# On the regularized solution of block banded block Toeplitz systems

D. A. Bini[a], A. Farusi[b], G. Fiorentino[a] and B. Meini[a]

[a]Dipartimento di Matematica, via Buonarroti 2, 56127 Pisa, Italy
[b]Istituto di Elaborazione dell'Informazione, CNR, S. Cataldo, Pisa, Italy

## ABSTRACT

Given two $n \times n$ Toeplitz matrices $T_1$ and $T_2$, and a vector $\boldsymbol{b} \in \boldsymbol{R}^{n^2}$, consider the linear system $A\boldsymbol{x} = \boldsymbol{b} - \boldsymbol{\eta}$, where $\boldsymbol{\eta} \in \boldsymbol{R}^{n^2}$ is an unknown vector representing the noise and $A = T_1 \otimes T_2$. Recovering approximations of $\boldsymbol{x}$, given $A$ and $\boldsymbol{b}$, is encountered in image restoration problems. We propose a method for the approximation of the solution $\boldsymbol{x}$ that has good regularization properties. The algorithm is based on a modified version of Newton's iteration for matrix inversion and relies on the concept of approximate displacement rank. We provide a formal description of the regularization properties of Newton's iteration in terms of filters and determine bounds to the number of iterations that guarantee regularization. The method is extended to deal with more general systems where $A = \sum_{i=1}^{h} T_1^{(i)} \otimes T_2^{(i)}$. The cost of computing regularized inverses is $O(n \log n)$ operations (ops), the cost of solving the system $A\boldsymbol{x} = \boldsymbol{b}$ is $O(n^2 \log n)$ ops. Numerical experiments which show the effectiveness of our algorithm are presented.

**Keywords:** Band Toeplitz systems, Newton's iteration, regularization, displacement rank, image processing.

## 1. INTRODUCTION

Solving linear systems with noise is commonly encountered in image restoration problems. Typically, an image is represented by an $m \times n$ array $X = (x_{i,j})$ where the entry $x_{i,j}$ provides the value of the pixel with coordinate $(i, j)$ of the image. Using a vector notation, it is more convenient to represent $X$ as a vector $\boldsymbol{x} = vec(X)$ obtained by arranging columnwise the entries of $X$. In this way, the typical problem of image restoration is to solve the linear system

$$A\boldsymbol{x} = \boldsymbol{b} - \boldsymbol{\eta} \tag{1}$$

where $\boldsymbol{x}$ is the original image, $A$ is the blur operator so that $A\boldsymbol{x}$ is the blurred image, and $\boldsymbol{\eta}$ is the additive noise of which we have a partial information, like some statistical properties or its norm. The vector $\boldsymbol{b} = vec(B)$ represents the blurred image $B$ with the additive noise so that, solving the linear system

$$A\boldsymbol{y} = \boldsymbol{b} \tag{2}$$

provides the vector $\boldsymbol{y} = \boldsymbol{x} + A^{-1}\boldsymbol{\eta}$ which may differ very much from $\boldsymbol{x}$ due to the noise and to the very large ill conditioning of $A$, that is typical for blur operators.

From the space invariance of the point spread function (PSF) that defines the matrix $A$ and from its local action (a single source of light is blurred into a small spot wherever the source is located), the matrix $A$ inherits the 2-level Toeplitz structure and the 2-level band structure. More precisely, if $M = (m_{i,j})_{i,j=-k,k}$ denotes the mask representing the PSF, i.e., such that a unit pixel of coordinates $(r, s)$ is spread onto the pixels $m_{r+i,s+j}$, $i, j = -k, k$, then the matrix $A$ is a block Toeplitz matrix with a $(2k+1)$-diagonal block banded structure where the general block $A_i$ on the $i$-th block diagonal, for $i = -k, \ldots, k$, is a $(2k+1)$-diagonal banded Toeplitz matrix with entry $m_{i,j}$ on the $j$-th diagonal for $j = -k, \ldots, k$.

In certain cases the matrix $A$ can be written as $T_1 \otimes T_2$, where $\otimes$ denotes the Kronecker product. We refer to this situation as to the case of separable PSF. An important instance of separable PSF occurs if the entries of the mask are defined by means of a Gaussian function, say, $m_{i,j} = \beta \exp(-\alpha(i^2 + j^2))$ for given $\alpha, \beta > 0$. In fact, $T_1 = T_2 = (t_{j-i})$, $t_h = \sqrt{\beta} \exp(-\alpha h^2)$, $h = -k, \ldots, k$, $t_h = 0$ elsewhere.

bini@dm.unipi.it, farusi@iei.pi.cnr.it, fiorent@dm.unipi.it, meini@dm.unipi.it

A fundamental problem in this context is to design algorithms for the solution of (2) providing a good approximation of the true image $\boldsymbol{x}$, solution of (1), at a low computational cost. While the latter feature relies on exploiting the specific structures of the matrix $A$, the former is usually based on suitable regularization techniques (like Tikhonoff regularization), or on regularizing features of the conjugate gradient iterations (see Ref. 1).

Here, we introduce a new method for the regularized solution of the system (2) based on Newton's iteration for matrix inversion and on the concept of approximate displacement rank of Ref. 2,3.

The new algorithm is particularly effective in the case of a separable PSF but can be used as a tool for speeding up the convergence, still keeping the regularizing properties, in the general case of nonseparable PSFs. The regularizing features are provided by the use of Newton's iteration that is a well known quadratically convergent method for the approximation of the inverse matrix. In fact, the convergence of Newton's iteration is much faster in the subspaces which do not contain the undesired noise components and is slow along the remaining components. This feature can be fine tuned and even amplified with the choice of a suitable initial approximation. We denote this choice as *preconditioning the Newton iteration* since its effect is somehow similar to the one of the preconditioned conjugate gradient.

The regularization property of Newton's iteration seems to have been neglected in the current and past literature. Probably the reason is that a single iteration step is so expensive that the method is actually unpractical. In fact for an $n \times n$ matrix the cost of one step of Newton's iteration amounts to $O(n^3)$ arithmetic operations (ops). However, using the tools introduced in Ref. 2,3 and in particular the concept of approximate displacement rank, we are able to dramatically reduce the cost of a single step applied to a Toeplitz matrix to $O(n \log n)$ ops. This makes Newton's iteration an effective tool for the design of regularizing algorithms.

Roughly speaking, our technique provides a displacement representation (see Ref. 4) of "regularized" inverses $T_1^+$ and $T_2^+$ of $T_1$ and $T_2$, respectively. In this way, if we denote $X$ and $B$ the images corresponding to $\boldsymbol{x}$ and $\boldsymbol{b}$ (i.e., such that $vec(X) = \boldsymbol{x}$ and $vec(B) = \boldsymbol{b}$, respectively), then a regularized approximation $\widehat{X}$ of $X$, solution of (1), is obtained as $\widehat{X} = T_2^+ B (T_1^+)^T$. Moreover, the displacement structures of $T_1^+$ and $T_2^+$ allow one to compute the products $T_2^+ B (T_1^+)^T$ with only $O(n^2 \log n)$ ops, where for simplicity here and hereafter we assume $m = n$. The computation of $T_1^+$ and $T_2^+$ is performed in just $O(n \log n)$ ops.

