**RC++ Execution Times  
(in seconds)**

**Matrix size: 100x100**

0.0020051

0.00197506

0.002007961

**Matrix size: 175x175**

0.009022951

0.008987904

0.008021116

**Matrix size: 250x250**

0.02406383

0.02456689

0.02406406

**Code to reproduce RC++ execution time results:**

x = matrix(runif(30625),nrow=175,ncol=175)  
start.time <- Sys.time()  
mySweepC(x,175)  
end.time <- Sys.time()  
time.taken <- end.time - start.time  
print(time.taken)

**R Execution Times  
(in seconds)**

**Matrix size: 100x100**

0.358119

0.3544559

0.4157109

**Matrix size: 175x175**

19.60833

19.81161

19.2025

**Matrix size: 250x250**

55.68337

56.17705

54.52212

**Code to reproduce R execution time results:**

x = matrix(runif(30625),nrow=175,ncol=175)  
start.time <- Sys.time()  
mySweep(x,175)  
end.time <- Sys.time()  
time.taken <- end.time - start.time  
print(time.taken)

0.002007961 sec vs 0.3544559 sec

0.008987904 sec vs 19.60833 sec

0.02456689 sec vs 55.68337 sec

Thus, we can see that the speedup that RCpp gives over R is:

100x100: 207 times

175x175: 2204 times

250x250: 2286 times

Thus, the larger the matrix, the better the speedup, thanks mainly to gcc’s native loop unrolling.