Computing Moore-Penrose Inverses of Toeplitz Matrices by Newton's Iteration *

Yimin Wei† Jianfeng Cai‡ Michael K. Ng§

Abstract

We modify the algorithm of [1], based on Newton's iteration and on the concept of ϵ-displacement rank, to the computation of the Moore-Penrose inverse of a rank-deficient Toeplitz matrix. Numerical results are presented to illustrate the effectiveness of the method.

Keywords: Newton's iteration, Moore-Penrose inverse, Toeplitz matrix

AMS subject classification: 15A09, 65F20

1 Introduction

Let A be an $m \times n$ Toeplitz matrix, i.e., $a_{i,j} = a_{i-j}$ for $i = 1, \cdots, m$ and $j = 1, \cdots, n$. In [1], Bini, Codevico and Van Barel have used Newton's iteration and the concept of ϵ-displacement rank to the computation of the generalized inverse $A^\dagger$ of A. In their papers, they assume that Toeplitz matrices A have full rank. The main contribution of this paper is to modify their algorithm based on Newton's iteration and on the concept of ϵ-displacement rank, to compute the Moore-Penrose inverse of a rank-deficient Toeplitz matrix.

The outline of this paper is as follows. In Section 2, we review displacement rank and ϵ-displacement rank. In Section 3, we introduce our modified Newton's iteration and present a simple residual error bound. We remark that the residual error bound contains the errors due to Newton's iterations and the errors in the approximation of the displacement representation

*Project supported by the National Natural Science Foundation of China and Hong Kong Research Grants Council Grant No. HKU 7130/02P.

†Department of Mathematics, Fudan University, Shanghai, 200433, P. R. China. E-mail: ymwei@fudan.edu.cn.

‡Institute of Mathematics, Fudan University, Shanghai, 200433, P. R. China.

§Department of Mathematics, The University of Hong Kong, Pokfulam Road, Hong Kong. E-mail: mng@maths.hku.hk.

of the Moore-Penrose inverse of a Toeplitz matrix. Finally, numerical results are reported to illustrate the convergence of our method.

2 Displacement rank and ϵ-displacement rank

Let us denote

$$Z_n \equiv \begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix} \in R^{n\times n},$$

and set $L(x)$ and $U(y)$ to be the lower triangular and the upper triangular Toeplitz matrices defined by the first column x and the first row y^T respectively. The matrix $L(x)$ is of dimension $m \times n$ if $m \geq n$ or $m \times m$ if $m < n$. Analogously, the matrix $U(x)$ is of dimension $n \times n$ if $m \geq n$ or $m \times n$ if $m < n$. We introduce the displacement operator

$$\Delta(A) = Z_m A - A Z_n$$

for $A \in R^{m\times n}$. We call the rank of $\Delta(A)$ the displacement rank of A [2] and denote it with $drk(A)$. It is easy to show that $drk(A)$ is at most 2 for a Toeplitz matrix $A \in R^{m\times n}$.

Theorem 1 ([3]) *Let $A \in R^{m\times n}$ has displacement rank k and $g_i \in R^m$, $h_i \in R^n$ for $i = 1, \cdots, k$ such that $\Delta(A) = \sum_{i=1}^k g_i h_i^T$. Then $A = L(Ae_1) + \sum_{i=1}^k L(g_i)U(Z_n h_i)$, where Ae_1 denotes the first column of A.*

The above theorem implies that one can compute the the product Ax by means of $2k + 2$ FFTs, $2k + 1$ convolutions of length $2m$ and $2k + 1$ FFTs, k convolutions of length $2n$, where k is the displacement rank of A. In order to reduce the computational cost per step of the modified Newton's iteration presented in the next section, we use the concept ϵ-displacement rank introduced in [4].

Definition 1 *For a given $\epsilon > 0$ define the ϵ-displacement rank of a matrix A as*

$$drk_\epsilon(A) = \min_{\|E\|\leq\epsilon} \operatorname{rank}(\Delta(A) + E).$$

With the help of the singular value decomposition (SVD) of $\Delta(A)$, one can easily get $drk_\epsilon(A)$.