For nonseparable problems, we use the same technique inside an iterative procedure for the successive approximation of the solution, based on the approach of Ref. 5,6 which relies on the singular values decomposition (SVD) of the mask $M$. More precisely, from the SVD $M = \sum_{j=1}^{2k+1} \sigma_j \boldsymbol{u}_j \boldsymbol{v}_j^T$, where $\sigma_1 \geq \ldots \geq \sigma_{2k+1} \geq 0$ are the singular values and $\boldsymbol{u}_j = (u_{j,h})_{h=-k,k}$, $\boldsymbol{v}_j = (v_{j,h})_{h=-k,k}$ are the left and right singular vectors, respectively, we obtain the splitting $A = T_1^{(1)} \otimes T_2^{(1)} + \sum_{j=2}^{2k+1} T_1^{(j)} \otimes T_2^{(j)}$, where $T_1^{(j)} = \sqrt{\sigma_j} U_j$, $T_2^{(j)} = \sqrt{\sigma_j} V_j$, $U_j$ and $V_j$ are the $(2k+1)$-diagonal Toeplitz matrices defined by the vectors $\boldsymbol{u}_j$ and $\boldsymbol{v}_j$, respectively, i.e., the entries on the $i$-th diagonal of $U_j$ and $V_j$ are $u_{j,i}$ and $v_{j,i}$, respectively, for $i = -k, \ldots, k$. For the PSFs encountered in practical problems the most part of information is contained in $\sigma_1 \boldsymbol{u}_1 \boldsymbol{v}_1^T$ (see Ref. 5,6) whereas the singular values $\sigma_i$, for $i > 1$, are relatively negligible. This means that $T_1^{(1)} \otimes T_2^{(1)}$ is already a good approximation of $A$ and can be used as a preconditioner in PCG iterations or in Jacobi-like iterations. Concerning the latter, the solution of the system is approximated with the iteration scheme

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \left( {T_1^{(1)}}^+ \otimes {T_2^{(1)}}^+ \right) (\boldsymbol{b} - A\boldsymbol{x}_i), \quad i \geq 0. \tag{3}$$

The paper is organized as follows. In section 2 we introduce the tools needed for the design of our algorithm including Newton's iterations, regularization properties expressed in terms of filters, displacement rank and $\epsilon$-displacement rank, orthogonal displacement representations. In section 3 we describe our modified version of Newton's iteration and apply it to the separable and to the nonseparable case. Finally in section 4 we describe and comment our numerical experiments.

## 2. THE TOOLS

### 2.1. Newton's Iteration

Newton's iteration is a valid tool for solving problems whose solution can be formally viewed as the zero of a known function. This method has been analyzed in sections 1.6, 2.11, 3.5 of Ref. 7 and, concerning Toeplitz matrices has been investigated in Ref. 8,2,3.

Let $A$ be a nonsingular matrix and consider the sequence of matrices generated, from an assigned $X_0$, in the following way

$$X_{i+1} = 2X_i - X_i A X_i. \tag{4}$$

This sequence is formally obtained by applying Newton's iteration $x_{i+1} = x_i - f(x_i)/f'(x_i)$ to the function $f(X) = X^{-1} - A$.

Denoting $R_i = I - X_i A$ the residual error at the $i$-th step, from (4) we obtain that $R_{i+1} = R_i^2$, that is, $R_i = R_0^{2^i}$. The above relation provides information on the convergence of the general iteration (4).

If $A \in \boldsymbol{R}^{n\times n}$ is symmetric and positive definite with eigenvalues $0 < \lambda_n \le \lambda_{n-1} \le \ldots \le \lambda_1 = \rho(A)$, then the choice $X_0 = \theta I$, with $\theta \le 1/\rho(A)$, say, $\theta = 1/||A||$ where $||\cdot||$ is any operator norm, provides a sequence $X_i$ converging to $A^{-1}$. In fact, the matrix $R_0 = I - \theta A$ is symmetric, nonnegative definite with eigenvalues $0 \le 1-\theta\lambda_1 \le \ldots \le 1-\theta\lambda_n < 1$, therefore $\rho(R_0) = 1 - \theta\lambda_n \ge 1 - 1/\mathrm{cond}_2(A)$, $\mathrm{cond}_2(A) = ||A||_2 ||A^{-1}||_2$ and $\rho(R_i) = (1-\theta\lambda_n)^{2^i}$. In this way, since $(1-1/x)^x < 1/e < 1/2$, after $i = \lceil \log_2(1/(\theta\lambda_n)) \rceil$ iterations, we have $\rho(R_i) \le 1/e < 1/2$, therefore, in order to have the residual error less than a given error bound $\epsilon$, it is sufficient to apply the iteration (4) for

$$\nu = \lceil \log_2(1/(\theta\lambda_n)) \rceil + \lceil \log_2 \log \epsilon^{-1} \rceil \ge \lceil \log_2 \mathrm{cond}_2(A) \rceil + \lceil \log_2 \log \epsilon^{-1} \rceil$$

steps.

The first additive term in the above expression, that in Ref. 2 is called *overhead cost* or *overhead number of iterations*, provides the number of steps needed in order to arrive at a *pure double exponential convergence* of order 2, where for pure double exponential convergence of order 2 we mean that the error tends to zero as $a^{2^i}$ for $a \le 1/2$. The second term provides the number of steps of pure double exponential convergence that is needed to reach the residual error bound $\epsilon$. A better choice of $\theta$ that still guarantees convergence is $\theta = 2/||A||$.

For a generic matrix $A$, the starting approximation $X_0 = \theta A^T$ with $\theta \ge ||A^T A||$, guarantees the convergence. In fact, $R_0 = I - \theta A^T A$ is such that $\rho(R_0) < 1$. Moreover, if $A = U\Sigma V^T = \sum_{j=1}^n \sigma_j \boldsymbol{u}_j \boldsymbol{v}_j^T$ is the SVD of $A$, where $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$, $\sigma_1 \ge \ldots \ge \sigma_n \ge 0$, we have $\rho(R_0) = 1 - \theta\sigma_n^2 \ge 1 - 1/\mathrm{cond}_2(A)^2$. Thus, the overhead cost becomes $2\lceil \log_2 \mathrm{cond}_2(A) \rceil$. That is, with respect to the positive definite case with the choice $X_0 = \theta I$, the overhead cost is doubled. Also for a general matrix $A$ we may choose $\theta = 2/||A^T A||$ and convergence is guaranteed.

## 2.2. Regularizing Properties of Newton's Iteration

Let $A\boldsymbol{x} = \boldsymbol{b} - \boldsymbol{\eta}$ be the system where the noise $\boldsymbol{\eta}$ has been removed from the known term vector $\boldsymbol{b}$, and let $A\boldsymbol{y} = \boldsymbol{b}$ the system with the noise. Indeed, $\boldsymbol{y} = \boldsymbol{x} + A^{-1}\boldsymbol{\eta}$. From the SVD of $A$, $A = U\Sigma V^T$, we easily deduce that

$$\boldsymbol{y} = \boldsymbol{x} + \sum_{j=1}^n \frac{\boldsymbol{u}_j^T \boldsymbol{\eta}}{\sigma_j} \boldsymbol{v}_j, \tag{5}$$

that is, the components of the error $\boldsymbol{\eta}$ along the vectors $\boldsymbol{v}_i$ are strongly amplified whenever $\sigma_j \approx 0$. This amplification of the noise spoils the solution in its components along the correspondent vectors $\boldsymbol{v}_j$.

