrix in the S4 `cvclMatrix` representation.

**Slots**

- **address**: Pointer to a complex float type matrix

**Author(s)**

Charles Determan Jr.

**See Also**

`cvclMatrix-class, ivclMatrix-class, fvclMatrix-class`
**deepcopy**

*Copy a "gpuR" object*

**Description**

This is needed to make a duplicate of a gpuR object

**Usage**

```r
deepcopy(object, ...)
```

```r
## S4 method for signature 'gpuVector'
deepcopy(object)
```

```r
## S4 method for signature 'vclMatrix'
deepcopy(object, source = FALSE)
```

```r
## S4 method for signature 'vclVector'
deepcopy(object)
```

```r
## S4 method for signature 'gpuMatrix'
deepcopy(object)
```

**Arguments**

- `object`: A gpuR object
- `...`: Additional arguments
- `source`: A boolean indicating if source matrix should be copied (only relevant for 'block' and 'slice' objects).

**Details**

This is needed to make a duplicate of a gpuR object (i.e. gpuMatrix, gpuVector, vclMatrix, vclVector because the traditional syntax would only copy the pointer of the object.

**Value**

A gpuR object

**Author(s)**

Charles Determan Jr.

**See Also**

- block
\textbf{det, vclMatrix-method  \hspace{1cm} Calculate Determinant of a Matrix on GPU}

\textbf{Description}

\hspace{1cm} \texttt{det} calculates the determinant of a matrix.

\textbf{Usage}

\hspace{1cm} \texttt{## S4 method for signature 'vclMatrix'}
\hspace{1cm} \texttt{det(x)}

\hspace{1cm} \texttt{## S4 method for signature 'gpuMatrix'}
\hspace{1cm} \texttt{det(x)}

\textbf{Arguments}

\hspace{1cm} \texttt{x} \hspace{0.5cm} A \texttt{gpuR} matrix object

\textbf{Value}

\hspace{1cm} The determinant of \texttt{x}

\textbf{Note}

\hspace{1cm} This function uses an LU decomposition and the \texttt{det} function is simply a wrapper returning the determinant product

\textbf{Author(s)}

\hspace{1cm} Charles Determan Jr.

\textbf{detectCPUs  \hspace{1cm} Detect Available OpenCL enabled CPUs}

\textbf{Description}

\hspace{1cm} Find out how many CPUs available

\textbf{Usage}

\hspace{1cm} \texttt{detectCPUs(platform_idx = NULL)}

\textbf{Arguments}

\hspace{1cm} \texttt{platform_idx} \hspace{0.5cm} An integer value indicating which platform to query. If \texttt{NULL} it will iterate over all platforms and sum results


detectGPUs

Value
   An integer representing the number of available CPUs

See Also
   detectPlatforms detectGPUs

Detect Available GPUs

Description
   Find out how many GPUs available

Usage
   detectGPUs(platform_idx = NULL)

Arguments
   platform_idx   An integer value indicating which platform to query. If NULL it will iterate over all platforms and sum results

Value
   An integer representing the number of available GPUs

See Also
   detectPlatforms

Detect Number of Platforms

Description
   Find out how many OpenCL enabled platforms are available.

Usage
   detectPlatforms()

Value
   An integer value representing the number of platforms available.

See Also
   detectGPUs
deviceHasDouble  
*Check GPU double precision support*

**Description**

This function checks the GPU device extensions for the variable cl_khr_fp64 which means the device supports double precision.

**Usage**

```c
deviceHasDouble(gpu_idx = currentDevice()
context_idx = currentContext())
```

**Arguments**

- `gpu_idx`  
  An integer value indicating which gpu to query.
- `context_idx`  
  An integer value indicating which context to query.

**Value**

A boolean designating whether the device supports double precision

**See Also**

- `gpuInfo`

deviceType  
*Check device type*

**Description**

Check what type a device is given platform and device indices

**Usage**

```c
deviceType(device_idx = NULL, context_idx = currentContext())
```

**Arguments**

- `device_idx`  
  An integer value indicating which device to query.
- `context_idx`  
  An integer value indicating which context to query.

**Value**

A character string indicating the device type
**dgpuMatrix-class**  

dgpuMatrix-class  

**Description**  

A double type matrix in the S4 gpuMatrix representation.

**Slots**  

- **address**: Pointer to a double type matrix

**Author(s)**

Charles Determan Jr.

**See Also**

_gpuMatrix-class, igpuMatrix-class, fgpuMatrix-class_

---

**dgpuVector-class**  

dgpuVector-class  

**Description**  

An double vector in the S4 gpuVector representation.

**Slots**  

- **address**: Pointer to a double typed vector

**Author(s)**

Charles Determan Jr.

**See Also**

_gpuVector-class_
Description

Extract or replace the diagonal of a matrix

Usage

```r
## S4 method for signature 'vclMatrix'
diag(x)
```

```r
## S4 replacement method for signature 'vclMatrix,vclVector'
diag(x) <- value
```

```r
## S4 method for signature 'gpuMatrix'
diag(x)
```

```r
## S4 replacement method for signature 'gpuMatrix,gpuVector'
diag(x) <- value
```

Arguments

- `x` A gpuR matrix object
- `value` A vector object (gpuR)

Value

A gpuR vector object of the matrix diagonal of `x`. The replacement form returns nothing as it replaces the diagonal of `x`.

Note

If an identity matrix is desired, please see `identity_matrix`.

Author(s)

Charles Determan Jr.

See Also

`identity_matrix`
dim, vclMatrix-method

Description
Retrieve dimension of object

Usage
## S4 method for signature 'vclMatrix'
dim(x)

## S4 method for signature 'gpuMatrix'
dim(x)

Arguments
x A gpuMatrix/vclMatrix object

Value
A length 2 vector of the number of rows and columns respectively.

Author(s)
Charles Determan Jr.

distance

GPU Distance Matrix Computations

Description
This function computes and returns the distance matrix computed by using the specified distance measure to compute the distances between the rows of a data matrix.

Usage
distance(x, y, method = "euclidean")

## S4 method for signature 'vclMatrix'
dist(x, method = "euclidean", diag = FALSE,
  upper = FALSE, p = 2)

## S4 method for signature 'vclMatrix, vclMatrix'
distance(x, y, method = "euclidean")
## dvclMatrix-method

```r
## S4 method for signature 'gpuMatrix'
dist(x, method = "euclidean", diag = FALSE,
     upper = FALSE, p = 2)

## S4 method for signature 'gpuMatrix,gpuMatrix'
distance(x, y, method = "euclidean")
```

### Arguments

- **x**: A `gpuMatrix` or `vclMatrix` object
- **y**: A `gpuMatrix` or `vclMatrix` object
- **method**: the distance measure to be used. This must be one of "euclidean" or "sqEuclidean".
- **diag**: logical value indicating whether the diagonal of the distance matrix should be printed
- **upper**: logical value indicating whether the upper triangle of the distance matrix
- **p**: The power of the Minkowski distance (not currently used)

### Value

A `gpuMatrix/vclMatrix` containing the corresponding distances

---

### dvclMatrix-class

**dvclMatrix Class**

### Description

An integer type matrix in the S4 `vclMatrix` representation.