Theorem 2 *Let $\Delta(A) = U\Sigma V^T = \sum_{i=1}^k \sigma_i u_i v_i^T$ ($\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k > 0$) be the SVD of $\Delta(A)$. Let ϵ be such that $\epsilon < \sigma_1$. Then $drk_\epsilon = r$ if and only if $\sigma_r > \epsilon \geq \sigma_{r+1}$.*

The orthogonal displacement representation (odr) [4] of A is defined as follows:

$$A = L(Ae_1) + \sum_{i=1}^{k} \sigma_i L(u_i) U(Z_n v_i)$$

and the corresponding orthogonal displacement generator (odg) is given by the quadruple (Ae_1, U, σ, V) where $\sigma = (\sigma_1, \cdots, \sigma_k)$. Given $0 < \epsilon < \sigma_1$, if $drk_\epsilon = r$, we can get an approximate A_ϵ to A:

$$A_\epsilon = L(Ae_1) + \sum_{i=1}^{r} \sigma_i L(u_i) U(Z_n v_i). \tag{1}$$

We call the above expression the approximate orthogonal displacement representation (aodr) of A and the associated generator is called approximate orthogonal displacement generator (aodg) denoted by $(Ae_1, \widehat{U}, \widehat{\sigma}, \widehat{V})$.

Finally, let us introduce the operator $trunc_\epsilon(\cdot)$ defined on the sets of orthogonal displacement generators as follows:

$$trunc_\epsilon((a, U, \sigma, V)) = (a, \widehat{U}, \widehat{\sigma}, \widehat{V}), \tag{2}$$

where

$$\widehat{\sigma} = (\sigma_1, \cdots, \sigma_{\widehat{k}}), \quad \widehat{U} = [u_1, \cdots, u_{\widehat{k}}], \quad \widehat{V} = [v_1, \cdots, v_{\widehat{k}}]$$

and

$$\sigma_{\widehat{k}} > \epsilon \geq \sigma_{\widehat{k}+1}.$$

Theorem 3 ([4]) *Let $A \in R^{m\times n}$, $r = drk_\epsilon(A) \leq drk(A) = k$, $\sigma_1, \cdots, \sigma_k$ be the singular values of $\Delta(A)$ and A_ϵ be defined by (1). Then*

$$\|A - A_\epsilon\| \leq \sqrt{nm} \sum_{i=r+1}^{k} \sigma_i \leq \sqrt{nm}(k-r)\epsilon.$$

We remark that there are many other displacement operators for Toeplitz matrices [4, 3, 11, 6]. For example,if A is a square matrix, we can define

$$\Delta^+(A) = C^+A - AC^-, \quad \Delta^-(A) = C^-A - AC^+,$$

where $C^+ = Z_n + e_1 e_n^T$ and $C^- = Z_n - e_1 e_n^T$. It is easy to check $\text{rank}(\Delta^+(A)) \leq 2$ and $\text{rank}(\Delta^-(A)) \leq 2$. Similarly, there exist theorems for such displacement operators corresponding to Theorem 3, see [4].

3 Modified Newton's Iteration

In this section, we give the algorithm to compute the Moore-Penrose inverse of A.

Definition 2 *Let $A \in R^{m\times n}$, the unique solution $X \in R^{n\times m}$ for the following four equations:*

$$AXA = A \quad XAX = X \quad (AX)^T = AX \quad \text{and} \quad (XA)^T = XA$$

is called the Moore-Penrose inverse of A, and we denote it $A^\dagger$.

There are many algorithms, such as recurrent neural networks, successive matrix squaring algorithm and singular value decomposition to compute $A^\dagger$ [5, 7, 8, 9], including Newton's iteration [1, 5, 6]. The classical Newton's iteration is

$$X_{i+1} = 2X_i - X_iAX_i, \quad i = 0, 1, 2, \ldots.$$

Newton's iteration for matrix inversion was proposed by Schultz in 1933 and was well studied [5]. Newton iteration is simple to describe and to analyze and is numerically stable. Since it is rich in matrix-matrix multiplication, it can be efficiently implemented on parallel computers.

For a rank-deficient matrix, if we set $X_0 = \alpha A^T$, where α is a proper scalar, then X_i will converge to the Moore-Penrose inverse $A^\dagger$ of A [10]. If A is a Toeplitz matrix, the displacement rank $A^\dagger$ is at most 4 [11], but the displacement rank of X_i grows exponentially with i until the value of n is reached. The computational cost at each Newton's iteration step increases significantly while i grows. In order to make the Newton's iteration a useful tool for computing the Moore-Penrose inverse of a Toeplitz matrix, we use the ϵ-displacement rank to control the growth of the displacement of X_i. We will present two modified Newton's iterations to compute $A^\dagger$.

3.1 Modified Newton's Iteration I

In our first modified Newton's iteration, we set $X_0 = \alpha A^TAA^T$. It easy to check that it satisfies $\|AA^\dagger - \alpha AA^TAA^T\| < 1$ and then naturally satisfies $\|A^\dagger A - \alpha A^TAA^TA\| < 1$ too. The parameter α is easy to choose. In fact, we set $\alpha = 1/\rho$, where ρ is the spectrum radius $\rho(AA^TAA^T) = \rho(A^TAA^TA)$, which can be computed by a few steps of power iteration [12] in a very low computational cost.

