Assignment 3

Chris Ostrouchov

1 Introduction

The last part of homework 3 was to implement Cholesky QR factorization. Below you can see the pseudocode for the following Algorithm 1 implemented in **src/**. As requested from part 2 comment are provided with the algorithm. The most expensive operations in code consist of the matrix multiplications, svd, and qr factorizations each consisting of $O(mn^2)$ operations. As we will see in homework 4 these operations can be speed up using GPUs.

Each of following tests were implemented on a 4 core intel i5 3570k processor. Table 1 details the specifications of the processor.

Algorithm 1 Cholesky QR Factorization

1: **procedure** CHOLESKY QR(A)
2: $Q \leftarrow A$
3: $G \leftarrow Q^T Q$ ▷ **gemm** matrix multiply $2mn^2$
4: $R \leftarrow I$
5: **while** $cond(G) > 100.0$ **do**
6: $U\Sigma V^T \leftarrow svd(G)$ ▷ **gesvd** svd $4mn^2 - \frac{4}{3}n^3 - \frac{2}{3}(m-n)^3$
7: $qr \leftarrow qr(\sqrt{\Sigma}V^T)$ ▷ **geqrf** householder qr $2mn^2 - \frac{2}{3}n^3$
8: $R \leftarrow rR$ ▷ **trmm** triangular matrix multiply mn^2
9: $Q \leftarrow Qr^{-1}$
10: **if** $cond(G) > 100.0$ **then**
11: $G = Q^T Q$
12: **end if**
13: **end while**
14: **return** Q, R
15: **end procedure**

2 Results/Discussion

The results shown for each of the Cholesky QR factorizations are consistent with what we would expect. Tables 2, 3, and 4 details the run times for the algorithm. To ensure that results were steady on the random matrices each test was repeated 10 times indicated by the column *Iteration*. The average statistics for each matrix size can be seen in Table 5. We seen that the matrix Q tends to be less and less orthogonal with itself as the matrix size increases. However as the matrix size increases the difference $A - QR$ remains consistent, indicating that our QR factorization scales. It

Table 1: Relevant processor specifications for the Intel i5 3570k processor

Processor	i5 3570k
Clock Rate	3.4 GHz
Cores	4
Hyper Threading	no
L1 Data Cache	4 x 32 KB
L1 Latency	4 cycles
L2 Data Cache	4 x 256 KB
L2 Latency	11 cycles
L3 Data Cache	6 MB shared
L3 Latency	28 cycles
Flops / Cycle	8 double precision

Table 2: Cholesky QR implementation run on uniform random matrices [0, 1] of size [1000, 32]

Iteration	Real time [s]	Proc time [s]	FLOPS	MFLOPS	norm(A - QR)	norm(I - Q'Q)
0	1.755000e-03	1.753000e-03	3274426	1867.898438	2.115893e-14	8.462742e-14
1	1.588000e-03	1.586000e-03	3252077	2050.489990	2.157851e-14	8.017566e-14
2	1.536000e-03	1.534000e-03	3261377	2126.060547	2.154383e-14	8.146731e-14
3	1.542000e-03	1.540000e-03	3271486	2124.341553	2.110507e-14	7.904130e-14
4	1.529000e-03	1.526999e-03	3250421	2128.632080	2.135736e-14	8.013199e-14
5	1.567001e-03	1.566000e-03	3311786	2114.805908	2.132766e-14	7.986240e-14
6	1.544999e-03	1.542998e-03	3282016	2127.035645	2.125873e-14	8.429233e-14
7	1.523001e-03	1.521999e-03	3255191	2138.758789	2.163561e-14	7.529909e-14
8	1.511000e-03	1.510000e-03	3232660	2140.834473	2.096920e-14	7.591334e-14
9	1.522001e-03	1.521001e-03	3245940	2134.082764	2.148730e-14	7.833960e-14
AVG:	1.561800e-03	1.560200e-03	3263738	2095.294019	2.134222e-14	7.991504e-14

is expected as discussed in Lecture 3 that the Q vectors become less orthogonal as the A matrix size increases.