### Slots

- **address**: Pointer to a double type matrix

### Author(s)

Charles Determan Jr.

### See Also

`vclMatrix-class`, `ivclMatrix-class`, `fvclMatrix-class`
**dvclVector-class**

**dvclVector Class**

**Description**

An double vector in the S4 vclVector representation.

**Slots**

- address: Pointer to a double typed vector

**Author(s)**

Charles Determan Jr.

**See Also**

- `vclVector-class`

---

**eigen,gpuMatrix-method**

**gpuMatrix Eigen Decomposition**

**Description**

Computes the eigenvalues and eigenvectors for gpuMatrix objects.

**Usage**

```r
## S4 method for signature 'gpuMatrix'
eigen(x, symmetric, only.values = FALSE,
   EISPACK = FALSE)

## S4 method for signature 'vclMatrix'
eigen(x, symmetric, only.values = FALSE,
   EISPACK = FALSE)
```

**Arguments**

- `x` A gpuMatrix object
- `symmetric` logical indication if matrix is assumed to be symmetric. If not specified or FALSE, the matrix is inspected for symmetry
- `only.values` if TRUE, returns only eigenvalues (internals still currently calculate both regardless)
- `EISPACK` logical. Defunct and ignored
Details
This function currently implements the qr_method function from the ViennaCL library. As such, non-symmetric matrices are not supported given that OpenCL does not have a 'complex' data type.
Neither the eigenvalues nor the eigenvectors are sorted as done in the base R eigen method.

Value
values A gpuVector containing the unsorted eigenvalues of x.
rectors A gpuMatrix containing the unsorted eigenvectors of x

Note
The sign's may be different on some of the eigenvector elements. As noted in the base eigen documentation:
Recall that the eigenvectors are only defined up to a constant: even when the length is specified they are still only defined up to a scalar of modulus one (the sign for real matrices).
Therefore, although the signs may be different, the results are functionally equivalent

fgpuMatrix-class  fgpuMatrix Class

Description
A float type matrix in the S4 gpuMatrix representation.

Slots
address: Pointer to a float matrix.

Author(s)
Charles Determan Jr.

See Also
gpuMatrix-class, igpuMatrix-class, dgpuMatrix-class
fgpuVector-class  

fgpuVector-class  

---

**fgpuVector-class**  

**fgpuVector Class**

---

**Description**

An float vector in the S4 gpuVector representation.

**Slots**

- **address**: Pointer to a float typed vector

**Author(s)**

Charles Determan Jr.

**See Also**

gpuVector-class

---

fvclMatrix-class  

fvclMatrix-class  

---

**fvclMatrix-class**  

**fvclMatrix Class**

---

**Description**

An integer type matrix in the S4 vclMatrix representation.

**Slots**

- **address**: Pointer to a float matrix.

**Author(s)**

Charles Determan Jr.

**See Also**

vclMatrix-class, ivclMatrix-class, dvclMatrix-class
**fvclVector-class  fvclVector Class**

**Description**
An float vector in the S4 vclVector representation.

**Slots**
- **address**: Pointer to a float typed vector

**Author(s)**
Charles Determan Jr.

**See Also**
- `vclVector-class`

---

**gpuInfo  Device Information**

**Description**
Get basic information about selected device (e.g. GPU)

**Usage**
- `gpuInfo(device_idx = NULL, context_idx = currentContext())`
- `cpuInfo(device_idx = NULL, context_idx = currentContext())`

**Arguments**
- **device_idx**: An integer value indicating which device to query.
- **context_idx**: An integer value indicating which context to query.

**Value**
- **deviceName**: Device Name
- **deviceVendor**: Device Vendor
- **numberOfCores**: Number of Computing Units (which execute the work groups)
- **maxWorkGroupSize**: Maximum number of work items per group
- **maxWorkItemDim**: Number of dimensions
GPU Matrix

maxWorkItemSizes  Maximum number of works items per dimension
deviceMemory     Global amount of memory (bytes)
clockFreq        Maximum configured clock frequency of the device in MHz
localMem         Maximum amount of local memory for each work group (bytes)
maxAllocatableMem Maximum amount of memory in a single piece (bytes)
available        Whether the device is available
deviceExtensions OpenCL device extensions available
double_support   Logical value if double type supported

Author(s)
Charles Determan Jr.

See Also
detectPlatforms detectGPUs detectCPUs cpuInfo

gpuMatrix

Construct a gpuMatrix

Description
Construct a gpuMatrix of a class that inherits from gpuMatrix

Usage

gpuMatrix(data = NA, nrow = NA, ncol = NA, type = NULL, ...)

# S4 method for signature 'matrix'
gpuMatrix(data, type = NULL, ctx_id = NULL)

# S4 method for signature 'missing'
gpuMatrix(data, nrow = NA, ncol = NA,
          type = NULL, ctx_id = NULL)

# S4 method for signature 'numeric'
gpuMatrix(data, nrow, ncol, type = NULL, ctx_id = NULL)

# S4 method for signature 'integer'
gpuMatrix(data, nrow, ncol, type = NULL, ctx_id = NULL)
Arguments

- **data**: An object that is or can be converted to a matrix object
- **nrow**: An integer specifying the number of rows
- **ncol**: An integer specifying the number of columns
- **type**: A character string specifying the type of gpuMatrix. Default is NULL where type is inherited from the source data type.
- **...**: Additional method to pass to gpuMatrix methods
- **ctx_id**: An integer specifying the object’s context

Value

A gpuMatrix object

Author(s)

Charles Determan Jr.

---

**gpuMatrix-class**

**gpuMatrix Class**

Description

This is the 'mother' class for all gpuMatrix objects. It is essentially a wrapper for a basic R matrix (possibly to be improved). All other gpuMatrix classes inherit from this class but there are no current circumstances where this class is used directly.

There are multiple child classes that correspond to the particular data type contained. These include igpuMatrix, fgpumatrix, and dgpumatrix corresponding to integer, float, and double data types respectively.

Slots

Common to all gpuMatrix objects in the package

- **address**: Pointer to data matrix
- **.context_index**: Integer index of OpenCL contexts
- **.platform_index**: Integer index of OpenCL platforms
- **.platform**: Name of OpenCL platform
- **.device_index**: Integer index of active device
- **.device**: Name of active device

Note

R does not contain a native float type. As such, the matrix data within a fgpuMatrix-class will be represented as double but downcast when any gpuMatrix methods are used.

May also remove the type slot
gpuVector  

Author(s)  
Charles Determan Jr.

See Also  
igpuMatrix-class, fgpuMatrix-class, dgpuMatrix-class

---

**gpuVector**  

**Construct a gpuVector**

---

**Description**  
Construct a gpuVector of a class that inherits from gpuVector

**Usage**  
gpuVector(data, length, type = NULL, ...)  

```r  
## S4 method for signature 'vector,missing'  
gpuVector(data, type = NULL, ctx_id = NULL)  

## S4 method for signature 'missingOrNULL,ANY'  
gpuVector(data, length, type = NULL,  
          ctx_id = NULL)  

## S4 method for signature 'numeric,numericOrInt'  
gpuVector(data, length, type = NULL,  
          ctx_id = NULL)  
```

**Arguments**  
- **data**  
  An object that is or can be converted to a vector

- **length**  
  A non-negative integer specifying the desired length.