We note that each X_i has a factor A^T on the left and the right sides, i.e., $X_i = A^TY_iA^T$. Therefore, we rewrite Newton's iteration as follows:

$$Y_0 = \alpha A, \quad X_i = A^TY_iA^T \quad \text{and} \quad Y_{i+1} = 2Y_i - Y_iA^TAA^TY_i, \quad i = 0, 1, 2, \ldots.$$

Our modified Newton's iteration I is based on the above formula and defined by the following recurrence:

$$W_i = 2Y_i - Y_iA^TAA^TY_i, \quad Y_{i+1} = trunc_{\epsilon_i}(W_i), \quad \text{and} \quad X_{i+1} = A^TY_{i+1}A^T \tag{3}$$

where $trunc_\epsilon(\cdot)$ is defined (2).

We compute and store the odg of Y_i instead of Y_i itself through out the iteration. If the odg of Y_i is $(y, U_{Y_i}, \sigma_{Y_i}, V_{Y_i})$, then $\Delta(Y_i) = U_{Y_i}\Sigma_{Y_i}V_{Y_i}^T$, where Σ_{Y_i}=diag(σ_{Y_i}). Observe that

$$\Delta(W_i) = 2\Delta(Y_i) - \Delta(Y_i)A^TAA^TY_i - Y_i\Delta(A^TAA^T)Y_i - Y_iA^TAA^T\Delta(Y_i), \tag{4}$$

we have $drk(W_i) \le k + 2drk(Y_i)$ where $k = drk(A^TAA^T) \le 6$.

The odg of A^TAA^T is (a, U_A, σ_A, V_A),that is $\Delta(A^TAA^T) = U_A\Sigma_AV_A^T$, where $\Sigma_A = diag(\sigma_A)$. Rewriting (4) into matrix form, we obtain

$$\Delta(W_i) = [\, U_{Y_i} \quad Y_iA^TAA^TU_{Y_i} \quad Y_iU_A \,] \begin{bmatrix} 2\Sigma_{Y_i} & 0 & -\Sigma_{Y_i} \\ -\Sigma_{Y_i} & 0 & 0 \\ 0 & -\Sigma_A & 0 \end{bmatrix} \begin{bmatrix} V_{Y_i}^T \\ V_A^TY_i \\ V_{Y_i}^TA^TAA^TY_i \end{bmatrix}. \tag{5}$$

We denote the above right three matrices $U_{W_i}, \Sigma_{W_i}, V_{W_i}^T$ respectively. Thus,the odg of $Y_{i+1} = trunc_{\epsilon_i}(W_i)$ can be computed by the following algorithm:

Algorithm 1 Compute the odg of Y_{i+1} (the aodg of $W_i = 2Y_i - Y_iA^TAA^TY_i$)

Input: The odg of Y_i $(y_i, U_{Y_i}, \sigma_{Y_i}, V_{Y_i})$ and the truncation value ϵ_i.

Output: The odg of Y_{i+1} $(y_{i+1}, U_{Y_{i+1}}, \sigma_{Y_{i+1}}, V_{Y_{i+1}})$.

Computation:

(i) Compute $y_{i+1} = 2y_i - Y_iA^TAA^Ty_i$.

(ii) Set Σ_{Y_i}= diag(σ_{Y_i}) and compute the matrices $U_{W_i}, V_{W_i}, \Sigma_{W_i}$.

(iii) Compute the QR decompositions $U_{W_i} = Q_1R_1$ and $V_{W_i} = Q_2R_2$.

(iv) Compute the SVD of $R_1\Sigma_{W_i}R_2^T = U\Sigma V$,where $\Sigma = diag(\sigma_1, \cdots, \sigma_k)$ and determine r such that $\sigma_{r+1} \le \epsilon_i < \sigma_r$.

(v) Set $\sigma_{Y_{i+1}} = (\sigma_1, \cdots, \sigma_r)$, $U_{Y_{i+1}} = Q_1U(:, 1:r)$ and $V_{Y_{i+1}} = Q_2V(:, 1:r)$.

It is easy to see that the computational cost of the above algorithm is about $O(hk)$ FFTs, $O((2h+k)^2(n+m))$ and $O((2h+k)^3)$ extra flops, where $h = drk(Y_i)$ and $k = drk(A^TAA^T)$.

The following theorem analyzes the convergence of our modified Newton iteration for a certain choice of ϵ_i.

