

# RcppArmadillo: Sparse Matrix Support

Binxiang Ni<sup>1</sup>, Dmitry Selivanov<sup>1</sup>, Dirk Eddelbuettel<sup>c</sup>, and Qiang Kou<sup>d</sup>

<sup>a</sup><https://github.com/binxiangni>; <sup>b</sup><https://github.com/dselivanov>; <sup>c</sup><http://dirk.eddelbuettel.com>; <sup>d</sup><https://github.com/thirdwing>

This version was compiled on February 10, 2020

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Sparse Matrix</b>	<b>2</b>
2.1	dgCMatrix . . . . .	2
2.2	dtCMatrix . . . . .	2
2.3	dsCMatrix . . . . .	3
2.4	dgTMatrix . . . . .	4
2.5	dtTMatrix . . . . .	5
2.6	dsTMatrix . . . . .	6
2.7	dgRMatrix . . . . .	6
2.8	dtRMatrix . . . . .	7
2.9	dsRMatrix . . . . .	8
2.10	indMatrix . . . . .	8
2.11	pMatrix . . . . .	9
2.12	ddiMatrix . . . . .	10

## 1. Introduction

The documentation is intended for the convenience of RcppArmadillo sparse matrix users based on integration of the documentation of library **Matrix** (Bates and Maechler, 2018) and **Armadillo** (Sanderson, 2010; Sanderson and Curtin, 2016).

There are 31 types of sparse matrices in the **Matrix** package that can be used directly. But for now, only 12 of them are supported in RcppArmadillo: dgCMatrix, dtCMatrix, dsCMatrix, dgTMatrix, dtTMatrix, dsTMatrix, dgRMatrix, dtRMatrix, dsRMatrix, indMatrix, pMatrix, ddiMatrix.

In the **Armadillo** library, sparse matrix content is currently stored as **CSC** format. Such kind of format is quite similar to numeric column-oriented sparse matrix in the library **Matrix** (including dgCMatrix, dtCMatrix and dsCMatrix). When a sparse matrix from the package **Matrix** is passed through the **RcppArmadillo** package (Eddelbuettel and Sanderson, 2014; Eddelbuettel *et al.*, 2018), it will be converted or mapped to CSC format, then undertaken operations on, and finally output as a dgCMatrix in R.

In what follows, we will always assume this common header:

```
#include <RcppArmadillo.h>
// [[Rcpp::depends(RcppArmadillo)]]

using namespace Rcpp;
using namespace arma;
```

but not generally show it.

## 2. Sparse Matrix

### 2.1. dgCMatrix.

#### **Synopsis.**

- Description: general column-oriented numeric sparse matrix.
- Constructor

```
- new("dgCMatrix", ...)  
  
- Matrix(*, sparse = TRUE)  
  
- sparseMatrix()
```

- Coercion

```
- as(*, "CsparseMatrix")  
  
- as(*, "dgCMatrix")
```

#### **C++ Code.**

```
// [[Rcpp::export]]  
sp_mat sqrt_(sp_mat X) {  
  return sqrt(X);  
}
```

#### **R Code.**

```
R> i <- c(1,3:8)  
R> j <- c(2,9,6:10)  
R> x <- 7 * (1:7)  
R> A <- sparseMatrix(i, j, x = x)  
R> sqrt_(A)  
8 x 10 sparse Matrix of class "dgCMatrix"  
  
[1,] . 2.645751 . . . . .  
[2,] . . . . .  
[3,] . . . . . 3.741657 .  
[4,] . . . 4.582576 . . . .  
[5,] . . . . . 5.291503 . . .  
[6,] . . . . . 5.91608 . . .  
[7,] . . . . . 6.480741 . .  
[8,] . . . . . 7 . . .
```

### 2.2. dtCMatrix.

### Synopsis.

- Description: triangular column-oriented numeric sparse matrix.
- Constructor

```
- new("dtCMatrix", ...)  
  
- Matrix(*, sparse = TRUE)  
  
- sparseMatrix(*, triangular=TRUE)
```

- Coercion

```
- as(*, "triangularMatrix")  
  
- as(*, "dtCMatrix")
```

### C++ Code.

```
// [[Rcpp::export]]  
sp_mat symmatl_(sp_mat X) {  
  return symmatl(X);  
}
```

### R Code.

```
R> dtC <- new("dtCMatrix", Dim = c(5L, 5L), uplo = "L",  
             x = c(10, 1, 3, 10, 1, 10, 1, 10, 10),  
             i = c(0L, 2L, 4L, 1L, 3L, 2L, 4L, 3L, 4L),  
             p = c(0L, 3L, 5L, 7, 9))  
R> symmatl_(dtC)  
5 x 5 sparse Matrix of class "dtCMatrix"  
  
[1,] 10 . 1 . 3  
[2,] . 10 . 1 .  
[3,] 1 . 10 . 1  
[4,] . 1 . 10 .  
[5,] 3 . 1 . 10
```