- **type**  
  A character string specifying the type of gpuVector. Default is NULL where type is inherited from the source data type.

- ...  
  Additional method to pass to gpuVector methods

- **ctx_id**  
  An integer specifying the object’s context

**Value**  
A gpuVector object

**Author(s)**  
Charles Determan Jr.
gpuVector-class  

gpuVector Class

Description
This is the 'mother' class for all gpuVector objects. All other gpuVector classes inherit from this class but there are no current circumstances where this class is used directly.

There are multiple child classes that correspond to the particular data type contained. These include igpuVector.

Slots
Common to all vclMatrix objects in the package

address: Pointer to data matrix

.context_index: Integer index of OpenCL contexts

.platform_index: Integer index of OpenCL platforms

.platform: Name of OpenCL platform

.device_index: Integer index of active device

.device: Name of active device

Author(s)
Charles Determan Jr.

See Also
igpuVector-class

has_cpu_skip  

Skip test for CPUs

Description
Function to skip test that tests if no valid CPU's are detected

Usage

has_cpu_skip()
**has_double_skip**  
*Skip test for GPU double precision*

**Description**
Function to skip test that tests if the detected GPU doesn’t support double precision

**Usage**
```r
has_double_skip()
```

---

**has_gpu_skip**  
*Skip test for GPUs*

**Description**
Function to skip test that tests if no valid GPUs are detected

**Usage**
```r
has_gpu_skip()
```

---

**has_multiple_double_skip**  
*Skip test for multiple GPUs with double precision*

**Description**
Function to skip test that tests if there aren’t multiple detected GPU with double precision

**Usage**
```r
has_multiple_double_skip()
```

---

**has_multiple_gpu_skip**  
*Skip test in less than 2 GPUs*

**Description**
Function to skip test that tests if less than 2 valid GPUs are detected

**Usage**
```r
has_multiple_gpu_skip()
```
### identity_matrix

**Identity Matrix on Device**

**Description**

Creates an identity matrix directly on the current device (e.g. GPU)

**Usage**

```r
identity_matrix(x, type = NULL)
```

**Arguments**

- `x` A numeric value indicating the order of the identity matrix
- `type` A character string specifying the type of `gpuMatrix`. Default is derived from `getOption("gpuR.default.type")`.

**Value**

A `vclMatrix` object

**Note**

This function was only created for `vclMatrix` objects as the copy from CPU to `gpuMatrix` is trivial using the base `diag` function.

**Author(s)**

Charles Determan Jr.

---

### igpuMatrix-class

**igpuMatrix Class**

**Description**

An integer type matrix in the S4 `gpuMatrix` representation.

**Slots**

- `address`: Pointer to a integer typed matrix

**Author(s)**

Charles Determan Jr.

**See Also**

`gpuMatrix-class, igpuMatrix-class, dgpuMatrix-class`
igpuVector-class

Description
An integer vector in the S4 gpuVector representation.

Slots
  address: An integer vector object

Author(s)
Charles Determan Jr.

See Also
gpuVector-class

inplace

Inplace Function Wrapper

Description
Applies the provided function in-place on the first object passed

Usage
inplace(f, x, y)

## S4 method for signature 'function,vclMatrix,vclMatrix'
inplace(f, x, y)

## S4 method for signature 'function,vclMatrix,missing'
inplace(f, x, y)

## S4 method for signature 'function,numeric,vclMatrix'
inplace(f, x, y)

## S4 method for signature 'function,vclMatrix,numeric'
inplace(f, x, y)

## S4 method for signature 'function,gpuMatrix,gpuMatrix'
inplace(f, x, y)

## S4 method for signature 'function,gpuMatrix,missing'
inplace(f, x, y)

## S4 method for signature 'function',numeric,gpuMatrix'
inplace(f, x, y)

## S4 method for signature 'function',gpuMatrix,numeric'
inplace(f, x, y)

## S4 method for signature 'function',vclVector,vclVector'
inplace(f, x, y)

## S4 method for signature 'function',vclVector,missing'
inplace(f, x, y)

## S4 method for signature 'function',vclVector,numeric'
inplace(f, x, y)

## S4 method for signature 'function',numeric,vclVector'
inplace(f, x, y)

## S4 method for signature 'function',gpuVector,gpuVector'
inplace(f, x, y)

## S4 method for signature 'function',gpuVector,missing'
inplace(f, x, y)

## S4 method for signature 'function',gpuVector,numeric'
inplace(f, x, y)

## S4 method for signature 'function',numeric,gpuVector'
inplace(f, x, y)

**Arguments**

- **f**  A function
- **x**  A gpuR object
- **y**  A gpuR object

**Value**

No return, result applied in-place

**Author(s)**

Charles Determan Jr.
ivclMatrix-class

ivclMatrix-class

**ivclMatrix Class**

**Description**

An integer type matrix in the S4 vclMatrix representation.

**Slots**

- **address**: Pointer to an integer typed matrix

**Author(s)**

Charles Determan Jr.

**See Also**

vclMatrix-class, ivclMatrix-class, dvclMatrix-class

ivclVector-class

**ivclVector Class**

**Description**

An integer vector in the S4 vclVector representation.

**Slots**

- **address**: An integer vector object

**Author(s)**

Charles Determan Jr.

**See Also**

vclVector-class
length, gpuVector-method

gpuMatrix/vclMatrix length method

Description
Retrieve number of elements in object

Usage

```r
## S4 method for signature 'gpuVector'
length(x)

## S4 method for signature 'vclMatrix'
length(x)

## S4 method for signature 'vclVector'
length(x)

## S4 method for signature 'gpuMatrix'
length(x)
```

Arguments

- `x`: A gpuMatrix/vclMatrix object

Value

A numeric value

Author(s)

Charles Determan Jr.

listContexts

Available OpenCL Contexts

Description
Provide a data.frame of available OpenCL contexts and associated information.

Usage

```r
listContexts()
```
log.gpuVector-method

Value
data.frame containing the following fields

context     Integer identifying context
platform    Character string listing OpenCL platform
platform_index Integer identifying platform
device      Character string listing device name
device_index Integer identifying device
device_type  Character string labeling device (e.g. gpu)

log, gpuVector-method  gpuR Logarithms and Exponentials

Description
log computes logarithms, by default natural logarithms and log10 computes common (i.e. base 10) logarithms. The general form log(x, base) computes logarithms with base base.
exp computes the exponential function.

Usage

## S4 method for signature 'gpuVector'
log(x, base = NULL)

## S4 method for signature 'vclMatrix'
log(x, base = NULL)

## S4 method for signature 'vclVector'
log(x, base = NULL)

## S4 method for signature 'gpuMatrix'
log(x, base = NULL)

Arguments

x     A gpuR object
base  A positive number (complex not currently supported by OpenCL): the base with respect to which logarithms are computed. Defaults to the natural log.