Theorem 4 *Let $X_0 = \alpha A^TAA^T$ be such that $\|AA^\dagger - AX_0\| \le 1 - \theta$. Let $R_i = AA^\dagger - AX_i$ and $\overline{R_i} = A^\dagger A - X_iA$ be the residual sequences. If $\epsilon_i = \min(\|R_i\|^2, \|\overline{R_i}\|^2)\theta/(2\sqrt{nm}(2h_i + k)\|A\|^3)$,where $h_i = drk(Y_i)$. Then it holds $\|R_i\|^2 \le (1-\theta/2)^{2^i}$ and $\|\overline{R_i}\|^2 \le (1-\theta/2)^{2^i}$.*

Proof: We denote $E_i = W_i - Y_{i+1}$, thus,

$$\|E_i\| \le \sqrt{nm}(2h_i + k)\epsilon_i = \min(\|R_i\|^2, \|\overline{R_i}\|^2)\theta/(2\|A\|^3).$$

We can prove

$$\begin{aligned}
R_{i+1} &= AA^\dagger - AX_{i+1} = AA^\dagger - AA^TY_{i+1}A^T \\
&= AA^\dagger - AA^TW_iA^T + AA^TE_iA^T \\
&= AA^\dagger - AA^T(2Y_i - Y_iA^TAA^TY_i)A^T + AA^TE_iA^T \\
&= AA^\dagger - AA^TY_iA^T - AA^TY_iA^T + AA^TY_iA^TAA^TY_iA^T + AA^TE_iA^T \\
&= AA^\dagger - AA^\dagger AA^TY_iA^T - AA^TY_iA^TAA^\dagger + AA^TY_iA^TAA^TY_iA^T + AA^TE_iA^T \\
&= (AA^\dagger - AA^TY_iA^T)^2 + AA^TE_iA^T \\
&= R_i^2 + AA^TE_iA^T.
\end{aligned}$$

Therefore,

$$\|R_{i+1}\| \le \|R_i\|^2 + \|A\|^3\|E_i\| \le \|R_i\|^2(1+\theta/2).$$

It follows that

$$\|R_i\| \le ((1-\theta)(1+\theta/2))^{2^i} \le (1-\theta/2)^{2^i}.$$

The case for the residual sequence $\{\overline{R_i}\}$ can be proved analogously. □

The above theorem reveals the quadratic convergence of our modified Newton's iteration for the certain ϵ_i. However, in practice the bound is too pessimistic and may lead to an unnecessary large growth of the displacement rank. A more realistic strategy is to choose a larger ϵ_i which may make the sequence $\{X_i\}$ converges to $A^\dagger$ slower.

On the other hand, it is expensive to compute the residual sequence since $A^\dagger$ is not known in advance. A cheap method is to compute the $res_I(X_i)$ defined by Definition 2

$$res_I(X_i) = max\{\|(A-AX_iA)e_1\|, \|(X_i-X_iAX_i)e_1\|, \|(AX_i-(AX_i)^T)e_1\|, \|(X_iA-(X_iA)^T)e_1\|\}, \tag{6}$$

where e_1 is the first column of the identity matrix. We note that each of the above term can be computed efficiently by using a few FFTs. We use this strategy in our numerical experiments.

Here we summarize the modified Newton's iteration I.

Algorithm 2 `Modified Newton's iteration I.`

`Input:` Integers n, m, a residual error bound ϵ, the first row and column vectors of the Toeplitz matrix A.

`Output:` An approximation X of $A^\dagger$ given in terms of A^TYA^T which Y is given by its odg (y, U_Y, σ_Y, V_Y).

`Computation:`

(i) Compute the odg of A^TAA^T

(ii) Choose α such that $\|AA^\dagger - \alpha AA^TAA^T\| < 1$, compute the odg of $Y_0 = \alpha A$ and set $i = 0$

(iii) Determine a ϵ_i and compute the odg of Y_{i+1} by means Algorithm 1

(iv) Set $i = i + 1$. Let $X_i = A^TY_iA^T$. If the residual $res_I(X_i) > \epsilon$ in (6), then goto (iii), otherwise output the result.

3.2 Modified Newton's Iteration II

Our second modified Newton's iteration is based on the following theorem [10]:

Theorem 5 *Let $A^{(1,3)}$ be a generalized inverse which satisfies the 1st and the 3rd equations in Definition 2 and $A^{(1,4)}$ be a generalized inverse which satisfies the 1st and the 4th equations in Definition 2 for $A \in R^{m\times n}$, then*

$$A^\dagger = A^{(1,4)}AA^{(1,3)}. \tag{7}$$

In our modified Newton's iteration II, we compute $A^{(1,3)}$ and $A^{(1,4)}$ respectively by means of Newton's iteration and then obtain $A^\dagger$ by (7).