## 2.3. dsCMatrix.

### Synopsis.

- Description: symmetric column-oriented numeric sparse matrix.
- Constructor

```
- new("dsCMatrix", ...)  
  
- Matrix(*, sparse = TRUE)
```

- `sparseMatrix(*, symmetric = TRUE)`
- Coercion
  - `as(*, "symmetricMatrix")`
  - `as(*, "dsCMatrix")`

#### C++ Code.

```
// [[Rcpp::export]]
sp_mat trimatu(sp_mat X) {
  return trimatu(X);
}
```

#### R Code.

```
R> i <- c(1,3:8)
R> j <- c(2,9,6:10)
R> x <- 7 * (1:7)
R> dsC <- sparseMatrix(i, j, x = x, symmetric = TRUE)
R> trimatu(dsC)
10 x 10 sparse Matrix of class "dgCMatrix"

[1,] . 7 . . . . . . . .
[2,] . . . . . . . . . .
[3,] . . . . . . . 14 .
[4,] . . . . . 21 . . . .
[5,] . . . . . . 28 . . .
[6,] . . . . . . . 35 . .
[7,] . . . . . . . . 42 .
[8,] . . . . . . . . . 49
[9,] . . . . . . . . . .
[10,] . . . . . . . . . .
```

## 2.4. dgTMatrix.

### Synopsis.

- Description: general numeric sparse matrix in triplet form.
- Constructor
  - `new("dgTMatrix", ...)`
  - `sparseMatrix(*, giveCsparse=FALSE)`
  - `spMatrix()`
- Coercion
  - `as(*, "TsparseMatrix")`
  - `as(*, "dgTMatrix")`

### C++ Code.

```
// [[Rcpp::export]]
sp_mat multiply(sp_mat A, sp_mat B) {
  return A * B;
}

// [[Rcpp::export]]
sp_mat trans_(sp_mat X) {
  return trans(X);
}

// [[Rcpp::export]]
int trace_(sp_mat X) {
  return trace(X);
}
```

### R Code.

```
R> dgT <- new("dgTMatrix",
             i = c(1L,1L,0L,3L,3L),
             j = c(2L,2L,4L,0L,0L),
             x=10*1:5, Dim=4:5)
R> dgT_t <- trans_(dgT)
R> prod <- multiply(dgT, dgT_t)
R> trace_(prod)
[1] 9900
```

## 2.5. dtTMatrix.

### Synopsis.

- Description: triangular numeric sparse matrix in triplet form.
- Constructor

– `new("dtTMatrix", ...)`

– `code{sparseMatrix(*, triangular=TRUE, giveCsparse=FALSE)}`

- Coercion

– `as(*, "triangularMatrix")`

– `as(*, "dtTMatrix")`

### C++ Code.

```
// [[Rcpp::export]]
sp_mat diag_ones(sp_mat X) {
  X.diag().ones();
  return X;
}
```

```
}
```

#### **R Code.**

```
R> dtT <- new("dtTMatrix", x= c(3,7),
              i= 0:1, j=3:2, Dim= as.integer(c(4,4)))
R> diag_ones(dtT)
4 x 4 sparse Matrix of class "dgCMatrix"

[1,] 1 . . 3
[2,] . 1 7 .
[3,] . . 1 .
[4,] . . . 1
```

## **2.6. dsTMatrix.**

#### **Synopsis.**

- Description: symmetric numeric sparse matrix in triplet form.
- Constructor

```
- new("dsTMatrix", ...)
```

```
- sparseMatrix(*, symmetric=TRUE, giveCsparse=FALSE)
```

- Coercion

```
- as(*, "symmetricMatrix")
```

```
- as(*, "dsTMatrix")
```

#### **C++ Code.**

```
// [[Rcpp::export]]
int trace_(sp_mat X) {
  return trace(X);
}
```

#### **R Code.**

```
R> mm <- Matrix(toeplitz(c(10, 0, 1, 0, 3)),
               sparse = TRUE)
R> mT <- as(mm, "dgTMatrix")
R> dsT <- as(mT, "symmetricMatrix")
R> trace_(dsT)
[1] 50
```

## **2.7. dgRMatrix.**

### Synopsis.

- Description: general row-oriented numeric sparse matrix.
- Constructor

```
– new("dgRMatrix", ...)
```

- Coercion

```
– as(*, "RsparseMatrix")
```

```
– as(*, "dgRMatrix")
```

### C++ Code.

```
// [[Rcpp::export]]  
sp_mat square_(sp_mat X) {  
  return square(X);  
}
```

### R Code.

```
R> dgR <- new("dgRMatrix", j=c(0L,2L,1L,3L),  
              p=c(0L,2L,3L,3L,4L),  
              x=c(3,1,2,1),  
              Dim=rep(4L,2))  
R> square_(dgR)  
4 x 4 sparse Matrix of class "dgCMatrix"  
  
[1,] 9 . 1 .  
[2,] . 4 . .  
[3,] . . . .  
[4,] . . . 1
```

## 2.8. dtRMatrix.

### Synopsis.

- Description: triangular row-oriented numeric sparse matrix.
- Constructor

```
– new("dtRMatrix", ...)
```