Consider the approximation $\boldsymbol{y}_i = X_i \boldsymbol{b}$, where $X_i$ is the $i$-th approximation to $A^{-1}$ provided by Newton's iteration with $X_0 = \theta A^T$. Then we have

$$\boldsymbol{y}_i = \sum_{j=1}^n \phi_j^{(i)} \frac{\boldsymbol{u}_j^T \boldsymbol{b}}{\sigma_j} \boldsymbol{v}_j \tag{6}$$

where the "filters" $\phi_j^{(i)}$ are such that

$$\phi_j^{(i)} = 1 - (1 - \theta\sigma_j^2)^{2^i}. \tag{7}$$

Moreover,

$$\boldsymbol{y}_i = \boldsymbol{x} - R_i \boldsymbol{x} + (I - R_i) A^{-1} \boldsymbol{\eta} = \boldsymbol{x} - \boldsymbol{\epsilon}_i + \boldsymbol{\delta}_i, \tag{8}$$

where $\boldsymbol{\epsilon}_i$ is the approximation error of Newton's iteration and $\boldsymbol{\delta}_i$ is the contribution of the noise $\boldsymbol{\eta}$.

We easily find that

$$\boldsymbol{\epsilon}_i = R_i \boldsymbol{x} = \sum_{j=1}^n (1 - \theta\sigma_j^2)^{2^i} (\boldsymbol{v}_j^T \boldsymbol{x}) \boldsymbol{v}_j, \quad \boldsymbol{\delta}_i = \sum_{j=1}^n \phi_j^{(i)} \frac{\boldsymbol{u}_j^T \boldsymbol{\eta}}{\sigma_j} \boldsymbol{v}_j. \tag{9}$$

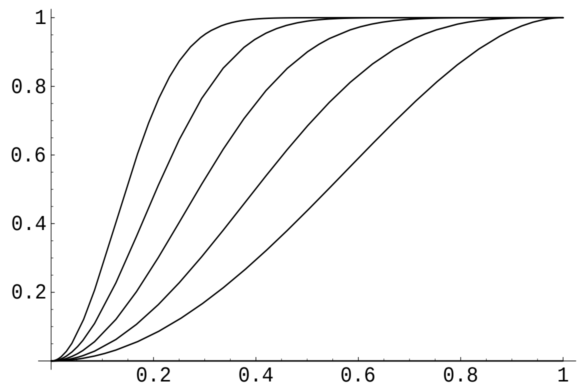



**Figure 1.** Functions $\phi^{(i)}(t) = 1 - (1-t^2)^{2^i}$ for $i = 1, \ldots, 5$ over $[0, 1]$

In particular we point out that, for $i \to +\infty$, the convergence to zero of the approximation error $\boldsymbol{\epsilon}_i$ is faster for the components along $\boldsymbol{v}_j$ such that $\sigma_j \approx 0$ than for the ones corresponding to larger $\sigma_j$'s. In other words, the smaller is $\sigma_j$, the larger is the overhead cost along the $j$-th component, thus the pure double exponential convergence starts earlier for the larger $\sigma_j$'s.

Concerning the noise, we point out that the amplification along the component $\boldsymbol{v}_j$ given by the factor $\frac{1}{\sigma_j}$ in (5), is damped in (9) by the "filter" $\phi_j^{(i)}$ that is naturally introduced by Newton's iteration. In order to better appreciate the role of this filter, let us observe that for the Euclidean norm $||\cdot||_2$ it holds $||A^T A||_2 = \sigma_1^2$, thus for $\theta = \frac{1}{||A^T A||_2}$ we have $\theta\sigma_j^2 = \left(\frac{\sigma_j}{\sigma_1}\right)^2 < 1$, moreover, if $\theta = \frac{2}{||A^T A||_2}$ then $\theta\sigma_j^2 = 2\left(\frac{\sigma_j}{\sigma_1}\right)^2$. Therefore we may plot the continuous counterpart $\phi^{(i)}(t) = 1 - (1-t^2)^{2^i}$ of $\phi_j^{(i)}$ obtained by replacing $\sqrt{\theta}\sigma_j$ with the continuous variable $t$, where $t$ ranges in $[0, 1]$, or in $[0, \sqrt{2}]$ if $\theta = 2/\sigma_1^2$. From figure 1 we may see that the components of the noise corresponding to the values of $t$ close to zero (i.e., the most dangerous ones) are strongly filtered since $\phi^{(i)}(t)$ is close to zero, whereas the remaining ones are almost left unchanged.

We may estimate the threshold value under which the filter acts. We find that for $t < \hat{t}(i) = \sqrt{1 - \frac{1}{2^{2^i - 1}}}$ the filter takes values less than $1/2$. We may choose the number of steps of Newton's iteration according to this threshold. Another possible criterion for determining the number of steps is based on the following observation. From (9) it follows that the noise is amplified by the factor $\psi^{(i)}(t) = \phi^{(i)}(t)/t$, moreover, if $i < \log_2(1/\tilde{t})$ then we have $\psi^{(i)}(t) < 1$ for $0 \le t \le \tilde{t}$. That is, after $\nu$ steps of Newton's iteration, the noise is not amplified along the components for which $\sigma_i < 1/2^\nu$. Once we have chosen the cut-off value $\tilde{t}$, based on suitable information about the noise, we may determine the number of steps of Newton's iteration which guarantee that the noise is not amplified under the cut-off level.

It is interesting to observe that the expressions (6), (7), (8) and (9) coincide with the analogous of the Landweber iteration (see Ref. 1, page 137) with the difference that the exponent in the filters $\phi_j^{(i)}$ in the Landweber iteration is just $i$, whereas in the Newton iteration is $2^i$. This reflects the quadratic convergence of Newton's iteration versus the linear convergence of Landweber's iteration. Similar regularization properties are obtained with $X_0 = \theta I$.

It is possible to improve the convergence of the Newton iteration still keeping the regularization properties, by choosing a different initial approximation $X_0$. Let us consider the case where $A$ is positive definite and observe that if $C$ is positive definite then $C^{1/2}$ is well defined and the matrix $CA$ is similar to the symmetric positive definite matrix $C^{1/2}AC^{1/2}$. Therefore, by choosing $X_0 = \theta C$ with $\theta \le 1/||CA||$, the residual $R_0 = I - X_0 A = I - \theta CA$ has eigenvalues $1 - \theta\lambda_j$ where $0 \le \lambda_1 \le \ldots \le \lambda_n$ are the eigenvalues of $CA$, and the same arguments used in the analysis of the case $X_0 = \theta I$ apply. In fact, it holds

$$R_i = \sum_{j=1}^{n} (1 - \theta\lambda_j)^{2^i} \boldsymbol{u}_j \boldsymbol{v}_j^T, \tag{10}$$

where $\boldsymbol{u}_j$, $\boldsymbol{v}_j$ are the right and left eigenvectors of $CA$, respectively.