We set $X_0 = \alpha A^T$ where α is a scalar which satisfies $\|AA^\dagger - \alpha AA^T\| < 1$. Similar to Method I, we choose $\alpha = 1/\rho$ where ρ is the spectrum radius of AA^T. Therefore, each term X_i has the form Y_iA^T. Then the classical Newton's iteration can be rewritten as follows:

$$Y_0 = \alpha I, \quad X_i = Y_iA^T, \quad \text{and} \quad Y_{i+1} = 2Y_i - Y_iA^TAY_i.$$

We truncate Y_i at each step

$$W_i = 2Y_i - Y_iA^TAY_i, \quad Y_{i+1} = trunc_{\epsilon_i}(W_i), \quad \text{and} \quad X_{i+1} = Y_{i+1}A^T. \tag{8}$$

The following theorem states that the iteration (8) with certain chosen ϵ_i converges quadratically to $A^{(1,3)}$.

Theorem 6 *Let $X_0 = \alpha A^T$ be such that $\|AA^\dagger - AX_0\| \le 1 - \theta$.Let $R_i = AA^\dagger - AX_i$ be the residual sequence. If $\epsilon_i = \|R_i\|^2\theta/(2\sqrt{nm}(2h_i + k)\|A\|^2)$, where $h_i = drk(Y_i)$. Then it holds $\|R_i\|^2 \le (1 - \theta/2)^{2^i}$.*

Proof: We denote $E_i = W_i - Y_{i+1}$, thus we have

$$\|E_i\| \le \sqrt{nm}(2h_i + k)\epsilon_i = \|R_i\|^2\theta/(2\|A\|^2).$$

We can show that

$$\begin{aligned} R_{i+1} &= AA^\dagger - AX_{i+1} = AA^\dagger - AY_{i+1}A^T \\ &= AA^\dagger - AW_iA^T + AE_iA^T \\ &= AA^\dagger - A(2Y_i - Y_iA^TAY_i)A^T + AE_iA^T \\ &= AA^\dagger - AY_iA^T - AY_iA^T + AY_iA^TAY_iA^T + AE_iA^T \\ &= AA^\dagger - AA^\dagger AY_iA^T - AY_iA^TAA^\dagger + AY_iA^TAY_iA^T + AE_iA^T \\ &= (AA^\dagger - AY_iA^T)^2 + AE_iA^T \\ &= R_i^2 + AE_iA^T. \end{aligned}$$

Therefore,

$$\|R_{i+1}\| \le \|R_i\|^2 + \|A\|^2\|E_i\| \le \|R_i\|^2(1+\theta/2).$$

It follows that

$$\|R_i\| \le ((1-\theta)(1+\theta/2))^{2^i} \le (1-\theta/2)^{2^i}.$$

□

In order to obtain $A^{(1,4)}$, we set $X_0 = \alpha A^T$, which is the same as the initial value in the iteration to obtain $A^{(1,3)}$. We extract a factor A^T on the left side of the iteration sequence X_i. Hence, we iterate as follows:

$$W_i = 2Y_i - Y_iAA^TY_i, \quad Y_{i+1} = trunc_{\epsilon_i}(W_i), \quad \text{and} \quad X_{i+1} = A^TY_{i+1}. \tag{9}$$

Similarly, we have the following convergent theorem.

Theorem 7 *Let $X_0 = \alpha A^T$ be such that $\|A^\dagger A - X_0A\| \le 1-\theta$. Let $R_i = A^\dagger A - X_iA$ be the residual sequence. If $\epsilon_i = \|R_i\|^2\theta/(2\sqrt{nm}(2h_i+k)\|A\|^2)$, where $h_i = drk(Y_i)$. Then it holds $\|R_i\|^2 \le (1-\theta/2)^{2^i}$.*

Here we summarize the modified Newton's iteration II.

Algorithm 3 `Modified Newton's iteration II.`

`Input:` Integers n, m, a residual error bound ϵ, the first row and column vectors of the Toeplitz matrix A.

`Output:` An approximation X of $A^\dagger$ given in terms of $A^TY'AYA^T$ which Y and Y' are given by their odgs.

`Computation:`

(i) Approximate an element YA^T in $A^{(1,3)}$ analogously to Algorithm 2 such that $\|(A - AYA^TA)e_1\| < \epsilon$ and $\|(AYA^T - (AYA^T)^T)e_1\| < \epsilon$.

(ii) Approximate an element A^TY' in $A^{(1,4)}$ analogously to Algorithm 2 such that $\|(A - AA^TY'A)e_1\| < \epsilon$ and $\|(A^TY'A - (A^TY'A)^T)e_1\| < \epsilon$.

(iii) Output the result X by (7) and compute the residual error $res_I(X)$ in (6).

