Comparison between C-implemented and R-implemented dual-loop matrix summing function performance

Running

To run this project, run the following commands:

```
# matrix test
Rscript mat_tests.R
```

Building and running

If you edit the C code, to recompile run:

```
bash make_c.sh
```

View Evaluation

To install packages necessary for this .rmd document, run:

```
Rscript install_libs.R
```

Evaluation

The experiment shows the performance comparison between the R-implemented and C-implemented matrix summing functions for different matrix sizes. As the matrix size increases, the C-implemented function demonstrates significantly better performance compared to the R-implemented function. Surprisingly, the speedup remains fairly constant in relative terms, stabilizing at about 4x

> Note: Evaluation script run on an AMD Ryzen 9 7950X3D cpu with enough RAM for all matrix sizes

Matrix size	sum1 run duration (secs)	sum2 run duration (secs)
5x5	7.152557e-06	7.867813e-06
10x10	9.775162e-06	5.00679e-06
50x50	9.346008e-05	8.106232e-06
100x100	0.0003376007	1.955032e-05
500x500	0.007472992	0.001415014
1000x1000	0.03007007	0.005748034
5000x5000	0.6559205	0.1854615
10000x10000	2.692389	0.6747584
20000x20000	10.67763	2.615553
30000x30000	24.33534	5.987761

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

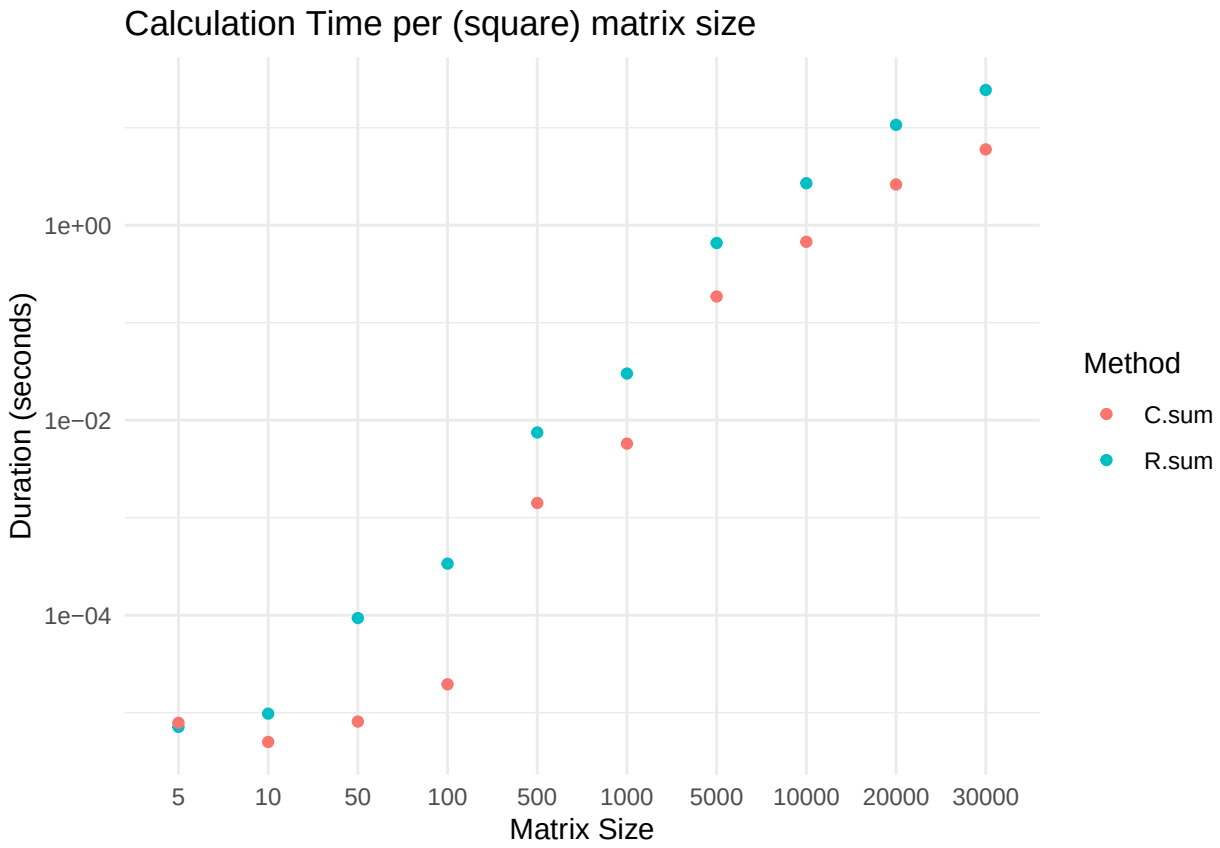
```r
library(tidyr)

# prepare data
data <- tribble(
  ~Matrix.size, ~R.sum, ~C.sum,
  "5", 7.152557e-06, 7.867813e-06,
  "10", 9.775162e-06, 5.00679e-06,
  "50", 9.346008e-05, 8.106232e-06,
  "100", 0.0003376007, 1.955032e-05,
  "500", 0.007472992, 0.001415014,
  "1000", 0.03007007, 0.005748034,
  "5000", 0.6559205, 0.1854615,
  "10000", 2.692389, 0.6747584,
  "20000", 10.67763, 2.615553,
  "30000", 24.33534, 5.987761
)

# Convert Matrix.size to factor with desired order
data$Matrix.size <- factor(data$Matrix.size, levels = data$Matrix.size)

# rearrange data
data_long <- data %>%
  pivot_longer(cols = c(R.sum, C.sum),
               names_to = "Method",
               values_to = "Duration")

# Create the plot
ggplot(data_long, aes(x = Matrix.size, y = Duration, color = Method)) +
  geom_point() +
  scale_y_log10() +
  labs(x = "Matrix Size", y = "Duration (seconds)", color = "Method") +
  ggtitle("Calculation Time per (square) matrix size") +
  theme_minimal()
```


Calculation Time per (square) matrix size
Duration (seconds)
1e+00
1e−02
1e−04
5
10
50
100
500
1000
5000
10000
20000
30000
Matrix Size
Method
C.sum
R.sum