Similarly, for a general matrix $A$ we may choose $X_0 = \theta A^T C^T C$, for any matrix $C$, $\theta \leq 1/||CA||^2$. In this case we have $R_0 = I - \theta(CA)^T(CA)$ and $\rho(R_0) < 1$. Alternative choices for $X_0$ are $X_0 = \theta CC^T A^T$ or $X_0 = \frac{\theta}{2}(CC^T A^T + A^T C^T C)$. A similar analysis can be carried out and the overhead cost now becomes $2 \log \mathrm{cond}_2(CA)$. Thus the convergence can be improved if $\mathrm{cond}_2(CA) << \mathrm{cond}_2(A)$. The analogy with the preconditioned conjugate gradient technique where the rate of (linear) convergence depends on $\mathrm{cond}_2(CA)$ (for the preconditioned version) and on $\mathrm{cond}_2(A)$ (for the unpreconditioned version), leads us to define the matrix $C$ as the *preconditioner* and to refer to Newton's iteration with $X_0$ depending on $C$ as to *preconditioned Newton's iteration*.

The wide literature on the design of preconditioners (see, for instance Ref. 9) provides us many techniques for the construction of a suitable $C$. However, the choice of $C$ is rather delicate since it may strongly accelerate the convergence but, at the same time, it may destroy the regularization features of Newton's iteration. In particular, this holds for the circulant preconditioner of R. Chan[9] unless its inversion is replaced by some sort of generalized inversion.

For a general $n \times n$ matrix $A$, Newton's iteration has the cost of $O(n^3)$ ops per step. This is the main drawback that makes this iteration unpracticable for the effective inversion of a matrix. However, for special classes of matrices, say, for Toeplitz matrices, the cost can be reduced to $O(n \log n)$ ops per step. This can be accomplished by using the concept of approximate displacement rank.

### 2.3. Approximate Displacement Rank

The concept of displacement operators and displacement rank, introduced in Ref. 4,10, and successively elaborated by other authors (see Ref. 11 and section 2.11 of Ref. 7), is a powerful tool for dealing with Toeplitz matrices. In certain situations this tool seems to be inadequate to cope with the specific computational issues encountered in practice and a modified version of it with better "numerical" features seems best suited for several applications. Here we recall the main results concerning displacement rank, *orthogonal displacement representation* and *approximate displacement rank* and prepare the tools for the effective implementation of Newton's iteration. Throughout this section we will follow Ref. 2,3,12.

Let us recall the operator

$$\Delta(A) = AZ - ZA, \quad Z = (z_{i,j}), \quad z_{i+1,i} = 1, \quad z_{i,j} = 0, \text{ otherwise}, \tag{11}$$

applied to $n \times n$ matrices $A$, and define the *displacement rank* of $A$ as $\mathrm{drk}(A) = \mathrm{rank}(\Delta(A))$. A Toeplitz matrix has displacement rank at most 2, moreover, $\mathrm{drk}(A)$ is a sort of measure of the distance of $A$ from the algebra of lower triangular Toeplitz matrices, for which the displacement rank is zero.

Define $L(\boldsymbol{a}) = \sum_{i=0}^{n-1} a_{i+1} Z^i$ the $n \times n$ lower triangular Toeplitz matrix having $\boldsymbol{a} = (a_i)_{i=1,n}$ as first column. It holds (see section 2.11 of Ref. 7)

$$\Delta(A) = \sum_{i=1}^{r} \boldsymbol{u}_i \boldsymbol{v}_i^T \quad \Rightarrow A = L(A\boldsymbol{e}_1) + \sum_{i=1}^{r} L(\boldsymbol{u}_i) L^T(Z\boldsymbol{v}_i), \tag{12}$$

where $\boldsymbol{u}_i$, $\boldsymbol{v}_i$, $i = 1, \ldots, r$, are $n$-dimensional vectors, $\boldsymbol{e}_1$ denotes the first column of the identity matrix. The expression in the right-hand side of (12) is called the *displacement representation of* $A$ and the vectors $\boldsymbol{a} = A\boldsymbol{e}_1$, $\boldsymbol{u}_i$, $\boldsymbol{v}_i$, $i = 1, \ldots, r$ are called a *set of generators for the displacement representation of* $A$. A matrix $A$ with displacement rank $r$ is called Toeplitz-like if $r << n$.

A simple consequence of the relation

$$\Delta(A^{-1}) = -A^{-1} \Delta(A) A^{-1}, \tag{13}$$

that can be directly derived from (11), is that $\mathrm{drk}(A^{-1}) = \mathrm{drk}(A)$ and the following displacement representation of $A^{-1}$ holds

$$A^{-1} = L(A^{-1}\boldsymbol{e}_1) - \sum_{i=1}^{r} L(A^{-1}\boldsymbol{u}_i) L^T(ZA^{-T}\boldsymbol{v}_i). \tag{14}$$

Since the product of an $n \times n$ Toeplitz matrix and a vector can be performed in $O(n \log n)$ ops by means of FFT, the representations (12), (14) allow us to make elementary operations among Toeplitz-like matrices at a low cost. In particular, computing the product of a matrix with displacement rank $r$ and a vector costs just $O(rn \log n)$ ops.

The set of generators for the displacement representation of $A$ is not unique. Indeed, observe that, if $\Delta(A) = UV^T$, for $U, V \in \boldsymbol{R}^{n\times r}$ then $\Delta(A) = \hat{U}\hat{V}^T$ for $\hat{U} = US$, $\hat{V} = VS^{-T}$, where $S \in \boldsymbol{R}^{r\times r}$ is any nonsingular matrix. Therefore another displacement representation can be derived by the latter decomposition. If the product of the norms of the columns of $U$ and the corresponding columns of $V$ is much larger than the norm of $A$, then computing the matrix vector product $A\boldsymbol{x}$ by means of the displacement representation formula (12) may lead to large numerical cancellation errors. This drawback can be overcome by using the concept of *orthogonal displacement representation.*[2,3]

Definition 2.1. If $\Delta(A) = U\Sigma V^T$, $\Sigma = \mathrm{diag}(\boldsymbol{\sigma})$, $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_r)$, $U$, $V \in \boldsymbol{R}^{n\times r}$, is the singular values decomposition of $\Delta(A)$, then the quadruple $(\boldsymbol{a}, U, \boldsymbol{\sigma}, V)$ defines a set of generators, such that

$$A = L(\boldsymbol{a}) + \sum_{i=1}^{r} \sigma_i L(\boldsymbol{u}_i) L^T(Z\boldsymbol{v}_i), \tag{15}$$

where $\boldsymbol{u}_i = U\boldsymbol{e}_i$, $\boldsymbol{v}_i = V\boldsymbol{e}_i$, $i = 1, \ldots, r$, and $\boldsymbol{e}_i$ denotes the $i$-th column of the $n \times n$ identity matrix. We refer to this particular set of generators as to an *orthogonal displacement generator* for the displacement representation of $A$, and we set $\mathrm{ODG}(A) = (\boldsymbol{a}, U, \boldsymbol{\sigma}, V)$. The matrix representation (15) is called *orthogonal displacement representation* of $A$ and we write $A = \mathrm{ODR}(\boldsymbol{a}, U, \boldsymbol{\sigma}, V)$.

Observe that for the Euclidean norm $||\cdot||$ we have $||\Delta(A)|| = ||AZ - ZA|| \le 2||A||$, whence for an orthogonal set of generators we have $||U|| = ||V|| = 1$, $\sigma_1 = ||\Delta(A)|| \le 2||A||$. This fact overcomes the possibility of numerical cancellation.