Value

A gpuR object of the same class as x
Math, gpuVector-method  

**gpuR Math methods**

**Description**

Methods for the base Math methods `S4groupGeneric`

**Usage**

```r
## S4 method for signature 'gpuVector'
Math(x)

## S4 method for signature 'vclMatrix'
Math(x)

## S4 method for signature 'vclVector'
Math(x)

## S4 method for signature 'gpuMatrix'
Math(x)
```

**Arguments**

- `x`  
  A gpuR object

**Details**

Currently implemented methods include:


**Value**

A gpuR object

**Author(s)**

Charles Determan Jr.
Compute the Norm of a Matrix

Description
Computes a matrix norm of x. The norm can be the one \(\|O\|\) norm, the infinity \(\|I\|\) norm, the Frobenius \(\|F\|\) norm, the maximum modulus \(\|M\|\) among elements of a matrix, or the "spectral" or "2"-norm, as determined by the value of type.

Usage
```r
## S4 method for signature 'vclMatrix,character'
norm(x, type)

## S4 method for signature 'vclMatrix,missing'
norm(x, type)

## S4 method for signature 'gpuMatrix,character'
norm(x, type)

## S4 method for signature 'gpuMatrix,missing'
norm(x, type)

## S4 method for signature 'ANY,missing'
norm(x, type)

## S4 method for signature 'ANY,character'
norm(x, type)
```

Arguments
- **x**: A gpuR matrix object
- **type**: character string, specifying the type of matrix norm to be computed.

Value
The matrix norm, a non-negative number

Author(s)
Charles Determan Jr.

See Also
- norm
nrow, vclMatrix-method  The Number of Rows/Columns of a gpuR matrix

Description

nrow and ncol return the number of rows or columns present in x respectively.

Usage

```r
## S4 method for signature 'vclMatrix'
nrow(x)

## S4 method for signature 'vclMatrix'
ncol(x)

## S4 method for signature 'gpuMatrix'
nrow(x)

## S4 method for signature 'gpuMatrix'
ncol(x)
```

Arguments

- **x**: A gpuMatrix/vclMatrix object

Value

An integer of length 1

Author(s)

Charles Determan Jr.

permute

Permuting functions for gpuR objects

Description

Generate a permutation of row or column indices

Usage

```r
permute(x, MARGIN, order)
```
platformInfo

Arguments

- **x**: A gpuR matrix object
- **MARGIN**: Dimension over which the ordering should be applied, 1 indicates rows, 2 indicates columns
- **order**: An integer vector indicating order of rows to assign

Value

- A gpuR object

Author(s)

Charles Determan Jr.

---

platformInfo

OpenCL Platform Information

Description

Get basic information about the OpenCL platform

Usage

platformInfo(platform_idx = 1L)

Arguments

- **platform_idx**: An integer value to specify which platform to check

Value

- **platformName**: Platform Name
- **platformVendor**: Platform Vendor
- **platformVersion**: Platform OpenCL Version
- **platformExtensions**: Available platform extensions

Author(s)

Charles Determan Jr.
**pmax**

'Parallel' Maxima and Minima

**Description**

`pmax` and `pmin` take one or more vectors as arguments and return a single vector giving the 'parallel' maxima (or minima) of the argument vectors.

**Usage**

```r
pmax(...)  
pmin(...)  
```

**Arguments**

```
...  
gpuR or numeric arguments  
```

**See Also**

`pmax` `pmin`

---

**pmin.vclVector**

# @rdname setGeneric("pmax", signature = ")

**Description**

# @rdname #' @method pmax vclVector #' @export setMethod("pmax", "vclVector", function(...,  

na.rm = FALSE) print("called correctly") gpuVecpmax(..., order = 1L))

**Usage**

```r
## S3 method for class 'vclVector'  
pmin(..., na.rm = FALSE)
```

**Arguments**

```
...  
a vclVector object  
na.rm  
a logical indicating whether missing values should be removed.
```
Details

#' @export setMethod("pmax", "vclMatrix", function(..., na.rm = FALSE) gpuMatpmax(..., order = 1L))
#' @export setMethod("pmax", "gpuVector", function(..., na.rm = FALSE) gpuVecpmax(..., order = 1L))
#' @export setMethod("pmax", "gpuMatrix", function(..., na.rm = FALSE) gpuMatpmax(..., order = 1L))

---

**pocl_check**

**POCL Version Check**

Description

Versions of POCL up to 0.15-pre have a bug which results in values being returned when NA values should be (e.g. fractional powers of negative values)

Usage

pocl_check()

---

**print.gpuMatrix**

**S3 print for gpuMatrix objects**

Description

prints a gpuMatrix object that is truncated to fit the screen

Usage

```r
## S3 method for class 'gpuMatrix'
print(x, ..., n = NULL, width = NULL)
```

Arguments

- `x`: A gpuMatrix object
- `...`: Additional arguments to print
- `n`: Number of rows to display
- `width`: Number of columns to display
The QR Decomposition of a gpuR matrix

**Description**

qr computes the QR decomposition of a gpuR matrix

**Usage**

```r
## S3 method for class 'gpuMatrix'
qr(x, ..., inplace = FALSE)

## S3 method for class 'vclMatrix'
qr(x, ..., inplace = FALSE)
```

**Arguments**

- `x`: A gpuR matrix
- `...`: further arguments passed to or from other methods
- `inplace`: Logical indicating if operations performed inplace

**Value**

A list containing the QR decomposition of the matrix of class gpuQR. The returned value is a list with the following components:

- `qr`: a matrix with the same dimensions as `x`.
- `betas`: vector of numeric values containing additional information of `qr` for extracting `Q` and `R` matrices.

**Note**

This an S3 generic of `qr`. The default continues to point to the default base function.

Furthermore, the list returned does not contain the exact same elements as `qr`. The matrix storage format applied herein doesn’t match the base compact form. The method also doesn’t return `qr`aux, `rank`, or `pivot` but instead returns `betas`.

**Author(s)**

Charles Determan Jr.

**See Also**

`qr`
Description

Reconstruct the Q or R Matrices from a gpuQR Object

Returns the components of the QR decomposition.

Usage

```r
## S4 method for signature 'gpuQR'
qr.R(qr, complete = FALSE)

## S4 method for signature 'gpuQR'
qr.Q(qr, complete = FALSE)
```

Arguments

- `qr`: gpuQR object
- `complete`: not currently used

Value

- `qr.R` returns all of R
- `qr.Q` returns all of Q

Author(s)

Charles Determan Jr.

See Also

- `qr.R`, `qr.Q`

---

setcontext

Set Context

Description

Change the current context used by default

Usage

```r
setContext(id = 1L)
```

Arguments

- `id`: Integer identifying which context to set
**See Also**

- `listContexts`
### slice

#### Vector Slices

**Description**

This doesn’t create a copy, it provides a child class that points to a contiguous subvector of a `gpuVector` or `vclVector`. Non-contiguous slices are currently not supported.