### C++ Code.

```
// [[Rcpp::export]]  
sp_mat repmat_(sp_mat X, int i, int j) {  
  return repmat(X, i, j);  
}
```

### R Code.

```
R> dtR <- new("dtRMatrix", Dim = c(2L,2L),
              x = c(5, 1:2), p = c(0L,2:3), j= c(0:1,1L))
R> repmat_(dtR, 2, 2)
4 x 4 sparse Matrix of class "dgCMatrix"

[1,] 5 1 5 1
[2,] . 2 . 2
[3,] 5 1 5 1
[4,] . 2 . 2
```

## 2.9. dsRMatrix.

### Synopsis.

- Description: symmetric row-oriented numeric sparse matrix.
- Constructor

```
- new("dsRMatrix", ...)
```

- Coercion

```
- as("dsCMatrix", "dsRMatrix")
```

### C++ Code.

```
// [[Rcpp::export]]
sp_mat sign_(sp_mat X) {
  return sign(X);
}
```

### R Code.

```
R> dsR <- new("dsRMatrix", Dim = c(2L,2L),
              x = c(-3,1), j = c(1L,1L), p = 0:2)
R> sign_(dsR)
2 x 2 sparse Matrix of class "dgCMatrix"

[1,] . -1
[2,] -1 1
```

## 2.10. indMatrix.

### Synopsis.

- Description: index matrix.
- Constructor



– new("indMatrix", ...)

- Coercion

– as(\*, "indMatrix")

#### **C++ Code.**

```
// [[Rcpp::export]]
sp_mat multiply(sp_mat A, sp_mat B) {
  return A * B;
}
```

#### **R Code.**

```
R> ind <- as(2:4, "indMatrix")
R> dgT <- new("dgTMatrix",
             i = c(1L,1L,0L,3L,3L),
             j = c(2L,2L,4L,0L,0L),
             x=10*1:5, Dim=4:5)
R> multiply(ind, dgT)
3 x 5 sparse Matrix of class "dgCMatrix"

[1,] . . 30 . .
[2,] . . . . .
[3,] 90 . . . .
```

## **2.11. pMatrix.**

### **Synopsis.**

- Description: permutation matrix.
- Constructor

– new("pMatrix", ...)

- Coercion

– as(\*, "pMatrix")

#### **C++ Code.**

```
// [[Rcpp::export]]
sp_mat multiply(sp_mat A, sp_mat B) {
  return A * B;
}
```

#### **R Code.**

```

R> pM <- as(c(2,3,1,4), "pMatrix")
R> dgT <- new("dgTMatrix",
              i = c(1L,1L,0L,3L,3L),
              j = c(2L,2L,4L,0L,0L),
              x=10*1:5, Dim=4:5)
R> multiply(pM, dgT)
4 x 5 sparse Matrix of class "dgCMatrix"

[1,] . . 30 . .
[2,] . . . . .
[3,] . . . . 30
[4,] 90 . . . .

```

## 2.12. ddiMatrix.

### Synopsis.

- Description: numeric diagonal Matrix.
- Constructor

```

- new("ddiMatrix", ...)

- Diagonal(*)

```

### C++ Code.

```

// [[Rcpp::export]]
sp_mat multiply(sp_mat A, sp_mat B) {
  return A * B;
}

```

### R Code.

```

R> ddi <- Diagonal(4)
R> dgR <- new("dgRMatrix", j=c(0L,2L,1L,3L),
              p=c(0L,2L,3L,3L,4L),
              x=c(3,1,2,1),
              Dim=rep(4L,2))
R> multiply(ddi, dgR)
4 x 4 sparse Matrix of class "dgCMatrix"

[1,] 3 . 1 .
[2,] . 2 . .
[3,] . . . .
[4,] . . . 1

```

## References

Bates D, Maechler M (2018). *Matrix: Sparse and Dense Matrix Classes and Methods*. R package version 1.2-14, URL <http://CRAN.R-Project.org/package=Matrix>.

- Eddelbuettel D, Francois R, Bates D, Ni B (2018). *RcppArmadillo: 'Rcpp' Integration for the 'Armadillo' Templated Linear Algebra Library*. R package version 0.8.600.0, URL <http://CRAN.R-Project.org/package=RcppArmadillo>.
- Eddelbuettel D, Sanderson C (2014). "RcppArmadillo: Accelerating R with high-performance C++ linear algebra." *Computational Statistics and Data Analysis*, **71**, 1054–1063. doi:10.1016/j.csda.2013.02.005. URL <http://dx.doi.org/10.1016/j.csda.2013.02.005>.
- Sanderson C (2010). "Armadillo: An open source C++ Algebra Library for Fast Prototyping and Computationally Intensive Experiments." *Technical report*, NICTA. URL <http://arma.sourceforge.net>.
- Sanderson C, Curtin R (2016). "Armadillo: A Template-Based C++ Library for Linear Algebra." *JOSS*, **1**(2). doi:10.21105/joss.00026. URL <http://dx.doi.org/10.21105/joss.00026>.