Package 'Rcublas'

June 10, 2016

Type Package

Title GPU Enabled BLAS, LAPACK and Statistical Functions and Random Number Generators

Version 0.1

Date 2016-1-07

Author Yuan Li, Hua Zhou

Maintainer Yuan Li <yli16@ncsu.edu>

Description Provides GPU-accelerated algebra and random numbergenerator functions by wrapping CUDA library. It also includes some self-defined GPU algebra functions which are unavailable from CUDA library.

License CPL

Depends R (>= 3.0.0)

NeedsCompilation yes

SystemRequirements Nvidia's CUDA toolkit (>= release 5.5); Linux operating system; GNU make.

URL https://github.ncsu.edu/yli16/R_CUBLAS

RoxygenNote 5.0.1

R topics documented:

`addgpu` *addgpu*

Description

This function computes the element-wise addition of two given vectors/matrices by using CUDA cublas function cublasDgeam

Usage

```
addgpu(x, y)
```

Arguments

`x`	list consisting of R external GPU pointer and dimension
`y`	list consisting of R external GPU pointer and dimension

Value

element-wise addition of two vectors/matrices (x + y), a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

subtractgpu

Examples

```
a <- 1:4
b <- 2:5
a_gpu <- creategpu(a)
b_gpu <- creategpu(b)
addgpu(a_gpu, b_gpu) -> c_gpu
gathergpu(c_gpu)
```

---

betagpu *betagpu*

Description

This function computes the beta function of the given vector/matrix by using self-defined CUDA function

Usage

```
betagpu(x, y)
```

Arguments

x	list consisting of R external GPU pointer and dimension
y	list consisting of R external GPU pointer and dimension

Value

beta function result of given vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

gammagpu

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
betagpu(a_gpu, a_gpu) -> b_gpu
gathergpu(b_gpu)
```

createpu *createpu*

Description

Create a GPU vector/matrix by copying from the input R vector

Usage

```
createpu(input, nrow = NULL, ncol = NULL)
```

Arguments

input	R vector to be copied
nrow	the desired number of rows
ncol	the desired number of columns

Details

This function creates a vector/matrix in GPU by calling the CUDA cudamalloc function, and then copys from input R vector. The output of this function is a list consisting of the GPU pointer and its dimension.

If either one of nrow or ncol is not given, an one column matrix/vector is returned. This function returns row-major matrix.

Value

a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

Note

output is a R external GPU pointer and can only be used in Rcublas functions

Author(s)

Yuan Li

See Also

gathergpu

Examples

```
a <- rnorm(6)
a_gpu <- creategpu(a, 2, 3)
gathergpu(a_gpu)
```

`dividegpu` *dividegpu*

Description

This function computes the element-wise division of two given vectors/matrices by using self-defined CUDA function

Usage

```
dividegpu(x, y)
```

Arguments

`x`	list consisting of R external GPU pointer and dimension
`y`	list consisting of R external GPU pointer and dimension

Value

element-wise division of vectors/matrices (x / y), a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

`multiplygpu`

Examples

```
a <- 1:4
b <- 2:5
a_gpu <- creategpu(a)
b_gpu <- creategpu(b)
dividegpu(a_gpu, b_gpu) -> c_gpu
gathergpu(c_gpu)
```

dnormgpu *dnormgpu*

Description

This function computes the normal distribution density of given vector/matrix

Usage

```
dnormgpu(input, mean = 0, sd = 1)
```

Arguments

input	list consisting of R external GPU pointer and dimension
mean	vector/matrix of mean
sd	vector/matrix of standard deviation

Details

If mean or sd are not specified they assume the default values of 0 and 1, respectively.

Value

normal distribution density vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

pnormgpu

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
dnormgpu(a_gpu) -> b_gpu
gathergpu(b_gpu)
```

dotgpu *dotgpu*

Description

This function computes the dot product of two given vectors/matrix by using CUDA cublas function cublasDdot

Usage

```
dotgpu(x, y)
```

Arguments

x	list consisting of R external GPU pointer and dimension
y	list consisting of R external GPU pointer and dimension

Value

the resulting dot product

See Also

norm2gpu

Examples

```
a <- 1:4
b <- 2:5
a_gpu <- creategpu(a)
b_gpu <- creategpu(b)
dotgpu(a_gpu, b_gpu)
```

expgpu *expgpu*

Description

This function computes the exponential of given vector/matrix by using self-defined CUDA function