**Usage**

```r
slice(object, start, end)
```

```r
data <- vec
slice <- slice(data, 1, 5)
```

```r
## S4 method for signature 'gpuVector,integer,integer'
slice(object, start, end)
```

```r
data <- vec
slice <- slice(data, 1, 5)
```

```r
## S4 method for signature 'vclVector,integer,integer'
slice(object, start, end)
```

**Arguments**

- **object**: A `gpuVector` or `vclVector` object
- **start**: An integer indicating the start of slice
- **end**: An integer indicating the end of slice

**Details**

This function allows a user to create a `gpuR` vector object that references a continuous subset of columns and rows of another `gpuR` vector object without a copy.

NOTE - this means that altering values in a vector slice object will alter values in the source vector.

**Value**

A `gpuVectorSlice` or `vclVectorSlice` object

**Author(s)**

Charles Determan Jr.
solve,vclMatrix,vclMatrix-method

Solve a System of Equations for gpuR objects

Description

This function solves the equation \( a \times x = b \) for \( x \), where \( b \) can be either a vector or a matrix.

Usage

```r
## S4 method for signature 'vclMatrix,vclMatrix'
solve(a, b, ...)

## S4 method for signature 'vclMatrix,missing'
solve(a, b, ...)

## S4 method for signature 'gpuMatrix,gpuMatrix'
solve(a, b, ...)

## S4 method for signature 'gpuMatrix,missing'
solve(a, b, ...)
```

Arguments

- `a` A gpuR object
- `b` A gpuR object
- `...` further arguments passed to or from other methods

Value

A gpuR object

Author(s)

Charles Determan Jr.

Summary,gpuVector-method

gpuR Summary methods

Description

Methods for the base Summary methods S4groupGeneric
svd

Usage

## S4 method for signature 'gpuVector'
Summary(x, ..., na.rm = FALSE)

## S4 method for signature 'vclMatrix'
Summary(x, ..., na.rm = FALSE)

## S4 method for signature 'vclVector'
Summary(x, ..., na.rm = FALSE)

## S4 method for signature 'gpuMatrix'
Summary(x, ..., na.rm = FALSE)

Arguments

x A gpuR object

... Additional arguments passed to method (not currently used)

na.rm a logical indicating whether missing values should be removed (not currently used)

Value

For min or max, a length-one vector

svd Singular Value Decomposition of a gpuR matrix

Description

Compute the singular-value decomposition of a gpuR matrix

Usage

svd(x, nu, nv, LINPACK)

## S3 method for class 'vclMatrix'
svd(x, nu, nv, LINPACK)

## S3 method for class 'gpuMatrix'
svd(x, nu, nv, LINPACK)

Arguments

x A gpuR matrix

nu ignored

nv ignored

LINPACK ignored
Value

The SVD decomposition of the matrix. The returned value is a list with the following components:

- d a vector containing the singular values of x
- u a matrix whose columns contain the left singular vectors of x.
- v a matrix whose columns contain the right singular vectors of x.

Note

This an S3 generic of `svd`. The default continues to point to the default base function.

Author(s)

Charles Determan Jr.

See Also

`svd`

---

*synchronize*  
*Synchronize Device Execution*

Description

This pauses execution until the processing is complete on the device (CPU/GPU/etc.). This is important especially for benchmarking applications.

Usage

`synchronize()`

Author(s)

Charles Determan Jr.

Examples

```r
## Not run:
mat <- vclMatrix(rnorm(500^2), ncol = 500, nrow = 500)
system.time(mat %% mat)
system.time(mat %% mat; synchronize())

## End(Not run)
```
Description

Given a gpur matrix x, t returns the transpose of x

Usage

```r
## S4 method for signature 'vclMatrix'
t(x)
## S4 method for signature 'gpuMatrix'
t(x)
```

Arguments

- `x`: A gpur matrix

Value

A gpur matrix

Author(s)

Charles Determan Jr.

tcrossprod,gpuVector,gpuVector-method

vclMatrix Crossproduct

Description

Return the matrix cross-product of two conformable matrices using a GPU. This is equivalent to t(x) or x device and host is required.

Usage

```r
## S4 method for signature 'gpuVector,gpuVector'
tcrossprod(x, y)
## S4 method for signature 'gpuVector,missing'
tcrossprod(x, y)
## S4 method for signature 'vclMatrix,missing'
crossprod(x, y)
```
Arguments

\[ x \] A vclMatrix

\[ y \] A vclMatrix

Value

A vclMatrix
**typeof.gpuMatrix-method**

*Get gpuR object type*

**Author(s)**

Charles Determan Jr.

---

**Description**

typeof determines the type (i.e. storage mode) of a gpuR object

**Usage**

```r
## S4 method for signature 'gpuMatrix'
typeof(x)

## S4 method for signature 'gpuVector'
typeof(x)

## S4 method for signature 'vclMatrix'
typeof(x)

## S4 method for signature 'vclVector'
typeof(x)
```

**Arguments**

- `x` A gpuR object

**Author(s)**

Charles Determan Jr.

---

**vclMatrix**

*Construct a vclMatrix*

**Description**

Construct a vclMatrix of a class that inherits from vclMatrix. This class points to memory directly on the GPU to avoid the cost of data transfer between host and device.
Usage

vcIMatrix(data = NA, nrow = NA, ncol = NA, type = NULL, ...)  

## S4 method for signature 'matrix'
vcIMatrix(data, type = NULL, ctx_id = NULL)

## S4 method for signature 'missing'
vcIMatrix(data, nrow = NA, ncol = NA,  
          type = NULL, ctx_id = NULL)

## S4 method for signature 'numeric'
vcIMatrix(data, nrow, ncol, type = NULL,  
          ctx_id = NULL)

## S4 method for signature 'integer'
vcIMatrix(data, nrow, ncol, type = NULL,  
          ctx_id = NULL)

Arguments

data An object that is or can be converted to a matrix object  
nrow An integer specifying the number of rows  
ncol An integer specifying the number of columns  
type A character string specifying the type of vclMatrix. Default is NULL where type is inherited from the source data type.  
... Additional method to pass to vclMatrix methods  
ctx_id An integer specifying the object’s context

Value

A vclMatrix object

Author(s)

Charles Determan Jr.

dimensions: 612.0x792.0

Description

This is the 'mother' class for all vclMatrix objects. These objects are pointers to viennacl matrices directly on the GPU. This will avoid the overhead of passing data back and forth between the host and device.
As such, any changes made to normal R 'copies' (e.g. A <- B) will be propagated to the parent object.

There are multiple child classes that correspond to the particular data type contained. These include ivclMatrix, fvclMatrix, and dvclMatrix corresponding to integer, float, and double data types respectively.

### Slots

Common to all vclMatrix objects in the package

- **address**: Pointer to data matrix
- **.context_index**: Integer index of OpenCL contexts
- **.platform_index**: Integer index of OpenCL platforms
- **.platform**: Name of OpenCL platform
- **.device_index**: Integer index of active device
- **.device**: Name of active device

### Note

R does not contain a native float type. As such, the matrix data within a fvclMatrix-class will be represented as double but downcast when any vclMatrix methods are used.

May also remove the type slot

### Author(s)

Charles Determan Jr.

