CSE6643 Numerical Linear Algebra HW4

November 25, 2016

1

Given $m \times m$ matrix A, $n \times n$ matrix B and $m \times n$ matrix C, the Lyapunov equation is

$$AX - XB = C$$

This is a system of mn linear equations with unknowns as entries of X. This exercise outlines a procedure to solve the Lyapunov equation.

- (1) Suppose the Schur factorization of A and B are known. Transform the equation $AX - XB = C$ to an equivalent system $A'Y - YB' = C'$ where A' and B' are upper triangular.
- (2) Design an algorithm (either by providing a pseudo code or MATLAB script) to find the entries of Y one at a time. What conditions on the eigenvalues of A and B guarantees that the system of questions is non-singular?
- (3) Solve for X.

(1)Solution: According to Schur factorization, we can obtain $A = QTQ^*$ and $B = PSP^*$, where T and S are upper triangular matrices. Therefore we can substitute the Schur factorization into the formula.

$$\begin{aligned} AX - XB &= QTQ^*X - XPSP^* = C \\ &= TQ^*X - Q^*XPSP^* = Q^*C \\ &= TQ^*XP - Q^*XPS = Q^*CP \end{aligned}$$

Let $A' = T$, $B' = S$, (A' and B' is upper triangular matrices) $C' = Q^*CP$ and $Y = Q^*XP$, then we can obtain:

$$A'Y - YB' = C'$$

(2)Solution:

$$A'Y - YB' = C' \Rightarrow TY - YS = D$$

$$\Rightarrow \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1m} \\ & t_{22} & \cdots & t_{2m} \\ & & \ddots & \vdots \\ & & & t_{mm} \end{bmatrix} \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mn} \end{bmatrix} - \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mn} \end{bmatrix} \begin{bmatrix} s_{11} & s_{12} & \cdots & t_{1n} \\ & s_{22} & \cdots & t_{2n} \\ & & \ddots & \vdots \\ & & & s_{nn} \end{bmatrix}$$

$$= \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mn} \end{bmatrix}$$

In the mth row of matrix D, we can obtain each entry of Y in row m, if we get the $y_{m,1}$, thus:

$$d_{m,1} = t_{m,m}y_{m,1} - y_{m,1}s_{1,1} \Rightarrow y_{m,1} = \frac{d_{m,1}}{(t_{m,m} - s_{1,1})}$$

$$d_{m,2} = t_{m,m}y_{m,2} - y_{m,1}s_{1,2} - y_{m,2}s_{2,2} \Rightarrow y_{m,2} = \frac{d_{m,2} + y_{m,1}s_{1,2}}{(t_{m,m} - s_{2,2})}$$

$$\vdots$$

In the similar manner, we can obtain the $m-1$th row of matrix D by doing the following implementations:

$$d_{m-1,1} = t_{m-1,m-1}y_{m-1,1} + t_{m-1,m}y_{m,1} - s_{11}y_{m-1,1} \Rightarrow y_{m-1,1} = \frac{d_{m-1,1} - t_{m-1,m}y_{m,1}}{(t_{m-1,m-1} - s_{1,1})}$$

$$d_{m-1,2} = t_{m-1,m-1}y_{m-1,2} + t_{m-1,m}y_{m,2} - y_{m-1,1}s_{1,2} - y_{m-1,2}s_{2,2} \Rightarrow y_{m-1,2} = \frac{d_{m-1,2} + y_{m-1,1}s_{1,2} - t_{m-1,m}y_{m,2}}{(t_{m-1,m-1} - s_{2,2})}$$

$$\vdots$$

Finally, we can obtain:

$$y_{i,j} = \frac{d_{i,j} + \sum_{k=1}^{j-1} y_{i,k}s_{k,j} - \sum_{l=i+1}^{m} t_{il}y_{l,j}}{t_{ii} - s_{jj}}$$

According to this logic, we can design the following algorithm to obtain matrix Y:

MATLAB Code:

```
function [X, Y]=p1(A, B, C)

    [Q,T]=schur(A); % Schur factorization to find similar matrix.

    [P,S]=schur(B); % Schur factorization to find similar matrix.

    D=Q'*C*P; % D: C after transformation.

    Y=zeros(size(C)); % Create an empty matrix Y

    [m,n]=size(C); % m and n values

    for i=m:-1:1
        for j=1:n
            Y(i,j)=(D(i,j)+sum(Y(i,1:(j-1))*S(1:(j-1),j))-
            sum(T(i,(i+1):m)*Y((i+1):m,j)))/(T(i,i)-S(j,j));
        end
    end
    X=inv(Q')*Y*inv(P); % Solve X
end
```

Claim: The system of equations is non-singular precisely when A and B have no eigenvalues in common.
Proof:

Assume A and B have eigenvalues λ in common. Then we have: $AX = \lambda X$ and $BX = \lambda X$. Substitute into the system of question, we can obtain: $AX - BX = \lambda X - X\lambda = 0$. Then X can't have a unique solutions. Therefore, equation is non-singular when A and B have no common eigenvalues.

Since matrix A and T are similar, the eigenvalues of A necessarily appear on the diagonal of T(Similarly with B and S). If we carefully observe the expression of entry $y_{i,j}$, its denominator is the difference between eigenvalues of A and B. If $(t_{i,i} - s_{j,j}) = 0$, that means $y_{i,j} = 0$. So in order to guarantee that the system of equations is non-singular precisely, we should guarantee full rank of matrix. when A and B have no eigenvalues in common.

(3)Solution:

Since $Y = Q^*XP$, Q and P can easily get by Schur factorization. Therefore as long as we get Y according to the algorithm in previous part, then we can solve X by $X = (Q^*)^{-1}YP^{-1}$.

2

Let A be a non-singular real matrix. Provide an algorithm that will produce a QL decomposition for A, that is, construct an orthogonal matrix Q and a lower triangular matrix L such that $A = QL$.

Solution:

Since A is a non-singular real matrix, assume A is $n \times n$ matrix, then we can derive QL decompostion according to the same way to derive QR decomposition by using Classical Gram-Schmidt algorithm.

$$A = QL \Rightarrow$$

$$\Rightarrow \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix} \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix}$$

$$\begin{aligned} a_1 &= l_{11}q_1 + l_{21}q_2 + \cdots + l_{n1}q_n \\ a_2 &= l_{22}q_2 + l_{32}q_3 + \cdots + l_{n2}q_n \\ &\vdots \\ a_{n-1} &= l_{n-1,n-1}q_{n-1} + l_{n,n-1}q_n \\ a_n &= r_{nn}q_n \end{aligned}$$

$$
\begin{aligned}
&\Rightarrow \\
q_n &= \frac{a_n}{l_{n,n}} \\
q_{n-1} &= \frac{a_{n-1} - l_{n,n-1} q_n}{l_{n-1,n-1}} \\
&\vdots \\
q_1 &= \frac{a_1 - \sum_{i=2}^{n} l_{i,1} q_i}{l_{1,1}} \\
\Rightarrow q_j &= \frac{a_j - \sum_{i=j+1}^{n} l_{i,j} q_i}{l_{j,j}}
\end{aligned}
$$

Also, it is evident that for coefficients l_{ij} is:

$$l_{ij} = q_i^* a_j \tag{1}$$

The coefficients l_{jj} are chosen for normalization:

$$|l_{jj}| = ||a_j - \sum_{i=j+1}^{n} l_{ij} q_i||_2 \tag{2}$$

At last, the algorithm embodied in the deduction of q_j and the equations (1) and (2) is the Gram-Schmidt iteration. The following MATLAB is the specific QL decomposition algorithm.