A matrix $A$ which is symmetric with respect to the secondary diagonal, i.e., $a_{i,j} = a_{n-j+1,n-i+1}$, is called *persymmetric*. A persymmetric matrix $A$ is such that $A^T = JAJ$, where $J$ is the permutation matrix having 1 in the anti-diagonal. An interesting example of persymmetric matrices is the class of Toeplitz matrices. It can be easily proved that $\Delta(A)J$ is skew-symmetric if $A$ is persymmetric. The following result holds:

Theorem 2.2. [Displacement representation of a persymmetric matrix]. *If the persymmetric matrix $A$ has displacement rank $r$, then $r = 2h$ and there exist matrices $W \in \boldsymbol{R}^{n\times r}$, $D = \mathrm{diag}(d_1, \ldots, d_h) \in \boldsymbol{R}^{h\times h}$ such that $W^TW = I_r$, $|d_1| \ge |d_2| \ge \ldots \ge |d_h|$, $|d_j| = \sigma_{2j-1} = \sigma_{2j}$, $j = 1, \ldots, h$, where $\sigma_1, \ldots, \sigma_r$ are the singular values of $\Delta(A)$, and*

$$\Delta(A) = W(D \otimes S)W^TJ, \quad S = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

*Therefore,*

$$A = L(\boldsymbol{a}) + \sum_{i=1}^{h} d_i(L(\boldsymbol{w}_{2i-1})L^T(ZJ\boldsymbol{w}_{2i}) - L(\boldsymbol{w}_{2i})L^T(ZJ\boldsymbol{w}_{2i-1})) \tag{16}$$

*where $\boldsymbol{a} = A\boldsymbol{e}_1$, $\boldsymbol{w}_i = W\boldsymbol{e}_i$, $i = 1, \ldots, r$. An orthogonal generator for $A$ is completely defined by the triple $(\boldsymbol{a}, W, \boldsymbol{d})$ where $\boldsymbol{d} = (d_i)_{i,1,\ldots,h}$.*

This fact can be exploited for reducing the computational cost of multiplying the matrix $A$ and a vector. Indeed, it is easy to prove that, given the FFTs associated with $\boldsymbol{x}$ and $\boldsymbol{y}$, only the FFTs associated with $\boldsymbol{w}$, $\boldsymbol{x}$ and $\boldsymbol{y}$ are sufficient for the computation of the products $L(\boldsymbol{w})\boldsymbol{x}$ and $L(ZJ\boldsymbol{w})\boldsymbol{y}$.

Definition 2.3. If $A$ is persymmetric and $\Delta(A) = W(D \otimes S)W^TJ$, we define the triple $(\boldsymbol{a}, W, \boldsymbol{d})$, where $\boldsymbol{a} = A\boldsymbol{e}_1$, $\boldsymbol{d} = (d_i)_{i=1,h}$ a *persymmetric orthogonal displacement generator* of $A$ and we use the same notation $\mathrm{ODG}(A) = (\boldsymbol{a}, W, \boldsymbol{d})$. Similarly, the representation

$$A = L(\boldsymbol{a}) + \sum_{i=1}^{h} d_i[L(\boldsymbol{w}_{2i-1})L^T(ZJ\boldsymbol{w}_{2i}) - L(\boldsymbol{w}_{2i})L^T(ZJ\boldsymbol{w}_{2i-1})]$$

is called *persymmetric orthogonal displacement representation* of $A$ and we use the same notation $A = \mathrm{ODR}(\boldsymbol{a}, W, \boldsymbol{d})$.

The following algorithms provide a detailed description of the computation of the ODG of a matrix.

Algorithm 2.4. [*Computing an ODG of a persymmetric matrix*].

Input: Matrices $U \in \boldsymbol{R}^{n\times k}$, $M \in \boldsymbol{R}^{k\times k}$ such that $\Delta(A) = FMF^TJ$, $M = -M^T$, where $A$ is persymmetric;

Output: $\mathrm{ODG}(A) = (\boldsymbol{a}, W, \boldsymbol{d})$.

Computation:

– Compute the QR factorization $F = QR$.
– Compute $H = RMR^T$.
– Compute the SVD $H = U\Sigma V$.
– Compute the sign $\gamma_j = \text{sign}(u_{i,2j-1}, v_{i,2j})$, $j = 1, \ldots, h$, where $i$ is such that $u_{i,2j-1}v_{i,2j} \neq 0$.
– Set $W = QU$, $d_i = \gamma_i \sigma_{2i-1}$, $i = 1, \ldots, h$, $\boldsymbol{d} = (d_i)$, $\boldsymbol{a} = A\boldsymbol{e}_1$.

Now we recall the concept of approximate displacement rank, or $\epsilon$-displacement rank, introduced and analyzed in Ref. 2,3.

DEFINITION 2.5. For a given $\epsilon > 0$ and for the Euclidean norm $||\cdot||$, define the *$\epsilon$-displacement rank* of a matrix $A$ as $\text{drk}_\epsilon(A) = \min_{||E|| \le \epsilon} \text{rank}(\Delta(A) + E)$, and the *relative $\epsilon$-displacement rank* as $\text{rdrk}_\epsilon(A) = \min_{||E|| \le \epsilon ||A||} \text{rank}(\Delta(A) + E)$. If $r = \text{drk}_\epsilon(A)$ ($r = \text{rdrk}_\epsilon(A)$), and if $\Delta(A) = \sum_{i=1}^r \boldsymbol{u}_i \boldsymbol{v}_i^T + E$, with $||E|| \le \epsilon$ ($||E|| \le \epsilon ||A||$), we say that the matrix

$$L(A\boldsymbol{e}_1) + \sum_{i=1}^r L(\boldsymbol{u}_i) L^T(Z\boldsymbol{v}_i)$$

is an *$\epsilon$-displacement (relative $\epsilon$-displacement) representation* of $A$.

The following properties hold (see Ref. 3,12).

THEOREM 2.6. *Let $\Delta(A) = U\Sigma V^H$ be the singular value decomposition of $\Delta(A)$, where $\sigma_1 \ge \ldots \ge \sigma_k > 0$ are the singular values of $\Delta(A)$. Let $\sigma_1 > \epsilon \ge \sigma_k$; then $\text{drk}_\epsilon(A) = r$ if and only if $\sigma_r > \epsilon \ge \sigma_{r+1}$. Similarly, let $\sigma_1 > \epsilon ||A|| \ge \sigma_k$; then $\text{rdrk}_\epsilon(A) = r$ if and only if $\sigma_r > \epsilon ||A|| \ge \sigma_{r+1}$.*

In the hypotheses of Theorem 2.6 we call the quadruple $(\boldsymbol{a}, U_\epsilon, \boldsymbol{\sigma}_\epsilon, V_\epsilon)$ *orthogonal (relative) $\epsilon$-displacement generator* of $A$ if $\boldsymbol{\sigma}_\epsilon = (\sigma_1, \ldots, \sigma_r)$, $U_\epsilon$ and $V_\epsilon$ are made up by the first $r$ columns of $U$ and $V$, respectively, and $\boldsymbol{a} = A\boldsymbol{e}_1$. Moreover, the representation

$$L(\boldsymbol{a}) + \sum_{i=1}^r \sigma_i L(\boldsymbol{u}_i) L^T(Z\boldsymbol{v}_i)$$

is called *orthogonal (relative) $\epsilon$-displacement representation* of $A$.