### See Also

ivclMatrix-class, fvclMatrix-class, dvclMatrix-class

---

**vclVector**

Construct a vclVector

---

**Description**

Construct a vclVector of a class that inherits from vclVector. This class points to memory directly on the GPU to avoid the cost of data transfer between host and device.
Usage

```r
vclVector(data, length, type = NULL, ...)
```

## S4 method for signature 'vector,missing'
```r
vclVector(data, length, type = NULL,
           ctx_id = NULL)
```

## S4 method for signature 'missing,ANY'
```r
vclVector(data, length, type = NULL,
           ctx_id = NULL)
```

## S4 method for signature 'numeric,numericOrInt'
```r
vclVector(data, length, type = NULL,
           ctx_id = NULL)
```

## S4 method for signature 'vclMatrix,missing'
```r
vclVector(data, length = NULL,
           type = NULL, ctx_id = NULL, col = NULL, row = NULL)
```

Arguments

- **data**: An object that is or can be converted to a vector
- **length**: A non-negative integer specifying the desired length.
- **type**: A character string specifying the type of vclVector. Default is NULL where type is inherited from the source data type.
- **...**: Additional method to pass to vclVector methods
- **ctx_id**: An integer specifying the object's context
- **col**: index of column to extract from vclMatrix
- **row**: index of row to extract from vclMatrix

Value

A vclVector object

Author(s)

Charles Determan Jr.

---

**vclVector-class**

### vclVector Class

Description

This is the 'mother' class for all vclVector objects. All other vclVector classes inherit from this class but there are no current circumstances where this class is used directly.

There are multiple child classes that correspond to the particular data type contained. These include ivclVector.
Slots

Common to all vclMatrix objects in the package

address: Pointer to data matrix
.context_index: Integer index of OpenCL contexts
.platform_index: Integer index of OpenCL platforms
.platform: Name of OpenCL platform
.device_index: Integer index of active device
.device: Name of active device

Author(s)

Charles Determan Jr.

See Also

ivclVector-class

gpuMatrix-class, igpuMatrix-class, dgpumatrix-class
zvclMatrix-class  
zvclMatrix Class

Description

An complex double type matrix in the S4 vclMatrix representation.

Slots

address: Pointer to a complex double type matrix

Author(s)

Charles Determan Jr.

See Also

vclMatrix-class, ivclMatrix-class, fvclMatrix-class

[.gpuMatrix,missing,missing,missing-method

Extract gpuR object elements

Description

Operators to extract or replace elements

Usage

## S4 method for signature 'gpuMatrix,missing,missing,missing'
x[i, j, drop]

## S4 method for signature 'gpuMatrix,missing,numeric,missing'
x[i, j, drop]

## S4 method for signature 'gpuMatrix,numeric,missing,missing'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'gpuMatrix,numeric,numeric,missing'
x[i, j, drop]

## S4 replacement method for signature 'gpuMatrix,numeric,missing,numeric'
x[i, j, ...] <- value

## S4 replacement method for signature 'igpuMatrix,numeric,missing,integer'
x[i, j] <- value

## S4 replacement method for signature 'gpuMatrix,missing,numeric,numeric'

x[i, j] <- value

## S4 replacement method for signature 'igpuMatrix,missing,numeric,integer'

x[i, j] <- value

## S4 replacement method for signature 'gpuMatrix,numeric,numeric,numeric'

x[i, j] <- value

## S4 replacement method for signature 'igpuMatrix,numeric,numeric,integer'

x[i, j] <- value

## S4 method for signature 'gpuVector,missing,missing,missing'

x[i, j, drop]

## S4 method for signature 'gpuVector,numeric,missing,missing'

x[i, j, drop]

## S4 replacement method for signature 'gpuVector,numeric,missing,numeric'

x[i, j] <- value

## S4 replacement method for signature 'gpuVector,numeric,missing,integer'

x[i, j] <- value

## S4 method for signature 'vclMatrix,missing,missing,missing'

x[i, j, drop]

## S4 method for signature 'vclMatrix,missing,numeric,missing'

x[i, j, drop]

## S4 method for signature 'vclMatrix,numeric,missing,missing'

x[i, j, ...,
    drop = TRUE]

## S4 method for signature 'vclMatrix,numeric,numeric,missing'

x[i, j, drop]

## S4 replacement method for signature 'vclMatrix,missing,numeric,numeric'

x[i, j] <- value

## S4 replacement method for signature 'ivclMatrix,missing,numeric,integer'

x[i, j] <- value

## S4 replacement method for signature 'vclMatrix,numeric,missing,numeric'

x[i, j, ...] <- value
## replacement method for signature 'ivclMatrix,numeric,missing,integer'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclMatrix,numeric,numeric,numeric'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'ivclMatrix,numeric,numeric,integer'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclMatrix,missing,missing,matrix'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclMatrix,missing,missing,vclMatrix'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclMatrix,missing,numeric,vclMatrix'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclMatrix,missing,missing,numeric'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclMatrix,missing,missing,vclVector'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclMatrix,missing,numeric,vclVector'
\[ x[i, j] \leftarrow \text{value} \]

## method for signature 'vclVector,missing,missing,missing'
\[ x[i, j, \text{drop}] \]

## method for signature 'vclVector,numeric,missing,missing'
\[ x[i, j, \text{drop}] \]

## replacement method for signature 'vclVector,numeric,missing,numeric'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'ivclVector,numeric,missing,integer'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclVector,logical,missing,numeric'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclVector,missing,missing,numeric'
\[ x[i, j] \leftarrow \text{value} \]

## replacement method for signature 'vclVector,missing,missing,vclVector'
\[ x[i, j] \leftarrow \text{value} \]
## Outer Product

### Description
The outer product of two gpuR vector objects

### Usage
```r
## S4 method for signature 'gpuVector,gpuVector'
X %o% Y
```

### Arguments
- **X**: A gpuR object
- **Y**: A gpuR object

### Author(s)
Charles Determan Jr.
## Description

Multiply two 

## Usage

```r
## S4 method for signature 'gpuVector,gpuVector'
x %*% y

## S4 method for signature 'gpuVector,gpuMatrix'
x %*% y

## S4 method for signature 'vclMatrix,vclMatrix'
x %*% y

## S4 method for signature 'vclMatrix,vclVector'
x %*% y

## S4 method for signature 'vclMatrix,matrix'
x %*% y

## S4 method for signature 'matrix,vclMatrix'
x %*% y

## S4 method for signature 'vclVector,vclVector'
```
%*%,gpuVector,gpuVector-method

x %*% y

## S4 method for signature 'vclVector,vclMatrix'
x %*% y

## S4 method for signature 'gpuMatrix,gpuMatrix'
x %*% y

## S4 method for signature 'gpuMatrix,gpuVector'
x %*% y

## S4 method for signature 'gpuMatrix,matrix'
x %*% y

## S4 method for signature 'matrix,gpuMatrix'
x %*% y

Arguments

x A gpuR object

y A gpuR object

Author(s)