As an experiment I implemented the Cholesky QR factorization in row and column major format. The difference can be seen in Table 5 and is quite significant!

Table 3: Cholesky QR implementation run on uniform random matrices [0, 1] of size [2000, 32]

Iteration	Real time [s]	Proc time [s]	FLOPS	MFLOPS	norm(A - QR)	norm(I - Q'Q)
0	2.708001e-03	2.705999e-03	5933804	2192.832275	2.978322e-14	1.090657e-13
1	2.694000e-03	2.691001e-03	5945926	2209.560059	2.989238e-14	1.149029e-13
2	2.601001e-03	2.597999e-03	5953210	2291.458740	2.976892e-14	1.115480e-13
3	2.604999e-03	2.598997e-03	5948217	2288.656006	3.001088e-14	1.037338e-13
4	2.628002e-03	2.626002e-03	6003736	2286.266602	2.987926e-14	1.074047e-13
5	2.583999e-03	2.577998e-03	5922186	2297.201660	3.013791e-14	1.128357e-13
6	2.609000e-03	2.605997e-03	5958144	2286.317627	2.924759e-14	1.060727e-13
7	2.599001e-03	2.596006e-03	5959188	2295.526855	3.043762e-14	1.072767e-13
8	2.563000e-03	2.559997e-03	5894307	2302.463623	2.915781e-14	1.096877e-13
9	2.659999e-03	2.641998e-03	5994745	2269.017822	3.045783e-14	1.107151e-13
AVG:	2.625100e-03	2.620199e-03	5951346	2271.930127	2.987734e-14	1.093243e-13

Table 4: Cholesky QR implementation run on uniform random matrices [0, 1] of size [3000, 32]

Iteration	Real time [s]	Proc time [s]	FLOPS	MFLOPS	norm(A - QR)	norm(I - Q'Q)
0	3.742002e-03	3.738001e-03	8612515	2304.043701	3.726676e-14	1.405381e-13
1	3.651001e-03	3.642999e-03	8556687	2348.802246	3.763546e-14	1.440986e-13
2	3.681995e-03	3.679000e-03	8608202	2339.821045	3.709919e-14	1.459294e-13
3	3.697000e-03	3.692001e-03	8601951	2329.889160	3.599132e-14	1.479825e-13
4	3.663003e-03	3.659002e-03	8574446	2343.385010	3.580429e-14	1.418014e-13
5	3.692001e-03	3.688000e-03	8629426	2339.865967	3.667232e-14	1.529708e-13
6	3.684998e-03	3.677011e-03	8605035	2340.232422	3.640847e-14	1.459143e-13
7	3.674999e-03	3.671989e-03	8604037	2343.147217	3.579514e-14	1.585664e-13
8	3.687009e-03	3.683999e-03	8630555	2342.713135	3.510808e-14	1.489244e-13
9	3.682002e-03	3.675014e-03	8607148	2342.081055	3.580766e-14	1.362737e-13
AVG:	3.685601e-03	3.680702e-03	8603000	2337.398096	3.635887e-14	1.463000e-13

Table 5: Average time of Cholesky QR implementation run on uniform random matrices [0, 1]. The first group represents row major ordering while the other represents column major ordering. Notice the HUGE speed difference and scaling!

Matrix size	Real time [s]	Proc time [s]	FLOPS	MFLOPS	norm(A - QR)	norm(I - Q'Q)
1000, 32	1.303700e-03	1.300800e-03	710303	546.777570	2.139051e-14	8.313594e-14
2000, 32	2.115399e-03	2.110701e-03	892099	422.699704	3.014203e-14	1.138414e-13
3000, 32	2.919903e-03	2.914798e-03	1024694	351.532965	3.629157e-14	1.520045e-13
1000, 32	1.561800e-03	1.560200e-03	3263738	2095.294019	2.134222e-14	7.991504e-14
2000, 32	2.625100e-03	2.620199e-03	5951346	2271.930127	2.987734e-14	1.093243e-13
3000, 32	3.685601e-03	3.680702e-03	8603000	2337.398096	3.635887e-14	1.463000e-13