Parallel QR with column pivoting using OpenMP

Team-No:31

Kaushik Gunda
Nikhil Badam

Content:

1. QR Decomposition

QR decomposition (also called a QR factorization) of a matrix is a decomposition of a matrix A into a product $A = QR$ of an orthogonal matrix Q and an upper triangular matrix R. QR decomposition is often used to solve the linear least squares problem and is the basis for a particular eigenvalue algorithm, the QR algorithm.

Let us take a matrix, $A \in R(m \times n)$ has linearly independent columns then it can be factored as

$$A = QR$$

Where,

Q = [q1 q2 q3 .....qn], ||qi|| = 1 & qi are orthonormal m-vectors.

R is a Upper Triangular Matrix, the diagonal elements R(i,i) should be positive.

2. Few Applications of QR Decomposition:

Linear equations
Least squares problems
Constrained least squares problems

3. Column pivoting in QR:

For a mxn matrix A, Decomposition of A is possible when the column of the A are Linearly independent. But for arbitrary A this can’t be true. So, we use P Permutation Matrix to convert A into a required one. Instead of decomposition of A we do decomposition of AP.

4. Algorithm for QR with Column Pivoting:

Given a Matrix A (Rmxn), the algorithm for QR decomposition using Column pivoting.

$\underline{\text{QP3Step} \ (\ m, \ n, \ nb, \ rowk, \ A \ )}$

Setup:

$\text{perm}(j) = j, \text{colnorms}(j) = ||Ae_j||_2^2, \quad j = 1:n$

$F(1\!:n, 1\!:\text{nb}) = 0$

Reduction Steps:

For $j = 1 : nb$

0. $k = rowk + j - 1$ % current row index
1. **Pivoting:** Choose p such that $\text{colnorms}(p) = \max(\text{colnorms}(j:n))$

   If $(\text{colnorms}(p) == 0)$ STOP

   If $(j \neq p)$ then % interchange

   $\text{perm}([j,p]) = \text{perm}([p,j]), \ A(:,[j,p]) = A(:,[p,j])$

   $\text{colnorms}([j,p]) = \text{colnorms}([p,j]), \ F([j,p],:) = F([p,j],:)$

end

2. **Update of pivot column:**

   $A(k:m,j) -= A(k:m, 1:j-1) * F(1:j-1, j)$
3. **Reduction:** Generate $H_j = I - tau(j)Y(j)Y(j)^T$ such that

   $H_j A(k:m,j) = \pm||(||_2 A(k:m,j)e_1$.
4. **Incremental Computation of** F**:**

   $F(j+1:n,j) = tau(j)A(j:m,j+1:n)^T Y(j:m,j)$.

   $F(1:n,j) -= tau(j)F(1:n,1:j-1)Y(j:m,1:j-1)^T Y(j:m,j)$.
5. **Update of pivot row:**

   $A(k,j+1:n) -= A(k,1:j) * F(j+1:n,1:j)^T$
6. **Norm downdate :**

   $\text{colnorms}(j+1\!:n) = \text{colnorms}(j+1:n) - A(k,j+1:n) .2$.

End For

Block update:

$A(k+1:m, \text{nb}+1:n) -= A(k+1:m, 1:\text{nb}) * F(\text{nb}+1:n, 1:\text{nb})^T$