We refer the reader to Ref. 2,3,12, for properties and results concerning the approximate displacement rank.

In order to construct orthogonal $\epsilon$-displacement generators for a matrix $A$, let us introduce the operator $\text{Trunc}_\epsilon(\cdot)$ defined on the sets of orthogonal displacement generators in the following way:

$$\begin{aligned} &\text{Trunc}_\epsilon(\boldsymbol{a}, U, \boldsymbol{\sigma}, V) = (\boldsymbol{a}, \widehat{U}, \widehat{\boldsymbol{\sigma}}, \widehat{V}), \\ &\widehat{\boldsymbol{\sigma}} = (\sigma_1, \ldots, \sigma_{\hat{k}}), \quad \widehat{U} = [U\boldsymbol{e}_1, \ldots, U\boldsymbol{e}_{\hat{k}}], \widehat{V} = [V\boldsymbol{e}_1, \ldots, V\boldsymbol{e}_{\hat{k}}] \in \boldsymbol{R}^{n \times \hat{k}} \\ &\sigma_{\hat{k}} > \sigma_1 \epsilon \ge \sigma_{\hat{k}+1}, \end{aligned} \tag{17}$$

for a general $A$, and in the following way for a persymmetric matrix $A$

$$\begin{aligned} &\text{Trunc}_\epsilon(\boldsymbol{a}, W, \boldsymbol{d}) = (\boldsymbol{a}, \widehat{W}, \widehat{\boldsymbol{d}}), \\ &\widehat{\boldsymbol{d}} = (d_1, \ldots, d_{\hat{k}/2}), \quad \hat{W} = [W\boldsymbol{e}_1, \ldots, W\boldsymbol{e}_{\hat{k}}] \in \boldsymbol{R}^{n \times \hat{k}} \\ &|d_{\hat{k}}| > |d_1| \epsilon \ge |d_{\hat{k}+1}|. \end{aligned} \tag{18}$$

Equations (17), (18) and Algorithm 2.4 provide a means for computing an $\epsilon$-ODG of $A$ whenever any orthogonal displacement representation is available.

## 3. NEWTON'S ITERATION FOR TOEPLITZ MATRICES

With the tools introduced in the previous section we are able to define our regularized version of Newton's iteration adjusted to cope with the case of Toeplitz matrices, based on Ref. 2,3. Local convergence analysis of a similar modification of Newton's iteration has been performed in Ref. 8.

If $X_0$ is such that $X_0 = JX_0^TJ$, then all the matrices $X_i$ satisfy the equation $X_i = JX_i^TJ$ and we may represent each $X_i$ by means of its persymmetric orthogonal displacement generator of Definition 2.3. Observe that the displacement rank of $X_i$ may grow as $i$ tends to infinity, even though the limit $A^{-1}$ of $X_i$ has displacement rank 2. However we may overcome this drawback by means of the $\epsilon$-displacement rank. In fact, for any $\epsilon > 0$, the value $\mathrm{drk}_\epsilon(X_i)$ is asymptotically bounded by $\mathrm{drk}(A^{-1})$.

We will analyze the modification of Newton's iteration defined by the following recurrence

$$\begin{aligned} Y_i &= 2X_i - X_iAX_i \\ X_{i+1} &= \mathrm{Trunc}_{\epsilon_i}(Y_i) \end{aligned} \tag{19}$$

where $\mathrm{Trunc}_\epsilon(\cdot)$ is defined in (18) and the choice of $\epsilon_i$ can be made according to different strategies. The simplest one is to choose a constant value of $\epsilon_i$, say, the square root of the machine precision of the computation. A dynamic choice of $\epsilon_i$ that guarantees the convergence of the modified Newton method can be easily designed. We recall the following result of Ref. 2,3.

THEOREM 3.1. *Let $X_0$ be such that $||X_0|| \le 1$ and set $\gamma = 1 - ||X_0||$, $\epsilon_i = \gamma/(4n(2h_i + k)\sigma_1^{(i)}||A||)$, $h_i = \mathrm{drk}(X_i)$, $\sigma_1^{(i)} = \rho(\Delta(X_i))$. Then for the residuals $R_i = I - X_iA$ of the sequence $\{X_i\}_i$ generated by the modified Newton iteration it holds $||R_i|| \le (1-\gamma/2)^{2^i}$.*

For $\theta = 1/\mathrm{cond}_2(A)$, the bound on $\epsilon$ turns into $\epsilon \le ||R_i||^2/(4n(2h_i+k)\sigma_1\mathrm{cond}_2(A))$. Indeed, this bound guarantees the convergence whatever the matrix $A$ is. However, due to the regularization properties, we do not need to perform many steps of Newton's iteration, rather, the value of $\log_2\theta^{-1}$ is customarily close to 1 so that our regularized version of Newton's iteration stops at the very early stages.

In order to analyze the iteration (19) we will relate $\mathrm{ODG}(X) = (\boldsymbol{x}, W_X, \boldsymbol{d}_X)$ with $\mathrm{ODG}(Y) = (\boldsymbol{y}, W_Y, \boldsymbol{d}_Y)$, where $Y = 2X - XAX$.

It is immediate to verify that:

$$\begin{aligned} \boldsymbol{y} &= 2\boldsymbol{x} - XA\boldsymbol{x} \\ \Delta(Y) &= FMF^TJ \\ M &= \begin{bmatrix} 2D_X & O & -D_X \\ 0 & -D_A & O \\ -D_X & O & O \end{bmatrix} \otimes \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \\ F &= \begin{bmatrix} W_X & XW_A & XAW_X \end{bmatrix} \end{aligned} \tag{20}$$

Recovering $\mathrm{ODG}(Y)$ from (20) is almost inexpensive if performed by means of Algorithm 2.4. Now we are ready to describe our implementation of Newton's iteration based on (19), where we use the infinite norm of a matrix $A$ defined by $||A||_\infty = \max_i \sum_j |a_{i,j}|$.

ALGORITHM 3.2. [*Regularized Newton's iteration with $\epsilon$-displacement rank*].

INPUT: Integer $n$, the $n \times n$ Toeplitz matrix assigned in terms of its first row and its first column; a strategy for the choice of the truncation value $\epsilon_i$; the number $k$ of steps.

OUTPUT: A regularized approximation $X$ of $A^{-1}$ given in terms of its ODG $(\boldsymbol{x}, W_X, \boldsymbol{d}_X)$.

COMPUTATION:

– Compute $\theta = 1/||A||_\infty||A^T||_\infty$, set $X_0 = \theta A^T$, $i = 0$ and compute $\mathrm{ODG}(A)$ and $\mathrm{ODG}(A^T)$.

– For $i = 0, 1, \ldots, k-1$ compute $(\boldsymbol{y}, W_Y, \boldsymbol{d}_Y) = \mathrm{ODG}(Y)$, where $Y = 2X_i - X_i A X_i$, by means of (20) and Algorithm 2.4.
– Compute $\mathrm{ODG}(X_{i+1}) = \mathrm{Trunc}_{\epsilon_i}(\mathrm{ODG}(Y))$, by means of (18).
– output $\mathrm{ODG}(X_k)$.

