# First Homework Report

Ran Zhao

October 1, 2012

## 1 Algorithm Summary

The task of this algorithm is to factorize the given matrix by using the Jacobi algorithm(A=usv).The matrix s should be diagonalized. On each iteration of sweeping, I choose to use two traditional orthogonal matrixs:

$$u = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}$$

$$v = \begin{bmatrix} \cos\beta & \sin\beta \\ -\sin\beta & \cos\beta \end{bmatrix}$$

For an example, we intend to diagonalize a 2x2 matrix by making it times two orthogonal matrixs above:

$$A = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} a_{11} & a_12 \\ a_21 & a_22 \end{bmatrix} \begin{bmatrix} \cos\beta & \sin\beta \\ -\sin\beta & \cos\beta \end{bmatrix}$$

In order to diagonalize the matrix, we can get two equation after the matrix multiplication:

$$\begin{cases} \sin\alpha\cos\beta a_{11} + \cos\alpha\cos\beta a_{21} - \sin\alpha\sin\beta a_{12} - \cos\alpha\sin\beta a_{22} = 0 \\ \cos\beta\sin\beta a_{11} - sin\alpha\sin\beta a_{21} + cos\alpha\cos\beta a_{12} - sin\alpha\cos\beta a_{22} = 0 \end{cases}$$

After I simplify these two equations, I could get the two equations below:

$$\tan(\alpha+\beta) = \frac{a_{12}+a_{21}}{a_{22}-a_{11}} \qquad tan(\alpha-\beta) = \frac{a_{12}-a_{21}}{a_{22}+a_{11}}$$

After we solve the two equations above, we are able to get two orthogonal matrixs. Then we multiple each matrix to the origial one based on the left and right order. Eventually, we get the result matrix after the rotation by two matrixs. One iteration process is done. We will set an threshold value to the matrix. When the max non-diagonal value is smaller than the threshold value, then our sweep process is done. We can represent the process by equations. The Matrix A is a orginal matrix.s is the spectrum of the matrix A, u is the left singular vector of matrixA, v is the right singular vector of the matrix a.

$$A = (u_1)^T(u2)^T...(un)^T u_n...u_2u_1A(v_1)^T(v_2)^T...(v_n)^T v_n...v_2v_1$$

$$\begin{cases} s = u_n...u_2u_1A(v_1)^T(v_2)^T...(v_n)^T \\ u = (u_1)^T(u_2)^T...(u_n)^T \\ v = v_n...v_2v_1 \end{cases}$$

I use commutative of the matrix to calculate the u and the same as v:

$$(u_1)^T(u_2)^T...(u_n)^T = (u_n...u_2u_1)^T$$

# 2 Test Summary

Test Case I, $A_{i,j}$=sqrt(i*i+j*j)

| Dimension | Iteration | (Non-zero singular value) |
|---|---|---|
| 10 | 185 | 9 |
| 20 | 849 | 11 |
| 40 | 2807 | 14 |

Test Case II, $A_{i,j}$=i*i+j*j

| Dimension | Iteration | (Non-zero singular value) |
|---|---|---|
| 10 | 52 | 2 |
| 20 | 115 | 2 |
| 40 | 651 | 2 |

Based on the result of the test, I find that with the increase of the matrix dimension the number of sweep iterations is increase. In other words, there are more sweep operations needed to be done before we diagonlize the target matrix in high dimension matrix. Secondly, the non-zero singluar value of case I is bigger than the case II. This result demostrates that the function of case II is easier to represent in the space than function of case I. Thirdly, in order to diagnolize the same dimension matrix, there are much more iterations need in first case than in the second case. Thus,the convergence rate of the first case function was slower than the second case. I have checked that the reiliablity of the tests approach to zero. The algorithm works properly.

# 3 Future Work

In our case,we only disscus about the way of diagnolizing the square matrix. If the matrix is not square matrix, we should calculate A$A^T$and$A^T$A. Then calculate their eigenvalue and eigen vector in order to singluar value decompse the target matrix.