We remark that for the displacement operator $\Delta^+(A)$ and $\Delta^-(A)$, Algorithms 1, 2 and 3 can be simplified because we do not require to compute the first column of Y_i at each step.

4 Numerical experiments

In this section, we perform numerical experiments on the following singular square Toeplitz matrices: the first column of the matrix is $(1, 1/2, \cdots, 1/(n-1), 1)^T$ and the first row is chosen such that the last column is same as the first column. The Moore-Penrose inverses can be expressed by the formula :

$$A^\dagger = \begin{bmatrix} I \\ e_1^T \end{bmatrix} (I + e_1e_1^T)^{-1}C^{-1}(I + e_1e_1^T)^{-1} \begin{bmatrix} I & e_1 \end{bmatrix},$$

where C is a circulant matrix whose first column is given by $(1, 1/2, \cdots, 1/(n-1))^T$, I is the $(n-1)$-by-$(n-1)$ identity matrix, and e_1 is the first column of I.

For our modified Newton's iteration I, the truncation value ϵ_i is set to be $res_I(X_i)/\|A\|^4$. In Table 1, we report the number of iterations $Nstep$ required for convergence. The maximum drk $Mdrk$ of Y_i and the sum $Sdrk$ of the drk of Y_i are also reported. According to Table 1, we see that the quantities $Nstep$, $Mdrk$ and $Sdrk$ grow slowly when n increases.

For our modified Newton's iteration II, we need two algorithms to compute the Moore-Penrose inverse of A. In the two algorithms, we set the truncation value $res_{II}(Y_i)/\|A\|^3$, where $res_{II}(Y_i) = \max\{\|(A - AYA^TA)e_1\|, \|(AYA^T - (AYA^T)^T)e_1\|\}$. In Table 2, we report the total number $Nstep$ of iterations required by two algorithms, the maximum drk $Mdrk$ and the sum $Sdrk$ of drk in the two algorithms. Again we find that when n increases, the quantities $Nstep$, $Mdrk$ and $Sdrk$ grow slowly.

In Tables 1 and 2, we also list the error $\|A^\dagger - X\|$ between the Moore-Penrose inverse $A^\dagger$ and the computed solution X. We see from the tables that $res(X)$ and $\|A^\dagger - X\|$ are about the same. These results demonstrate that our proposed method can compute quite good estimate of the Moore-Penrose inverse. We also note that the $Mdrk$ (the maximum drk of Y_i) and $Sdrk$ in the modified Newton's Method II are smaller than those in the modified Newton's Method I. Also the $Nstep$ of Method II is slightly larger than that of Method I. According to the tests, the computational times required by Method II are about half less times than the times required by Method I. We remark that the computational times required by the original

Newton's iteration listed in Table 3 are significantly greater than those required by the proposed algorithms especially when n is large. The memory requirement of the original Newton's iteration is also very large, for instance, there is not enough memory for the calculation when $n = 4096$.

To illustrate the convergence of the method, we give in Figures 1, 2 and 3 the convergence of the Newton iterations I and II for different n. For the Newton iterations II, we require to determine two generalized inverses $A^{(1,3)}$ and $A^{(1,4)}$. We see in the figures that the proposed method converges very quickly (cf. Theorems 4 and 6), especially when the iterates are close to the solutions. To further illustrate the convergence of the method, we show $\|A^\dagger - X_i\|$ in Figure 4. It is clear that such errors decrease very quickly as iterations increase.

As a conclusion, the main contribution of this paper is to modify the algorithm of [1], based on Newton's iteration and on the concept of ϵ-displacement rank, to the computation of the Moore-Penrose inverse of a rank-deficient Toeplitz matrix. Numerical results are presented to demonstrate the effectiveness of the proposed method.

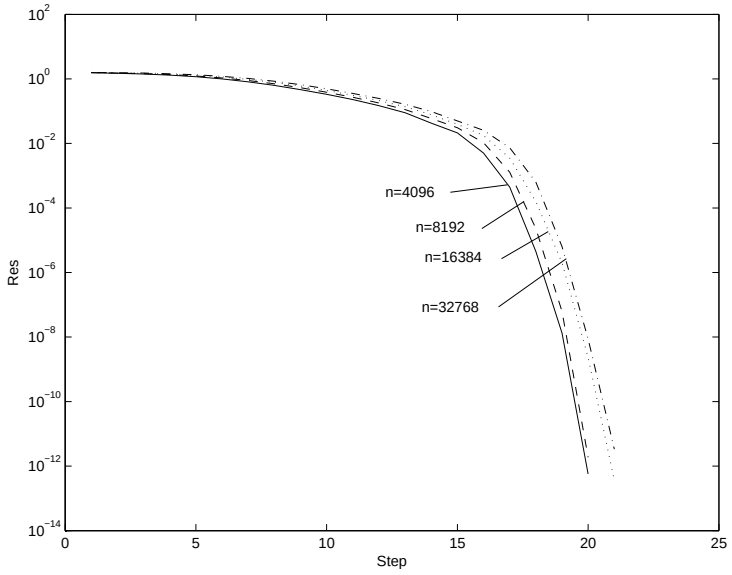

Figure 1: Convergence of the modified Newton's iterations I.