The algorithm can be easily modified by introducing the preconditioning stage where $X_0 = \theta C C^T A^T$ for a suitable matrix $C$ having low displacement rank, or $X_0 = \frac{\theta}{2}(CC^T A^T + A^T C^T C)$ if we want to preserve the persymmetry of the iterates.

### 3.1. The Separable Case

Consider the system (2) where $A = T_1 \otimes T_2$ and $T_1$, $T_2$ are (banded) Toeplitz matrices. This case is encountered in practice, for instance, for the PSF given by the mask $M = (m_{i,j})_{i,j=-k,k}$, where $m_{i,j} = 1/\alpha$, $\alpha = (2k+1)^2$; in this case $T_1 = T_2$ is a $2k+1$ diagonal Toeplitz matrix whose nonzero entries are equal to $1/(2k+1)$. Another interesting case is given by the mask $M = (m_{i,j})$ defined by a Gaussian function, i.e., $m_{i,j} = \beta \exp(-\alpha i^2 - \alpha j^2)$. In fact, in this case $T_1 = T_2$ is the $(2k+1)$-diagonal Toeplitz matrix whose nonzero entries are $\sqrt{\beta}\exp(-\alpha i^2)$.

By representing the unknown vector $\boldsymbol{y}$ by means of a matrix $Y$ and $\boldsymbol{b}$ by $B$ where $\boldsymbol{y} = vec(Y)$, $\boldsymbol{b} = vec(B)$, the system $A\boldsymbol{y} = \boldsymbol{b}$ turns into $T_2 Y T_1^T = B$, from which we have $Y = T_2^{-1} B T_1^{-T}$. A regularized approximation of $Y$ is given by $T_2^+ B T_1^+$, where $T_2^+$ and $T_1^+$ are regularized approximations to $T_1^{-1}$ and $T_2^{-1}$, respectively, obtained by applying a suitable (small) number of steps of Newton's iteration to $T_1$ and $T_2$. The computation of $T_2^+ B T_1^+$ consists in performing 2 products between a Toeplitz-like matrix and a general $n \times n$ matrix (if the image has size $n \times n$); this computation requires $O(n^2 \log n)$ arithmetic operations.

### 3.2. The General Case

Let $M = \sum_{j=1}^{2k+1} \sigma_j \boldsymbol{u}_j \boldsymbol{v}_j^T$ be the SVD of the mask $M$ associated with the PSF matrix $A$. Let $T_1^{(j)} = \sqrt{\sigma_j} U_j$, $T_2^{(j)} = \sqrt{\sigma_j} V_j$, where $U_j$ and $V_j$ are the $(2k+1)$-diagonal Toeplitz matrices defined by the vectors $\boldsymbol{u}_j$ and $\boldsymbol{v}_j$, respectively. Let $(T_1^{(1)})^+$ and $(T_2^{(1)})^+$ be the regularized approximations to $T_1^{(1)}$ and $T_2^{(1)}$ provided by a suitable number of steps of Newton's iteration. Consider the splitting

$$A = \left((T_1^{(1)})^+ \otimes (T_2^{(1)})^+\right)^{-1} - \left(\left((T_1^{(1)})^+ \otimes (T_2^{(1)})^+\right)^{-1} - A\right)$$

and observe that the iterative scheme associated with this partitioning is (3); moreover, the iteration matrix is $P = I - \left((T_1^{(1)})^+ \otimes (T_2^{(1)})^+\right) A$. If $\rho\left(((T_1^{(1)})^+ \otimes (T_2^{(1)})^+)A\right) < 1$ the iterative scheme (3) converges. If the singular values and the singular vectors of $A$ do not differ much from the corresponding singular values and the singular vectors of $T_1^{(1)} \otimes T_2^{(1)}$ then the iterative scheme has regularizing properties.

## 4. NUMERICAL EXPERIMENTS

We have applied our technique to the case where $A$ is the two-level banded Toeplitz matrix defined by the $31 \times 31$ mask $m_{i,j} = \beta e^{-\alpha(i^2+j^2)}$, $i,j = -15, 15$, with $\alpha = 0.1$, and $\beta$ so that $\sum_{i,j} m_{i,j} = 1$. In this way it holds $A = T \otimes T$, $T = (t_{i,j})$, $t_{i,j} = \sqrt{\beta} e^{-\alpha(i-j)^2}$ if $|i-j| \leq 15$, $t_{i,j} = 0$ elsewhere. We set $n = m = 512$, i.e., our image is formed by $512^2$ pixels, where each pixel is represented by an integer between 0 and 255. We have considered the case where the noise $\boldsymbol{\eta}$ is originated by the numerical rounding to the integer part of the blurred image (*numerical noise*) which has real entries.

We have applied Newton's iteration with the parameter $\theta = 1$, in fact it holds $\sigma_1 \leq 1$, and with the initial approximation $X_0 = \theta I$ and $X_0 = T^T$. The results of the numerical experiments are reported in figure 2.

The regularized features of Newton's iteration are evident from figure 2: in fact, three iterations with $X_0 = I$ are enough to provide a good approximation of the image still keeping the noise under a suitable level, while a larger number of iterations amplifies the error. With the choice $X_0 = T^T$ the number of iterations sufficient to recover the image is larger (according to our theoretical estimates): 8 iterations are enough to recover the image without much amplifying the noise. With 10 iterations the noise slightly appears in the image. It is interesting that the

