

Parallel Numerical Algorithms

Chapter 11 – QR Factorization

Prof. Michael T. Heath

Department of Computer Science
University of Illinois at Urbana-Champaign

CSE 512 / CS 554

Outline

- 1 QR Factorization
- 2 Householder Transformations
- 3 Givens Rotations

QR Factorization

- For given $m \times n$ matrix A , with $m > n$, *QR factorization* has form

$$A = Q \begin{bmatrix} R \\ O \end{bmatrix}$$

where matrix Q is $m \times m$ and orthogonal, and R is $n \times n$ and upper triangular

- Can be used to solve linear systems, least squares problems, etc.
- As with Gaussian elimination, zeros are introduced successively into matrix A , eventually reaching upper triangular form, but using orthogonal transformations instead of elementary eliminators

Methods for QR Factorization

- Householder transformations (elementary reflectors)
- Givens transformations (plane rotations)
- Gram-Schmidt orthogonalization

Householder Transformations

- *Householder transformation* has form

$$H = I - 2 \frac{vv^T}{v^T v}$$

where v is nonzero vector

- From definition, $H = H^T = H^{-1}$, so H is both orthogonal and symmetric
- For given vector a , choose v so that

$$Ha = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \alpha \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \alpha e_1$$

Householder Transformations

- Substituting into formula for H , we see that we can take

$$\mathbf{v} = \mathbf{a} - \alpha \mathbf{e}_1$$

and to preserve norm we must have $\alpha = \pm \|\mathbf{a}\|_2$, with sign chosen to avoid cancellation

Householder QR Factorization

for $k = 1$ **to** n

$$\alpha_k = -\text{sign}(a_{kk})\sqrt{a_{kk}^2 + \cdots + a_{mk}^2}$$

$$\mathbf{v}_k = [0 \ \cdots \ 0 \ a_{kk} \ \cdots \ a_{mk}]^T - \alpha_k \mathbf{e}_k$$

$$\beta_k = \mathbf{v}_k^T \mathbf{v}_k$$

if $\beta_k = 0$ **then** continue with next k **for** $j = k$ **to** n

$$\gamma_j = \mathbf{v}_k^T \mathbf{a}_j$$

$$\mathbf{a}_j = \mathbf{a}_j - (2\gamma_j/\beta_k)\mathbf{v}_k$$

end**end**

Parallel Householder QR

- Householder QR factorization is similar to Gaussian elimination for LU factorization
- Forming Householder vector v_k is analogous to computing multipliers in Gaussian elimination
- Subsequent updating of remaining unreduced portion of matrix is also analogous to Gaussian elimination
- Thus, parallel implementation is similar to parallel LU, but with Householder vectors broadcast horizontally instead of multipliers
- For this reason, we will not go into details

Givens Rotations

- *Givens rotation* operates on pair of rows to introduce single zero
- For given 2-vector $\mathbf{a} = [a_1 \ a_2]^T$, if

$$c = \frac{a_1}{\sqrt{a_1^2 + a_2^2}}, \quad s = \frac{a_2}{\sqrt{a_1^2 + a_2^2}}$$

then

$$\mathbf{G}\mathbf{a} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \alpha \\ 0 \end{bmatrix}$$

- Scalars c and s are cosine and sine of angle of rotation, and $c^2 + s^2 = 1$, so \mathbf{G} is orthogonal



Givens QR Factorization

- Givens rotations can be systematically applied to successive pairs of rows of matrix A to zero entire strict lower triangle
- Subdiagonal entries of matrix can be annihilated in various possible orderings (but once introduced, zeros should be preserved)
- Each rotation must be applied to all entries in relevant pair of rows, not just entries determining c and s
- Once upper triangular form is reached, product of rotations, Q , is orthogonal, so we have QR factorization of A



Parallel Givens QR Factorization

- With 1-D partitioning of A by columns, parallel implementation of Givens QR factorization is similar to parallel Householder QR factorization, with cosines and sines broadcast horizontally and each task updating its portion of relevant rows
- With 1-D partitioning of A by rows, broadcast of cosines and sines is unnecessary, but there is no parallelism unless multiple pairs of rows are processed simultaneously
- Fortunately, it is possible to process multiple pairs of rows simultaneously without interfering with each other



Parallel Givens QR Factorization

- Stage at which each subdiagonal entry can be annihilated is shown here for 8×8 example

$$\begin{bmatrix} \times & & & & & & & \\ 7 & \times & & & & & & \\ 6 & 8 & \times & & & & & \\ 5 & 7 & 9 & \times & & & & \\ 4 & 6 & 8 & 10 & \times & & & \\ 3 & 5 & 7 & 9 & 11 & \times & & \\ 2 & 4 & 6 & 8 & 10 & 12 & \times & \\ 1 & 3 & 5 & 7 & 9 & 11 & 13 & \times \end{bmatrix}$$

- Maximum parallelism is $n/2$ at stage $n - 1$ for $n \times n$ matrix



Parallel Givens QR Factorization

- Communication cost is high, but can be reduced by having each task initially reduce its entire local set of rows to upper triangular form, which requires no communication
- Then, in subsequent phase, task pairs cooperate in annihilating additional entries using one row from each of two tasks, exchanging data as necessary
- Various strategies can be used for combining results of first phase, depending on underlying network topology
- With hypercube, for example, final upper triangular form can be reached in $\log p$ combining steps

Parallel Givens QR Factorization

- With 2-D partitioning of A , parallel implementation combines features of 1-D column and 1-D row algorithms
- In particular, sets of rows can be processed simultaneously to annihilate multiple entries, but updating of rows requires horizontal broadcast of cosines and sines

References

- E. Chu and A. George, QR factorization of a dense matrix on a hypercube multiprocessor, *SIAM J. Sci. Stat. Comput.* 11:990-1028, 1990
- M. Cosnard, J. M. Muller, and Y. Robert, Parallel QR decomposition of a rectangular matrix, *Numer. Math.* 48:239-249, 1986
- M. Cosnard and Y. Robert, Complexity of parallel QR factorization, *J. ACM* 33:712-723, 1986
- A. Pothen and P. Raghavan, Distributed orthogonal factorization: Givens and Householder algorithms, *SIAM J. Sci. Stat. Comput.* 10:1113-1134, 1989