Charles Determan Jr.
Index

[,gpuMatrix,missing,missing,missing-method] ([,gpuMatrix,missing,missing,missing-method), 62
[,gpuMatrix,missing,numeric,missing-method] <-[,gpuMatrix,numeric,numeric,numeric-method ([[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,gpuMatrix,numeric,missing,missing-method] <-[,gpuVector,numeric,missing,missing-method ([[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,gpuVector,missing,missinng,missing-method] <-[,igpuMatrix,numeric,numeric,missing-method ([[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,gpuVector,numeric,missing,missing-method] <-[,igpuMatrix,numeric,missing,missing-method ([[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclMatrix,missing,missing,missing-method] <-[,igpuMatrix,numeric,numeric,integer-method ([,[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclMatrix,numeric,missing,missing-method] <-[,ivclMatrix,numeric,missing,missing-method ([[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclMatrix,numeric,missing,missing-method] <-[,ivclMatrix,numeric,missing,missing-method ([,[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclMatrix,numeric,missing,missing-method] <-[,ivclMatrix,numeric,missing,missing-method ([,[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclVector,missing,missing,missing-method] <-[,ivclVector,numeric,missing,missing-method ([,[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclVector,numeric,missing,missing-method] <-[,vclMatrix,numeric,missing,missing-method ([,[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclMatrix,missing,numeric,numeric-method] <-[,vclMatrix,missing,missing,numeric-method ([,[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclMatrix,numeric,missing,numeric-method] <-[,vclMatrix,numeric,missing,numeric-method ([,[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62
[,vclMatrix,numeric,missing,numeric-method] <-[,vclMatrix,numeric,numeric-method ([,[,gpuMatrix,missing,missing,missing-method), ([,gpuMatrix,missing,missing,missing-method), 62

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>colnames&lt;-,gpuMatrix-method(colnames)</td>
<td>11</td>
</tr>
<tr>
<td>colnames&lt;-,vclMatrix-method(colnames)</td>
<td>11</td>
</tr>
<tr>
<td>colSums,gpuMatrix(Mmethod(colSums,gpuMatrix-method), 12</td>
<td>12</td>
</tr>
<tr>
<td>colSums,vclMatrix(Mmethod(colSums,vclMatrix-method), 13</td>
<td>13</td>
</tr>
<tr>
<td>Compare,gpuVector,vector-method(Mmethod(Compare,vector,gpuVector-method), 13</td>
<td>13</td>
</tr>
<tr>
<td>Compare,gpuVector-vector-method(Mmethod(Compare,vector,gpuVector-method), 13</td>
<td>13</td>
</tr>
<tr>
<td>Compare-vector-gpuVector(Mmethod(Compare,vector,gpuVector-method), 13</td>
<td>13</td>
</tr>
<tr>
<td>cov,gpuMatrix,gpuMatrix,missing,character-method(Mmethod(cov,vclMatrix,missing,missing,missing-method), 14</td>
<td>14</td>
</tr>
<tr>
<td>cov,gpuMatrix,gpuMatrix,missing,missing-method(Mmethod(cov,vclMatrix,missing,missing,missing-method), 14</td>
<td>14</td>
</tr>
<tr>
<td>cov,gpuMatrix,missing,missing,character-method(Mmethod(cov,vclMatrix,missing,missing,missing-method), 14</td>
<td>14</td>
</tr>
<tr>
<td>cov,gpuMatrix,missing,missing,missing-method(Mmethod(cov,vclMatrix,missing,missing,missing-method), 14</td>
<td>14</td>
</tr>
<tr>
<td>cov,vclMatrix,missing,missing,character-method(Mmethod(cov,vclMatrix,missing,missing,missing-method), 14</td>
<td>14</td>
</tr>
<tr>
<td>cov,vclMatrix,missing,missing,missing-method(Mmethod(cov,vclMatrix,missing,missing,missing-method), 14</td>
<td>14</td>
</tr>
<tr>
<td>det,gpuMatrix,Mmethod(det,gpuMatrix,Mmethod()), 20</td>
<td>20</td>
</tr>
<tr>
<td>detectCPUs(Mmethod(detcPUs), 20</td>
<td>20</td>
</tr>
<tr>
<td>detectGPUs,21,21,31</td>
<td>21</td>
</tr>
<tr>
<td>detectPlatforms,17,21,31</td>
<td>21</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,Mmethod(crossprod,gpuMatrix,Mmethod()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,missing,method(crossprod,gpuMatrix,missing,method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,matrix-method(crossprod,gpuMatrix,matrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,missing-method(crossprod,gpuMatrix,missing-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,matrix,gpuMatrix-method(crossprod,matrix,gpuMatrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,matrix,vclMatrix-method(crossprod,matrix,vclMatrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,vclMatrix,Mmethod(crossprod,vclMatrix,Mmethod()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,vector-method(crossprod,vclMatrix,vector-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,missing-method(crossprod,vclMatrix,missing-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,vector-method(crossprod,vclMatrix,vector-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,missing-method(crossprod,vclMatrix,missing-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>currentContext,16</td>
<td>16</td>
</tr>
<tr>
<td>currentDevice,17</td>
<td>17</td>
</tr>
<tr>
<td>currentPlatform,17</td>
<td>17</td>
</tr>
<tr>
<td>custom_opencl,18,50</td>
<td>18</td>
</tr>
<tr>
<td>cpuInfo,31</td>
<td>31</td>
</tr>
<tr>
<td>cpuInfo(gpuInfo),30</td>
<td>30</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,Mmethod(crossprod,gpuMatrix,Mmethod()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,missing,method(crossprod,gpuMatrix,missing,method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,matrix-method(crossprod,gpuMatrix,matrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,missing-method(crossprod,gpuMatrix,missing-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,matrix,gpuMatrix-method(crossprod,matrix,gpuMatrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,matrix,vclMatrix-method(crossprod,matrix,vclMatrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,vclMatrix,Mmethod(crossprod,vclMatrix,Mmethod()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,vector-method(crossprod,vclMatrix,vector-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,missing-method(crossprod,vclMatrix,missing-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,vector-method(crossprod,vclMatrix,vector-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,missing-method(crossprod,vclMatrix,missing-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>currentContext,16</td>
<td>16</td>
</tr>
<tr>
<td>currentDevice,17</td>
<td>17</td>
</tr>
<tr>
<td>currentPlatform,17</td>
<td>17</td>
</tr>
<tr>
<td>custom_opencl,18,50</td>
<td>18</td>
</tr>
<tr>
<td>cpuInfo,31</td>
<td>31</td>
</tr>
<tr>
<td>cpuInfo(gpuInfo),30</td>
<td>30</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,Mmethod(crossprod,gpuMatrix,Mmethod()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,missing,method(crossprod,gpuMatrix,missing,method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,matrix-method(crossprod,gpuMatrix,matrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,gpuMatrix,missing-method(crossprod,gpuMatrix,missing-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,matrix,gpuMatrix-method(crossprod,matrix,gpuMatrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,matrix,vclMatrix-method(crossprod,matrix,vclMatrix-method()), 15</td>
<td>15</td>
</tr>
<tr>
<td>crossprod,vclMatrix,Mmethod(crossprod,vclMatrix,Mmethod()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,vector-method(crossprod,vclMatrix,vector-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,missing-method(crossprod,vclMatrix,missing-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,vector-method(crossprod,vclMatrix,vector-method()), 55</td>
<td>55</td>
</tr>
<tr>
<td>crossprod,vclMatrix,missing-method(crossprod,vclMatrix,missing-method()), 55</td>
<td>55</td>
</tr>
</tbody>
</table>
inplace, function, numeric, gpuVector-method
   (inplace), 37
inplace, function, numeric, vclMatrix-method
   (inplace), 37
inplace, function, numeric, vclVector-method
   (inplace), 37
inplace, function, vclMatrix, missing-method
   (inplace), 37
inplace, function, vclMatrix, numeric-method
   (inplace), 37
inplace, function, vclMatrix, vclMatrix-method
   (inplace), 37
inplace, function, vclVector, missing-method
   (inplace), 37
inplace, function, vclVector, numeric-method
   (inplace), 37
inplace, function, vclVector, vclVector-method
   (inplace), 37
ivclMatrix-class, 39
ivclVector-class, 39
length, gpuMatrix-method
   (length, gpuVector-method), 40
length, gpuVector-method, 40
length, vclMatrix-method
   (length, gpuVector-method), 40
length, vclVector-method
   (length, gpuVector-method), 40
length-gpuMatrix
   (length, gpuVector-method), 40
length-vclMatrix
   (length, gpuVector-method), 40
listContexts, 16, 40, 50
log, gpuMatrix-method
   (log, gpuVector-method), 41
log, gpuVector-method, 41
log, vclMatrix-method
   (log, gpuVector-method), 41
log, vclVector-method
   (log, gpuVector-method), 41
log-gpuR-method (log, gpuVector-method), 41
Math, gpuMatrix-method
   (Math, gpuVector-method), 42
Math, gpuVector-method, 42
Math, vclMatrix-method
   (Math, gpuVector-method), 42
Math-gpuR-method
   (Math, gpuVector-method), 42
ncol, gpuMatrix-method
   (nrow, vclMatrix-method), 44
ncol, vclMatrix-method
   (nrow, vclMatrix-method), 44
norm, 43
norm, ANY, character-method
   (norm, vclMatrix, character-method), 43
norm, ANY, missing-method
   (norm, vclMatrix, character-method), 43
norm, gpuMatrix, character-method
   (norm, vclMatrix, character-method), 43
norm, gpuMatrix, missing-method
   (norm, vclMatrix, character-method), 43
norm, vclMatrix, character-method, 43
norm, vclMatrix, missing-method
   (norm, vclMatrix, character-method), 43
nrow, gpuMatrix-method
   (nrow, vclMatrix-method), 44
nrow, vclMatrix-method, 44
permute, 44
platformInfo, 45
pmax, 46, 46
pmin, 46
pmin (pmax), 46
pmin.vclVector, 46
pocl_check, 47
print.gpuMatrix, 47
qr, 48
qr.gpuMatrix, 48
qr.gpuR (qr.gpuMatrix), 48
qr.Q, 49
qr.Q, gpuQR-method (qr.R, gpuQR-method), 49
qr.R, 49
qr.R, gpuQR-method, 49
qr.vclMatrix (qr.gpuMatrix), 48
INDEX