MATLAB Code:

```
function [Q, L] = p2(A)
    [m, n] = size(A);
    L = zeros(n, n);
    Q = zeros(m, n);
    for j = n : -1 : 1
        v = A(:, j);
        for i = j + 1 : n
            L(i, j) = Q(:, i)' * A(:, j);
            v = v - L(i, j) * Q(:, i);
        end
        L(j, j) = norm(v);
        Q(:, j) = v / L(j, j);
    end
end
```

3 Exercise 29.1

This five-part problem asks you to put together a MATLAB program that finds all the eigenvalues of areal symmetric matrix, using only elementary building blocks. It is not necessary to achieve optimal constant factors by exploiting symmetry or zero structure optimally. It is possible to solve the whole problem by a program about fifty lines long.

Answer:

(a) Function `T = tridiag(A)` is written in MATLAB file `tridiag.m`. Output matrix is symmetric and tridiagonal. After applying `A=hilb(4)`, the output `T` is:

```
 1.000000000000000  -0.650854139658888  -0.000000000000000  -0.000000000000000
-0.650854139658887   0.650585480093676   0.063911879959869   0.000000000000000
 0.000000000000000   0.063911879959868   0.025320143416558  -0.001165208041306
 0.000000000000000                   0  -0.001165208041306   0.000284852680241
```

(b) Function `Tnew = qralg(T)` is written in MATLAB file `qralg.m`. For the QR factorization at each step, I use MATLAB's command `qr` and assure symmetry and tridiagonality at each step.

(c) A driver program for question c is written in MATLAB file `driver_c.m`. After applying `A=hilb(4)`, the output eigenvalue set is:

```
1.500214280059243
0.169141220221450
0.006738273605761
0.000096702304023
```

Figure 1 is a semilogy plot of $|t_{m,m-1}|$ values as a function of the number of QR factorization.(`A=hilb(4)`)

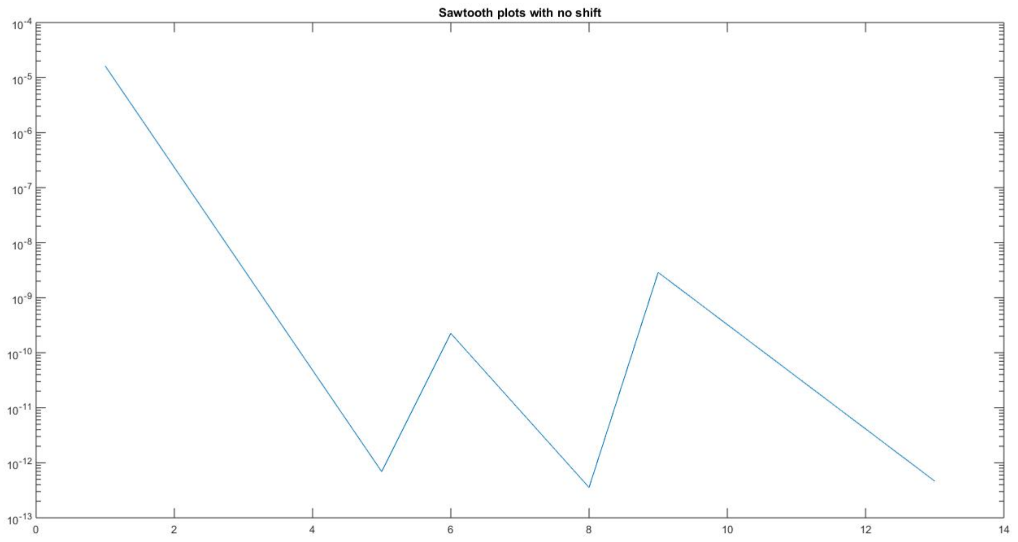

Figure 1: Sawtooth plot of $|t_{m,m-1}|$ with no shift

(d) The function who uses the Wilkinson shift at each step is written in MATLAB file `qralg_new.m`. And if we run the MATLAB file `driver_d.m`. It will generate the result by invoking algorithm `qralg_new.m`. The new sawtooth plot fot `A=hilb(4)` is showed in figure 2.