Usage

```
expgpu(input)
```

Arguments

input	list consisting of R external GPU pointer and dimension

Value

exponential of vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

`loggpu`

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
expgpu(a_gpu) -> b_gpu
gathergpu(b_gpu)
```

---

gammagpu *gammagpu*

Description

This function computes the gammma function of given vector/matrix by using self-defined CUDA function

Usage

```
gammagpu(input)
```

Arguments

`input`	list consisting of R external GPU pointer and dimension

Value

gamma result of vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

`betagpu`

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
gammagpu(a_gpu) -> b_gpu
gathergpu(b_gpu)
```

---

gathergpu *gathergpu*

Description

Copy GPU matrix/vector to R vector

Usage

```
gathergpu(input)
```

Arguments

input	list consisting of R external GPU pointer and its dimension

Details

This function copys GPU vector/matrix to R vector

The output is always R vector, and GPU matrix will be copied by row-major. For example, an m by n GPU matrix will be converted to a m*n R vector.

Value

R vector

Note

output is R vector and can be used by any R functions

Author(s)

Yuan Li

See Also

gathergpu creategpu

Examples

```
a <- 1:6
am_gpu <- creategpu(a, 3, 2)
gathergpu(am_gpu)
```

GPUobject *GPUobject*

Description

classify the input as GPU vector/matrix and assign its dimension

Usage

```
GPUobject(input, nrow, ncol)
```

Arguments

input	R external pointer
nrow	number of rows
ncol	number of columns

Details

This function classifies the input object as GPU vector/matrix and assign its dimension The output of this function is a list consisting of the GPU pointer and its dimension

Value

a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

Note

output is a R external GPU pointer and can only be used in Rcublas functions

Author(s)

Yuan Li

See Also

gathergpu

gpuquery *gpuquery*

Description

This function returns the information of available GPU device in system

Usage

```
gpuquery()
```

See Also

creategpu

Examples

```
gpuquery()
```

inversegpu *inversegpu*

Description

This function computes the inversion of given matrix (squared) by using CUDA cublas function cublasDgetrfBatched and cublasDgetriBatched (LU decomposition)

Usage

```
inversegpu(X)
```

Arguments

X	input matrix; list of R external GPU pointer and dimension

Value

matrix inversion, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

mmgpu creategpu

Examples

```
a <- 1:9
a_gpu <- creategpu(a, 3, 3)
inversegpu(a_gpu) -> c_gpu
gathergpu(c_gpu)
```

`loggpu` *loggpu*

Description

This function computes the natural logarithms of given vector/matrix by using self-defined CUDA function

Usage

```
loggpu(input)
```

Arguments

`input`	list consisting of R external GPU pointer and dimension

Value

natural logarithms of vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

`expgpu`

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
loggpu(a_gpu) -> b_gpu
gathergpu(b_gpu)
```

maxgpu *maxgpu*

Description

This function finds the (smallest) index of the element with the maximum magnitude of given vector/matrix by using CUDA cublas function cublasIdamin

Usage

```
maxgpu(input)
```

Arguments

input	list consisting of R external GPU pointer and dimension

Value

the resulting index

See Also

mingpu

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
maxgpu(a_gpu)
```

meangpu *meangpu*

Description

Compute the mean of given vector/matrix

Usage

```
meangpu(x)
```

Arguments

x	list consisting of R external GPU pointer and dimension

Details

This function computes the mean of given vector/matrix by using self-defined CUDA function

Value

vector/matrix mean

Author(s)

Yuan Li

See Also

`sumgpu`

Examples

```
a <- creategpu(1:4)
meangpu(a)
```

`mingpu` *mingpu*

Description

This function finds the (smallest) index of the element with the minimum magnitude of given vector by using CUDA cublas function cublasIdamin

Usage

```
mingpu(input)
```

Arguments

`input`	list consisting of R external GPU pointer and dimension

Value

the resulting index

See Also

`maxgpu`

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
mingpu(a_gpu)
```

mmgpu *mmgpu*

Description

This function computes the matrix-matrix multiplication (X * Y) by using CUDA cublas function cublasDgemm

Usage

```
mmgpu(X, Y)
```

Arguments

X	input matrix; list of R external GPU pointer and dimension
Y	input matrix; list of R external GPU pointer and dimension

Value

matrix-matrix multiplication (X * Y), a list consisting of

- ptr: GPU pointer
- m: matrix X's number of rows
- n: matrix Y's number of columns

See Also

mmgpu

Examples