Calculating A, F, T from HouseHolder H

For j = 1:r

  if j = 1 then

    $Y \leftarrow [\, v_1 \,]; T \leftarrow [\, -2 \,]$

  else

    $z \leftarrow -2 \cdot T \, Y^T v_j$

    $Y \leftarrow [\, Y \ v_j \,]$

$$T \leftarrow \begin{bmatrix} T & z \\ 0 & -2 \end{bmatrix}$$

  endif

end j

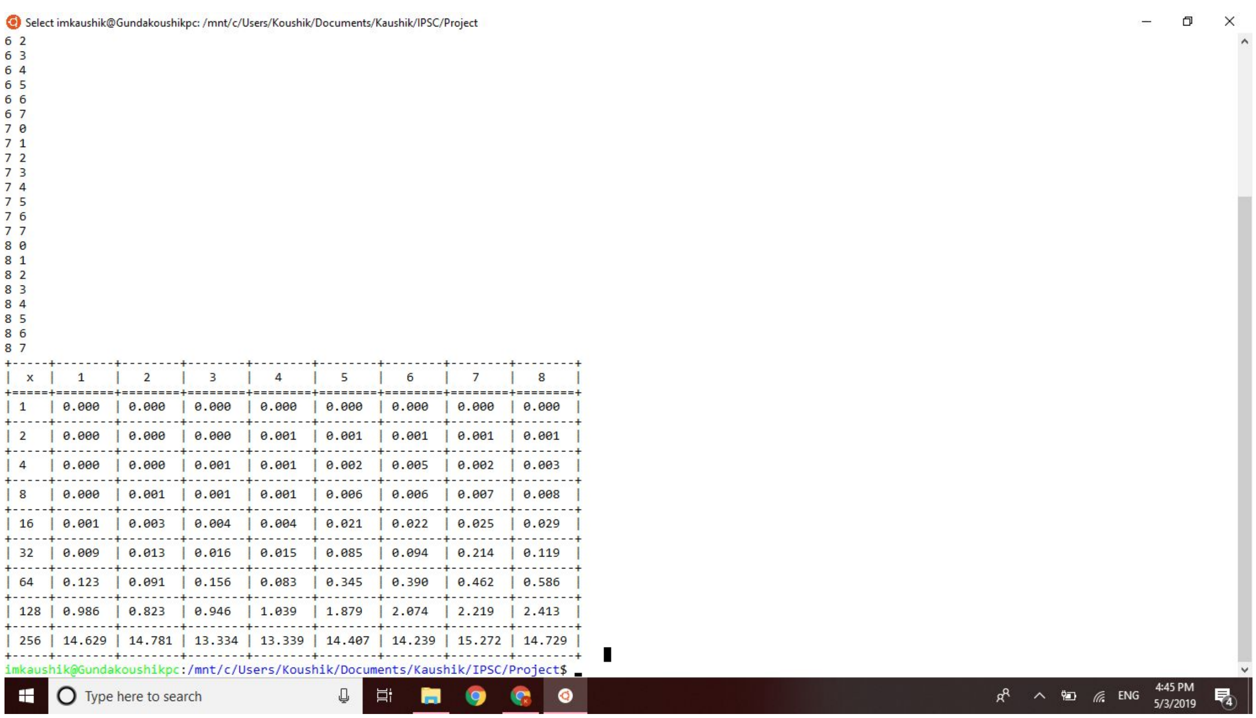
$$F^{\mathrm{T}} = TY^{\mathrm{T}}A(:, 1:n)$$

By the above algorithm we will get Q, R.

5. Parallelisation using OpenMP :

Generally, OpenMP is used for matrix multiplication, scalar product of matrix and also in similar cases. OpenMP is also used at several places like modifying T matrix, Updating Matrix A, F by the new HouseHolder, Updating A, with F,Y at last in the Algorithm.

6. Runtime Comparison:


Select imkaushik@Gundakoushikpc: /mnt/c/Users/Koushik/Documents/Kaushik/IPSC/Project

```
6 2
6 3
6 4
6 5
6 6
6 7
7 0
7 1
7 2
7 3
7 4
7 5
7 6
7 7
8 0
8 1
8 2
8 3
8 4
8 5
8 6
8 7
```

x	1	2	3	4	5	6	7	8
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.001	0.001	0.001	0.001	0.001
4	0.000	0.000	0.001	0.001	0.002	0.005	0.002	0.003
8	0.000	0.001	0.001	0.001	0.006	0.006	0.007	0.008
16	0.001	0.003	0.004	0.004	0.021	0.022	0.025	0.029
32	0.009	0.013	0.016	0.015	0.085	0.094	0.214	0.119
64	0.123	0.091	0.156	0.083	0.345	0.390	0.462	0.586
128	0.986	0.823	0.946	1.039	1.879	2.074	2.219	2.413
256	14.629	14.781	13.334	13.339	14.407	14.239	15.272	14.729

imkaushik@Gundakoushikpc:/mnt/c/Users/Koushik/Documents/Kaushik/IPSC/Project$