(e) For matrix `A=diag(15:-1:1) + ones(15:15)`, two sawtooth plots corresponding to shift and no shift are showed in figure 3 and 4.

- For the earlier matrix, since the size of A is small, it is hard to tell what the specific converge pattern is. However, from figure 1 and 2, we can see the value of $|t_{m,m-1}|$ is converging at least. Besides, for the bigger matrix (15×15), I eliminate some outlier points and keep the main trend of the graph so that I can obtain the following graphs.

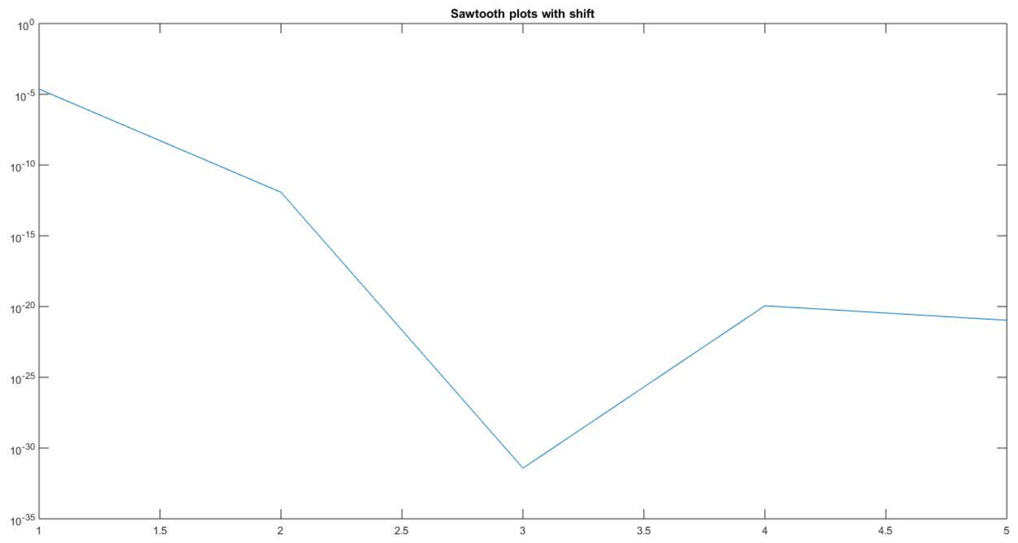

Figure 2: Sawtooth plot of $|t_{m,m-1}|$ with shift

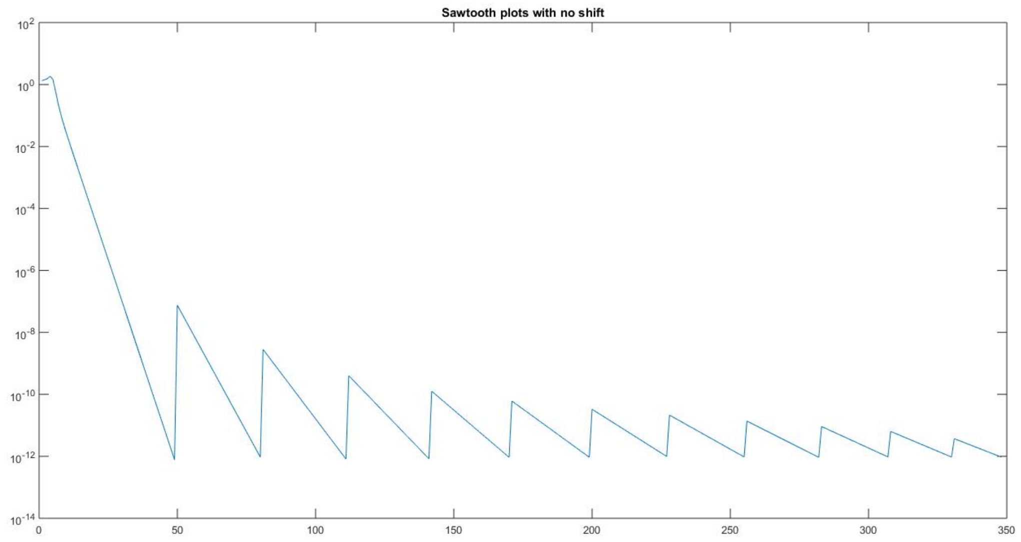

Figure 3: Sawtooth plot of $|t_{m,m-1}|$ with no shift

- No matter what size it is for both matrices, we can easily know that it takes bigger times of QR factorization to converge with no shift while it only takes smaller amount of QR factorization to converge.

- Suppose that the sequence x_k converges to the number L. We say that this sequence converges linearly to L, if $\frac{|x_{n+1}-x_*|}{|x_n-x_*|} \approx \mu$, for some $\mu < 1$. When we compute the rate of convergence about algorithm with no shift by using this formula, since $|t_{m,m-1}|$ converges to 0, $\frac{|t_{m,m-1}|}{|t_{m-1,m-2}|} \approx 0.922742061 < 1$. Thus we can say the convergence of algorithm with no shift is linear. (The