References

[1] D. Bini, G. Codevico and M. Van Barel, Solving Toeplitz least squares problems by means of Newton's iteration, Numer. Algo., to appear.

[2] T. Kailath an A. Sayed, Displacement structure:Theory and applications, SIAM Review, 37 (1995) 297-386.

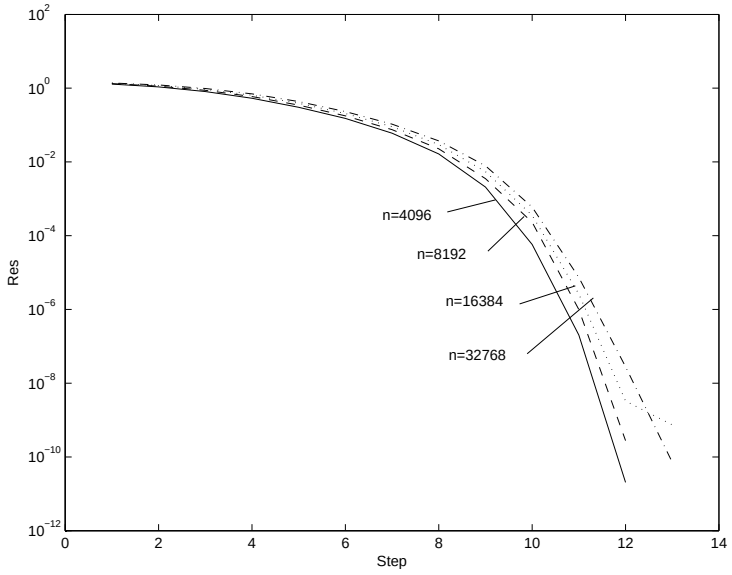

Figure 2: Convergence of the modified Newton's iteration II for $A^{(1,3)}$.

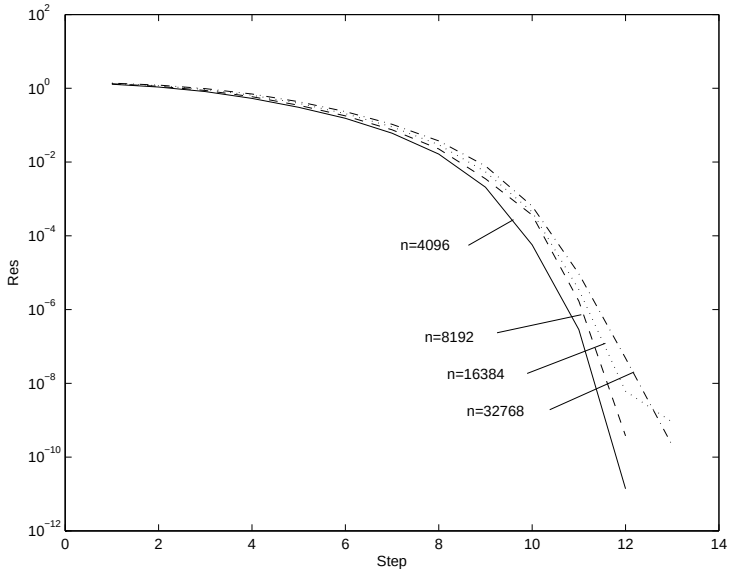

Figure 3: Convergence of the modified Newton's iteration II for $A^{(1,4)}$.

n	$Nstep$	$Mdrk$	$Sdrk$	$res(X)$	$\|A^\dagger - X\|$	time (seconds)
32	16	11	103	1.2e-013	6.9e-014	1.4e00
64	17	11	112	5.6e-014	3.7e-014	2.6e00
128	17	12	112	1.5e-013	5.7e-014	3.1e00
256	18	12	122	1.9e-013	6.4e-014	5.9e00
512	18	12	123	2.5e-012	2.1e-012	9.8e00
1024	19	13	130	2.7e-013	1.4e-013	2.6e01
2048	19	13	133	9.0e-012	7.1e-012	5.4e01
4096	20	13	141	5.7e-013	2.8e-013	2.7e02
8192	20	14	149	1.9e-012	3.3e-012	5.8e02
16384	21	15	154	3.7e-013	2.2e-013	1.4e03
32768	21	15	158	3.4e-012	3.0e-012	3.8e03