rowMeans, gpuMatrix-method (colSums, gpuMatrix-method), 12
rowMeans, vclMatrix-method (colSums, vclMatrix-method), 13
rowSums, gpuMatrix-method (colSums, gpuMatrix-method), 12
rowSums, vclMatrix-method (colSums, vclMatrix-method), 13

S4groupGeneric, 4, 42, 52
set_device_context, 50
setContext, 49
slice, 51
slice, gpuVector, integer, integer-method (slice), 51
slice, vclVector, integer, integer-method (slice), 51
solve, gpuMatrix, gpuMatrix-method (solve, vclMatrix, vclMatrix-method), 52
solve, gpuMatrix, missing-method (solve, vclMatrix, vclMatrix-method), 52
solve, vclMatrix, missing-method (solve, vclMatrix, vclMatrix-method), 52
solve, vclMatrix, vclMatrix-method, 52
Summary, gpuMatrix-method (Summary, gpuVector-method), 52
Summary, gpuVector-method, 52
Summary, vclMatrix-method (Summary, gpuVector-method), 52
Summary, vclVector-method (Summary, gpuVector-method), 52
Summary-gpuR-method (Summary, gpuVector-method), 52
svd, 53, 54
synchronize, 54
t, gpuMatrix (t, vclMatrix-method), 55
t, gpuMatrix-method (t, vclMatrix-method), 55
t, vclMatrix (t, vclMatrix-method), 55
t, vclMatrix-method, 55
tcrossprod, gpuMatrix, gpuMatrix-method (tcrossprod, gpuMatrix, missing-method), 15
tcrossprod, gpuMatrix, missing-method (tcrossprod, gpuMatrix, missing-method), 15
tcrossprod, gpuVector, gpuVector-method, 55
tcrossprod, gpuVector, missing-method (tcrossprod, gpuVector, gpuVector-method), 55
tcrossprod, matrix, gpuMatrix-method (tcrossprod, gpuMatrix, missing-method), 15
tcrossprod, matrix, vclMatrix-method (tcrossprod, gpuVector, gpuVector-method), 55
tcrossprod, vclMatrix, matrix-method (tcrossprod, gpuVector, gpuVector-method), 55
tcrossprod, vclMatrix, missing-method (tcrossprod, gpuVector, gpuVector-method), 55
tcrossprod, vclMatrix, vclMatrix-method (tcrossprod, gpuVector, gpuVector-method), 55
tcrossprod, vclMatrix, vclVector-method (tcrossprod, gpuVector, gpuVector-method), 55
tcrossprod, vclVector, vclMatrix-method (tcrossprod, gpuVector, gpuVector-method), 55
tcrossprod, vclVector, vclVector-method (tcrossprod, gpuVector, gpuVector-method), 55
typeof, gpuMatrix-method, 57
typeof, gpuVector-method (typeof, gpuMatrix-method), 57
typeof, vclMatrix-method (typeof, gpuMatrix-method), 57
typeof, vclVector-method (typeof, gpuMatrix-method), 57

vclMatrix, 9, 19, 57
vclMatrix, integer (vclMatrix), 57
vclMatrix, integer-method (vclMatrix), 57
vclMatrix, matrix (vclMatrix), 57
vclMatrix, matrix-method (vclMatrix), 57
vclMatrix, missing (vclMatrix), 57
vclMatrix, missing-method (vclMatrix), 57
vclMatrix, numeric-method (vclMatrix), 57
vclMatrix, vector (vclMatrix), 57
vclMatrix-class, 58
vclVector, 19, 51, 59
vclVector, missing (vclVector), 59
vclVector, missing, ANY-method
  (vclVector), 59
vclVector, numeric, numericOrInt-method
  (vclVector), 59
vclVector, vclMatrix, missing-method
  (vclVector), 59
vclVector, vector (vclVector), 59
vclVector, vector, missing-method
  (vclVector), 59
vclVector-class, 60
zgpuMatrix-class, 61
zvclMatrix-class, 62