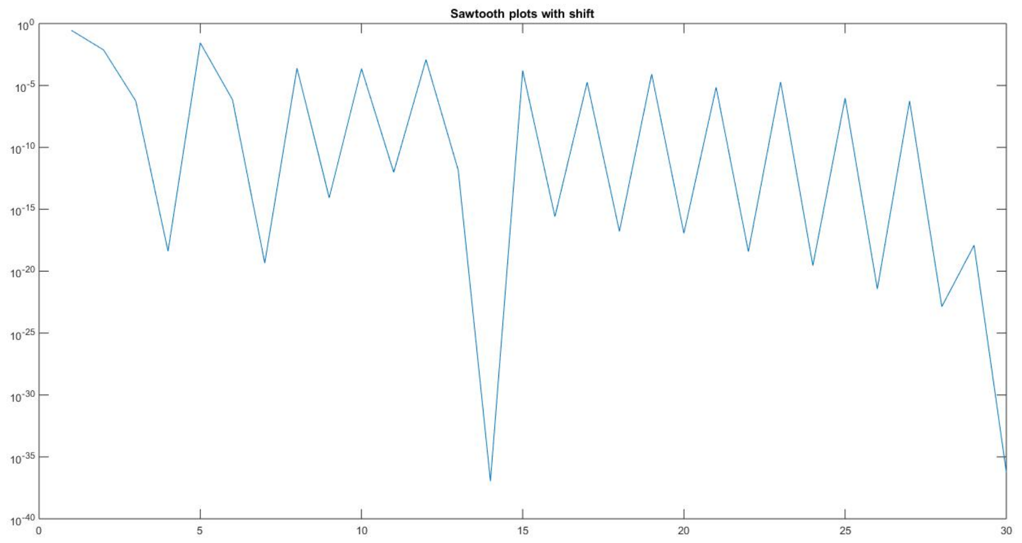

Figure 4: Sawtooth plot of $|t_{m,m-1}|$ with shift

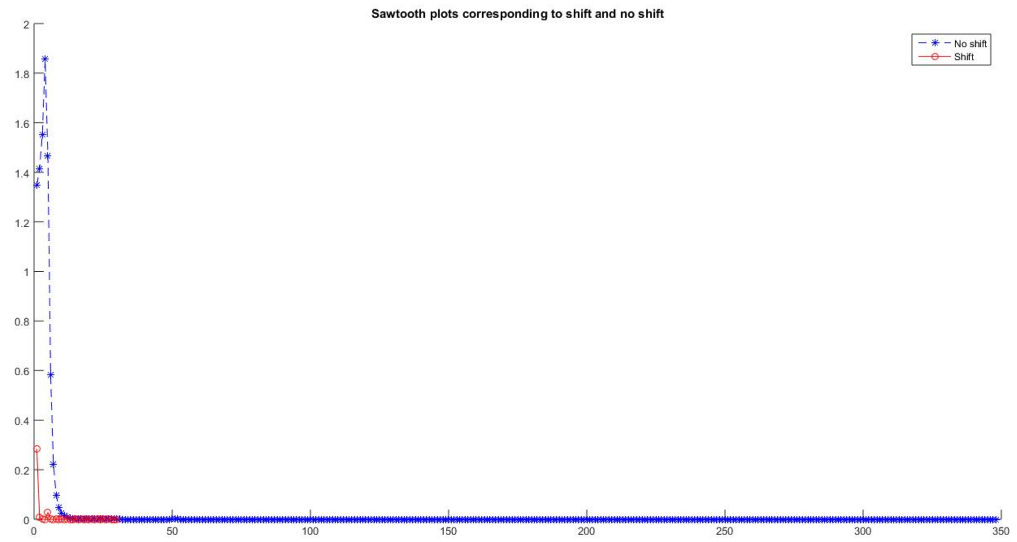

Figure 5: Comparison of two sawtooth plots of $|t_{m,m-1}|$ corresponding to shift and no shift

convergence result is showed in figure 6.)

- If the sequence converges, and $\mu = \mu_k$ varies from step to step with $\mu_k \to 0$ for $k \to \infty$, then the sequence is said to converge superlinearly. When we compute the rate of convergence about algorithm with shift by using this formula, the $\mu \to 0$ for $k \to \infty$. Thus we can say the convergence of algorithm with shift is superlinear. (The convergence result is showed in figure 7.)

- It is meaningful to speak of a certain "number of QR iterations per eigenvalue". Since QR