```
a <- 1:6
b <- 2:7
a_gpu <- creategpu(a, 2, 3)
b_gpu <- creategpu(b, 3, 2)
mmgpu(a_gpu, b_gpu) -> c_gpu
gathergpu(c_gpu)
```

multiplygpu *multiplygpu*

Description

This function computes the element-wise multiplication of two given vectors/matricesby using CUDA cublas function cublasDdgmm

Usage

```
multiplygpu(x, y)
```

Arguments

x	list consisting of R external GPU pointer and dimension
y	list consisting of R external GPU pointer and dimension

Value

element-wise multiplication of vectors/matrices (x * y), a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

dividegpu

Examples

```
a <- 1:4
b <- 2:5
a_gpu <- creategpu(a)
b_gpu <- creategpu(b)
multiplygpu(a_gpu, b_gpu) -> c_gpu
gathergpu(c_gpu)
```

mvgpu *mvgpu*

Description

This function computes the matrix-vector multiplication (X * y) by using CUDA cublas function cublasDgemv

Usage

```
mvgpu(X, y)
```

Arguments

X	input matrix; list of R external GPU pointer and dimension
y	input vector; list of R external GPU pointer and dimension

Value

matrix-vector multiplication (X * y), a list consisting of

- ptr: GPU pointer
- m: matrix X's number of rows
- n: matrix X's number of columns; vector y's number of elements

See Also

mmgpu

Examples

```
a <- 1:4
b <- 2:3
a_gpu <- creategpu(a, 2, 2)
b_gpu <- creategpu(b)
mvgpu(a_gpu, b_gpu) -> c_gpu
gathergpu(c_gpu)
```

norm2gpu *norm2gpu*

Description

This function computes Euclidean norm of given vector/matrix by using CUDA cublas function cublasDnrm2

Usage

```
norm2gpu(input)
```

Arguments

input	list consisting of R external GPU pointer and dimension

Value

vector Euclidean norm, a non-negative number

Author(s)

Yuan Li

See Also

gathergpu

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
norm2gpu(a_gpu)
```

pnormgpu *pnormgpu*

Description

This function computes the standard normal distribution cumulative density (CDF) of given vector/matrix

Usage

```
pnormgpu(input)
```

Arguments

input	list consisting of R external GPU pointer and dimension

Value

standard normal CDF, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

dnormgpu

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
pnormgpu(a_gpu) -> b_gpu
gathergpu(b_gpu)
```

---

powergpu *powergpu*

Description

This function computes the power of given vector/matrix by using self-defined CUDA function

Usage

```
powergpu(input, alpha = 1)
```

Arguments

input	list consisting of R external GPU pointer and dimension
alpha	power factor

Value

powered vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

sqrtgpu

Examples

```
a <- 1:4
b <- 2
a_gpu <- creategpu(a)
powergpu(a_gpu, b) -> b_gpu
gathergpu(b_gpu)
```

Rcublas *GPU enabled BLAS functions and random number generators*

Description

This package provides GPU-accelerated algebra and random numbergenerator functions by wrapping CUDA library. It also includes some self-defined GPU algebra functions which are unavailable from CUDA library.

Details

Package:	Rcublas
Type:	Package
Version:	1.0
Date:	2016-01-07
License:	CPL

~~ An overview of how to use the package, including the most important ~~ ~~ functions ~~

Author(s)

Yuan Li, Hua Zhou

Maintainer: Yuan Li <yli16@ncsu.edu>

---

rlognormgpu *rlognormgpu*

Description

This function generates log-normally distributed random numbers by using CUDA curand function CURAND_RNG_PSEUDO_DEFAULT and curandGenerateLogNormalDouble

Usage

```
rlognormgpu(n, mean = 0, sd = 1, seed = 1)
```

Arguments

n	number of random numbers
mean	mean of log-normal distribution; default value 0
sd	standard deviation of log-normal distribution; default value 1
seed	random number generator seed; default value 1

Value

generated random numbers vector, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

rnormgpu

Examples

```
a_gpu <- rlognormgpu(100, 0, 1, 15)
gathergpu(a_gpu)
```

---

rnormgpu *rnormgpu*

Description

This function generates normally distributed random numbers by using CUDA curand function CURAND_RNG_PSEUDO_DEFAULT and curandGenerateNormalDouble

Usage

```
rnormgpu(n, mean = 0, sd = 1, seed = 1)
```

Arguments

n	number of random numbers
mean	mean of normal distribution; default value 0
sd	standard deviation of normal distribution; default value 1
seed	random number generator seed; default value 1

Value

generated random numbers vector, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

`rlognormgpu`

Examples