Original image

Blurred image

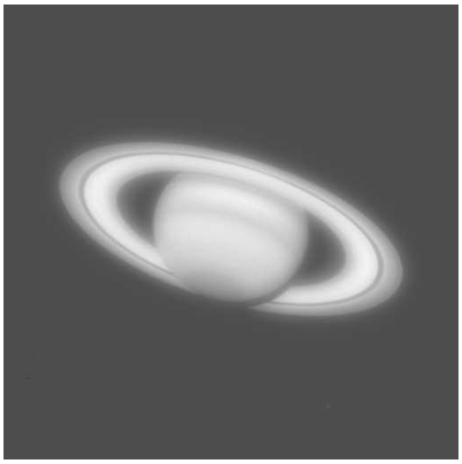
3 Newton's iterations with $X_0 = I$

4 Newton's iterations, $X_0 = I$

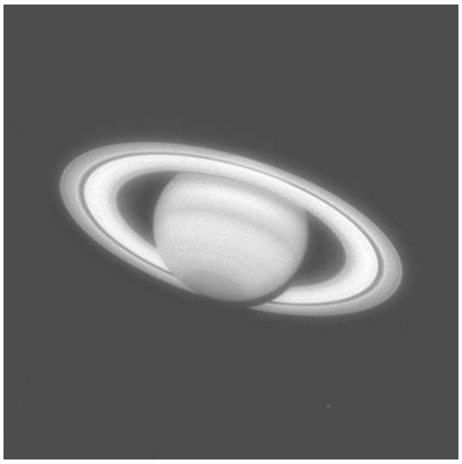
8 Newton's iterations, $X_0 = T^T$

10 Newton's iterations, $X_0 = T^T$

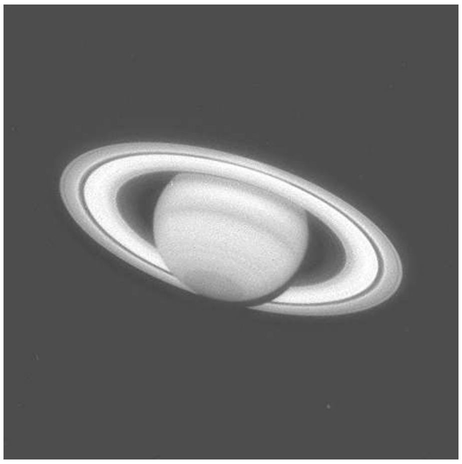
**Figure 2.** Separable PSF



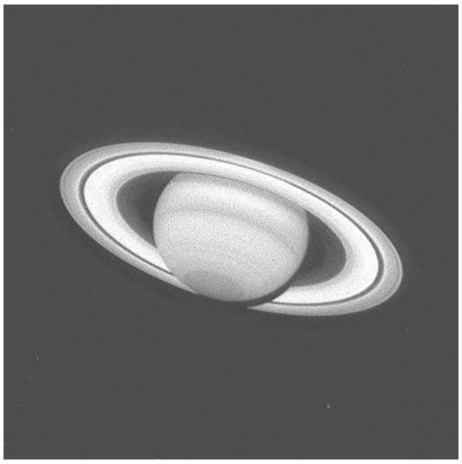
Blurred image



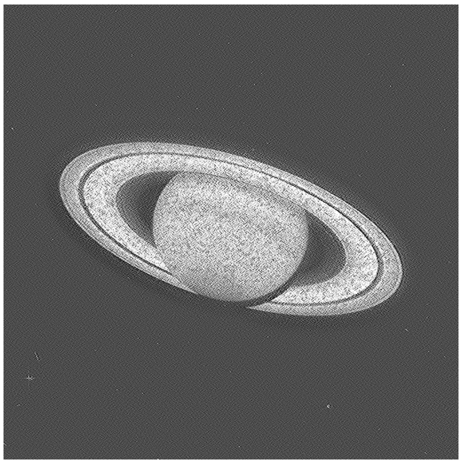
$it = 1,\ k = 3,\ \theta = 0.1$



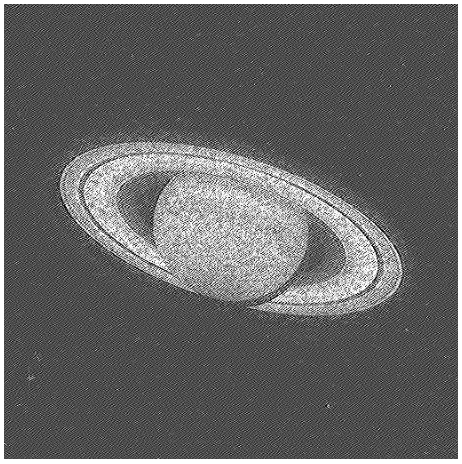
$it = 2,\ k = 3,\ \theta = 0.1$



$it = 4,\ k = 3,\ \theta = 0.1$



$it = 1,\ k = 6,\ \theta = 0.1$



$it = 2,\ k = 6,\ \theta = 0.1$

**Figure 3.** Nonseparable PSF

amplification of the noise in the cases $X_0 = I$ and $X_0 = T^T$ occurs with different features, in particular the noise is coarse-grained if $X_0 = T^T$.

We have applied our technique to the reconstruction of the image of Saturn blurred with a nonseparable PSF, that can be obtained by ftp from `ftp.stsci.edu` (see also Ref. 13,1).

We report the images obtained by applying *it* iteration of our scheme (3) where the regularized inversion of the matrices $T_1^{(1)}$ and $T_2^{(1)}$ is obtained by means of $k$ Newton's iterations with the initial point $X_0 = \theta I$. The results are reported in figure 3. We have chosen $\theta = 0.1$ since a larger value, say $\theta = 1$ still guaranteeing convergence, would have caused a too fast convergence of Newton's iteration with a consequent amplification of the noise even with a single step. As we can see from figure 3, the image obtained after 4 iterations of (3), where the regularized inverses of $T_1^{(1)}$ and $T_2^{(1)}$ are obtained with 3 Newton's steps and with $X_0 = \theta I$, $\theta = 0.1$, is the one which provides a good reconstruction with a very small noise. If we use more accurate approximations of the inverses of $T_1^{(1)}$ and $T_2^{(1)}$, say with $k = 6$ Newton's steps, the amplification of the noise is very strong.

## REFERENCES


1. M. Bertero and P. Boccacci, *Introduction to inverse problems in imaging*, Institute of Physics Publishing, Bristol, 1998.
2. D. A. Bini and B. Meini, "Solving block banded block Toeplitz systems with structured blocks: algorithms and applications," in *Structured matrices: recent developments in theory and computation*, D. A. Bini, E. Tyrtyshnikov, and P. Yalamov, eds., NOVA, 2000. In press.
3. D. A. Bini and B. Meini, "Approximate displacement rank and applications," in *Proceedings AMS Conference "Structured Matrices in Operator Theory, Numerical Analysis, Control, Signal and Image Processing", Boulder*, V. Olshevsky, ed., AMS, 1999. to appear.
4. T. Kailath, S. Kung, and M. Morf, "Displacement ranks of matrices and linear equations," *J. Math. Anal. Appl.* **68**, pp. 395–407, 1979.
5. J. Kamm and J. G. Nagy, "Kronecker product and SVD approximations in image restoration," *Linear Algebra Appl.* **284**(1-3), pp. 177–192, 1998. ILAS Symposium on Fast Algorithms for Control, Signals and Image Processing (Winnipeg, MB, 1997).
6. J. Kamm and J. G. Nagy, "Optimal Kronecker product approximation of block Toeplitz matrices," *SIAM J. Matrix Anal. Appl.* **22**(1), pp. 155–172, 2000.
7. D. A. Bini and V. Pan, *Matrix and Polynomial Computations, Vol. 1: Fundamental Algorithms*, Birkhäuser, Boston, 1994.
8. V. Y. Pan, S. Branham, R. E. Rosholt, and A.-L. Zheng, "Newton's iteration for structured matrices," in *Fast reliable algorithms for matrices with structure*, T. Kailath and A. H. Sayed, eds., ch. 7, pp. 189–210, SIAM, Philadelphia, 1999.
9. R. H. Chan and M. K. Ng, "Conjugate gradient methods for Toeplitz systems," *SIAM Review* **38**, pp. 427–482, 1996.
10. T. Kailath, S. Kung, and M. Morf, "Displacement rank of a matrix," *Bulletin of the American Mathematical Society* **1**, pp. 769–773, 1979.
11. G. Heinig and K. Rost, *Algebraic Methods for Toeplitz-like Matrices and Operators*, Akademie-Verlag, Berlin, and Birkhäuser, Boston, 1984.
12. D. A. Bini and B. Meini, "Solving block banded block Toeplitz systems with banded Toeplitz blocks," in *Advanced Signal Processing Algorithms, Architectures, and Implementations IX*, F. T. Luk, ed., vol. 3807, pp. 300–311, Proceedings of SPIE, 1999.
13. J. G. Nagy, "Decomposition of block Toeplitz matrices into a sum of Kronecker products with applications in image restoration," Tech. Rep. 96-1, SMU Department of Mathematics, 1996.