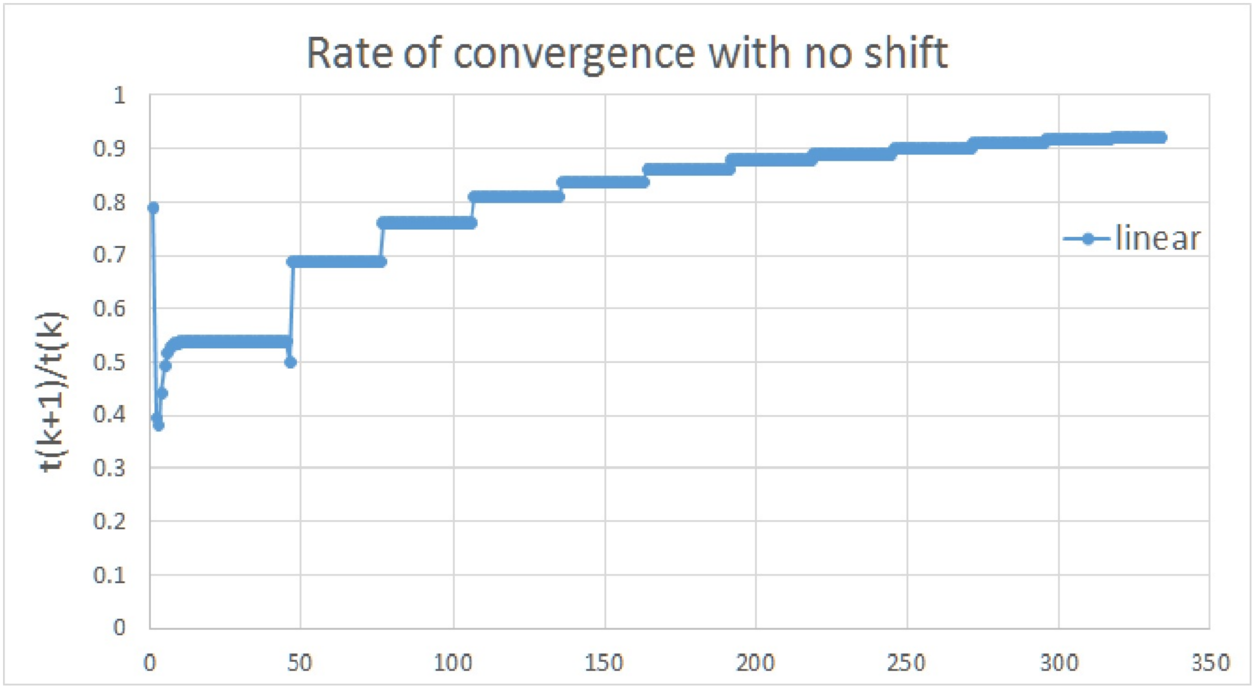

Figure 6: Rate of convergence with no shift corresponding to $|t_{m,m-1}|/|t_{m-1,m-2}|$

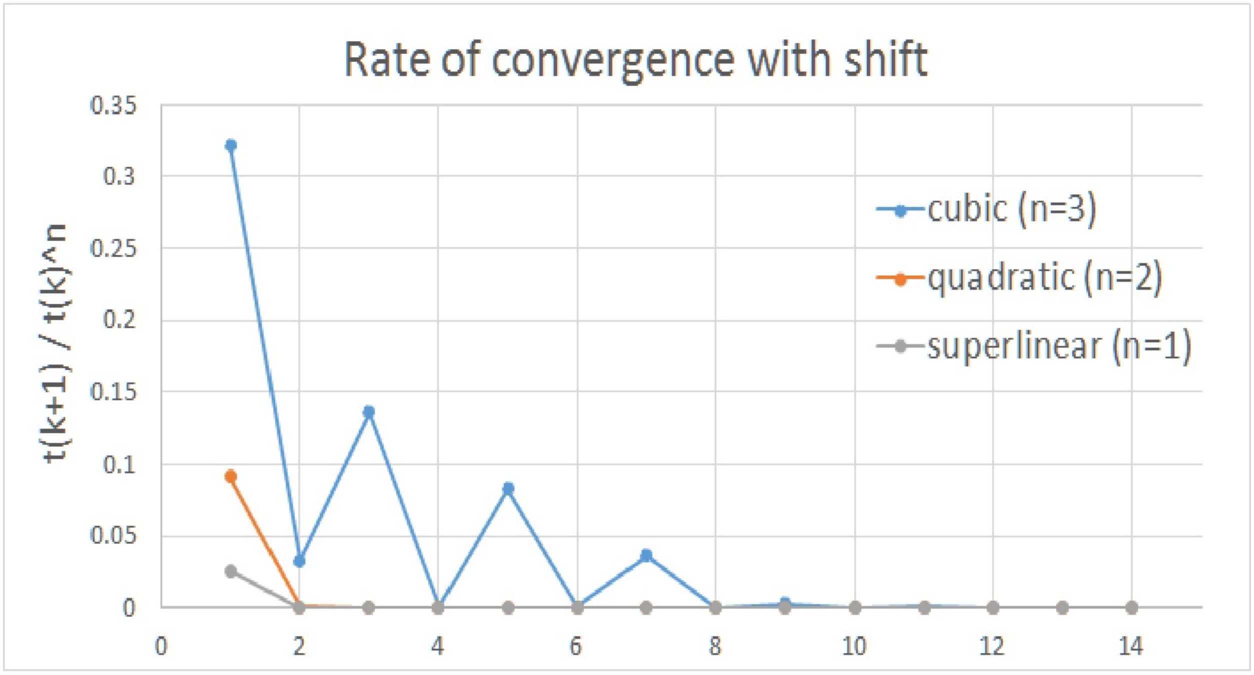

Figure 7: Rate of convergence with shift corresponding to $|t_{m,m-1}|/|t_{m-1,m-2}|$

factorization is the main operation of both algorithms. By evaluating number of QR iteration per eigenvalue, we can compare the performance of algorithms to calculate eigenvalues.