```
a_gpu <- rnormgpu(100, 0, 1, 15)
gathergpu(a_gpu)
```

---

rpoisgpu *rpoisgpu*

Description

This function generates Poisson distributed random numbers by using CUDA curand function CURAND_RNG_PSEUDO_DEFAULT and curandGeneratePoisson

Usage

```
rpoisgpu(n, lambda = 1, seed = 1)
```

Arguments

`n`	number of random numbers
`lambda`	mean of Poisson distribution; default value 1
`seed`	random number generator seed; default value 1

Value

generated random numbers vector, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

`runifgpu`

Examples

```
a_gpu <- rpoisgpu(100, 1)
```

runifgpu *runifgpu*

Description

This function generates uniformly distributed random numbers between 0 and 1 by using CUDA curand function CURAND_RNG_PSEUDO_DEFAULT and curandGenerateUniformDouble

Usage

```
runifgpu(n, seed = 1)
```

Arguments

n	number of random numbers
seed	random number generator seed; default value 1

Value

generated random numbers vector, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

creategpu

Examples

```
a_gpu <- runifgpu(100, 15)
gathergpu(a_gpu)
```

scalegpu *scalegpu*

Description

This function scales the given vector/matrix by a scalar by using CUDA cublas function cublasDcopy

Usage

```
scalegpu(input, alpha)
```

Arguments

input	list consisting of R external GPU pointer and dimension
alpha	scale factor

Value

scaled vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

expgpu

Examples

```
a <- 1:4
b <- 2
a_gpu <- creategpu(a)
scalegpu(a_gpu, b) -> b_gpu
gathergpu(b_gpu)
```

---

sqrtgpu *sqrtgpu*

Description

This function computes the square root of given vector/matrix by using self-defined CUDA function

Usage

```
sqrtgpu(input)
```

Arguments

input	list consisting of R external GPU pointer and dimension

Value

square root of vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

`expgpu`

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
sqrtgpu(a_gpu) -> b_gpu
gathergpu(b_gpu)
```

`subsetgpu` *subsetgpu*

Description

This function returns the specified subset of given GPU vector/matrix by using self-defined CUDA function

Usage

```
subsetgpu(input, index)
```

Arguments

`input`	list consisting of R external GPU pointer and dimension
`index`	index of the vector/matrix subset

Value

subset of the given vector/matrix, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

`creategpu`

Examples

```
a <- 1:4
a_gpu <- creategpu(a)
subsetgpu(a_gpu,c(1, 2))->b_gpu
gathergpu(b_gpu)
```

subtractgpu *subtractgpu*

Description

This function computes the element-wise subtraction of two given vectors/matrices by using CUDA cublas function cublasDgeam

Usage

```
subtractgpu(x, y)
```

Arguments

x	list consisting of R external GPU pointer and dimension
y	list consisting of R external GPU pointer and dimension

Value

element-wise subtraction of vectors or matrices (x - y), a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

addgpu

Examples

```
a <- 1:4
b <- 2:5
a_gpu <- creategpu(a)
b_gpu <- creategpu(b)
subtractgpu(a_gpu, b_gpu) -> c_gpu
gathergpu(c_gpu)
```

`sumgpu` *sumgpu*

Description

Compute the summation of given vector/matrix

Usage

```
sumgpu(x)
```

Arguments

`x`	list consisting of R external GPU pointer and dimension

Details

This function computes the summation of given vector/matrix by using self-defined CUDA function

Value

vector/matrix summation

Author(s)

Yuan Li

See Also

`meangpu`

Examples

```
a <- creategpu(1:4)
sumgpu(a)
```

tgpu *tgpu*

Description

This function transposes the given matrix by using CUDA cublas cublasDgeam

Usage

```
tgpu(X)
```

Arguments

X	input matrix; list of R external GPU pointer and dimension

Value

matrix transpose, a list consisting of

- ptr: GPU pointer
- m: number of rows
- n: number of columns

See Also

creategpu

Examples

```
a <- 1:12
a_gpu <- creategpu(a, 3, 4)
tgpu(a_gpu) -> c_gpu
gathergpu(c_gpu)
```

vargpu *vargpu*

Description

Compute the variance of given vector/matrix

Usage

```
vargpu(x)
```

Arguments

x	list consisting of R external GPU pointer and dimension

Details

This function computes the variance of given vector/matrix by using self-defined CUDA function

Value

vector/matrix variance

Author(s)

Yuan Li

See Also

sumgpu

Examples

```
a <- creategpu(1:4)
vargpu(a)
```