Table 1: Results for the modified Newton's iteration I.

n	$Nstep$	$Mdrk$	$Sdrk$	$res(X)$	$\|A^\dagger - X\|$	time (seconds)
32	20	7	99	2.5e-012	4.2e-012	1.4e00
64	20	7	102	3.1e-010	2.1e-009	2.2e00
128	22	7	114	1.2e-011	3.0e-012	3.0e00
256	22	7	110	8.5e-012	1.1e-011	4.0e00
512	22	8	111	6.5e-010	2.3e-009	4.8e00
1024	24	8	122	2.0e-011	7.2e-012	1.5e01
2048	24	9	126	7.1e-012	3.3e-012	2.9e01
4096	24	9	128	2.7e-011	3.4e-011	1.5e02
8192	24	8	126	3.2e-010	1.1e-009	2.8e02
16384	26	9	140	1.9e-010	9.7e-011	7.1e02
32768	26	9	142	1.6e-010	8.4e-011	1.8e03

Table 2: Results for the modified Newton's iteration II.

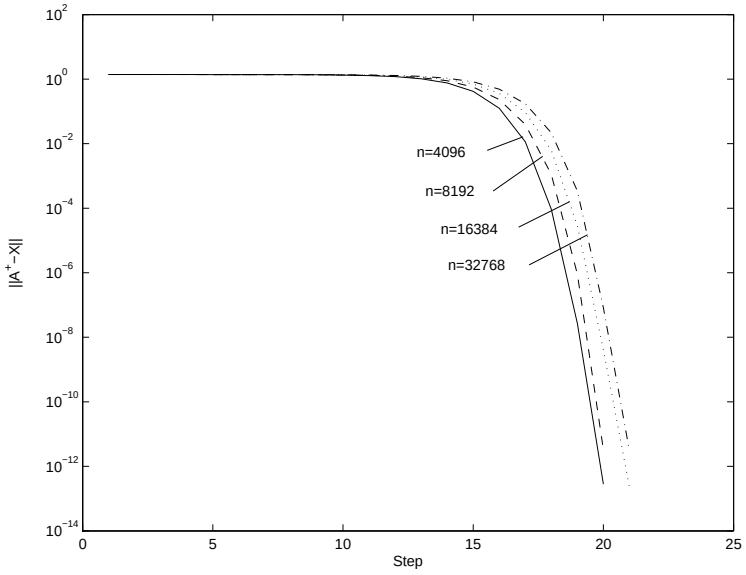

Figure 4: $\|A^{\dagger} - X_i\|$ of the modified Newton's iterations I.

[3] D. Bini and V. Pan, Matrix and Polynomial Computations, Vol 1: Fundamental Algorithms, Birkauser, Boston, 1994.

[4] D. Bini and B. Meini, Approximate displacement rank and applications, in Structured Matrices in Mathematics,Computer Science and Engineering II, V. Olshevsky Editor, Contemporary Mathematics 281, pp. 215-232, American Mathematical Society, Rhode Island, 2001.

[5] V. Pan and R. Schreiber, An improved Newton iteration for generalized inverse of a matrix with applications, SIAM J. Sci. Stat. Comput., 12 (1991) 1109-1131.

[6] V. Pan, Y. Rami and X. Wang, Structured matrices and Newton's iteration: Unified approach, Linear Algebra Appl., 343-344(2002) 233-265.

[7] Y. Wei, Recurrent neural networks for computing weighted Moore-Penrose inverse, Appl. Math. Comput., 116 (2000) 279-287.

[8] Y. Wei, H. Wu and J. Wei, Successive matrix squaring algorithm for parallel computing the weighted generalized inverse A^{+}_{MN}, Appl. Math. Comput., 116 (2000) 289-296.

[9] Y. Wei and M. Ng, Weighted Tikhonov filter matrices for ill-posed problem, Applied Mathematics and Computation, to appear.

[10] A. Ben-Israel and T. Greville, Generalized Inverses: Theory and Applications, John Wiley, New York, 1974; 2nd Edition, Springer-Verlag, New York, 2003.

n	time (seconds)
32	1.0e-02
64	5.0e-02
128	2.7e-01
256	2.7e00
512	1.9e01
1024	1.5e02
2048	1.3e03
4096	>2.0e04
8192	out of memory

Table 3: Computational times required by the original Newton's iteration

[11] G. Heinig and K. Rost, Algebraic Methods for Toeplitz-like Matrices and Operators, Akademie-Verlag, Berlin, 1984.

[12] G. Golub and C. Van Loan, Matrix Computations, The John Hopkins University Press, Third Edition, 1996.