Parallel deconvolution methods for three dimensional image restoration

B. Lewis [a] and L. Reichel [b]

[a]Rocketcalc, LLC, Richfield, OH 44286

[b]Department of Mathematical Sciences, Kent State University, Kent, OH 44242

ABSTRACT

Restoration by deconvolution of three-dimensional images that have been contaminated by noise and spatially invariant blur is computationally demanding. We describe efficient parallel implementations of iterative methods for image deconvolution on a distributed memory computing cluster.

Keywords: Image restoration, ill-posed problem, Krylov iterative methods, parallel computation

1. INTRODUCTION

Three-dimensional images arise in a variety of applications in science and engineering, such as in confocal microscopy and biomedical imaging. This paper considers the computational problem of restoring three-dimensional images that have been degraded by noise and spatially invariant blur. The blurring of an image by spatially invariant blur can be modeled by a convolution,

$$g(\xi,\eta,\gamma) = \int_{\mathbf{R}^3} h(\xi-\xi',\eta-\eta',\gamma-\gamma')f(\xi',\eta',\gamma')d\xi'd\eta'd\gamma', \tag{1}$$

where the function f represents the blur-free image and the function g represents the blurred image. The smooth kernel h often is referred to as a *point-spread function.* The problem of determining the function f, given the function g and kernel h, by solving the integral equation (1) is ill-posed, because the solution f, when it exists, does not depend continuously on the function g; see, e.g., Groetsch[8] for a discussion.

We assume that the functions f and g in (1) are identically zero outside the unit cube $\Omega \subset \mathbf{R}^3$, and that equation (1) is discretized on a regular $m \times m \times m$ mesh to yield a system of linear (algebraic) equations

$$Ax = b, \qquad A \in \mathbf{R}^{m^3 \times m^3}, \qquad x, b \in \mathbf{R}^{m^3}, \tag{2}$$

where x and b represent discretizations of the functions f and g in (1), respectively. Specifically, the vectors x and b contain lexicographically ordered pixel values, and the matrix A represents a discretization of the integral operator (1); it is determined by the point-spread function h and the discretization method. Typically, the matrix A has many singular values of different orders of magnitude close to the origin; thus, A is of ill-determined rank. In particular, the matrix A is severely ill-conditioned and may be singular. We refer to linear systems of equations (2) with a matrix of ill-determined rank as discrete ill-posed problems. Recent discussions of discrete ill-posed problems in image processing are provided by, e.g., Hansen[11,12] and Jain.[13]

In typical image restoration problems that arise in science and engineering, the right-hand side b in (2) is not available. Instead, the vector

$$\tilde{b} := b + d \tag{3}$$

is known, where the entries of the vector $d \in \mathbf{R}^{m^3}$ contain measurement and transmission errors. We refer to d as "noise." Thus, the vector $\tilde{b}$ represents an available discretized image that is contaminated by blur and noise.

E-mail: B.L.: blewis@rocketcalc.com, L.R.: reichel@math.kent.edu

The desired, but unknown, discretized image x solves the linear systems of equations (2) with the unknown right-hand side vector b. We seek to determine an approximation of x by computing an approximate solution of the linear system of equations

$$A\tilde{x} = \tilde{b}, \qquad \tilde{x}, \tilde{b} \in \mathbf{R}^{m^3}, \tag{4}$$

with the available right-hand side (3). Let $A^\dagger$ denote the Moore-Penrose pseudoinverse of the matrix A. Then $\tilde{x} := A^\dagger \tilde{b}$ is the least-squares solution of (4) of minimal Euclidean norm. Due to the severe ill-conditioning of A and the noise d in the right-hand side $\tilde{b}$, the least-squares solution $\tilde{x}$ of (4) typically is a poor approximation of the desired solution x of (2).

In order to determine a meaningful approximation of x, the linear system of equations (4) is replaced by a nearby system, whose solution is less sensitive to the noise d in the right-hand side $\tilde{b}$ than the solution of (4). This replacement is referred to as regularization. We will regularize by applying ℓ steps of a Krylov subspace iterative method to the solution of (4), with ℓ chosen sufficiently small.

Krylov subspace methods are popular iterative methods for the solution of large-scale linear systems of equations. The ℓth Krylov subspace for the matrix A and the vector $\tilde{b}$ is defined by

$$\mathcal{K}_\ell(A, \tilde{b}) := \text{span}\{\tilde{b}, A\tilde{b}, A^2\tilde{b}, \ldots, A^{\ell-1}\tilde{b}\}. \tag{5}$$

Krylov subspace iterative methods determine approximate solutions of (4) by projecting the system of equations into a Krylov subspace of (low) dimension ℓ, and solving the (small) projected problem. The GMRES method is one of the most popular Krylov subspace iterative methods for the solution of linear systems of equations that arise from the discretization of well-posed problems. Calvetti et al.[2] have shown that under suitable conditions the GMRES method is a regularization method in a well-defined sense, provided that the number of iterations ℓ is chosen appropriately. Previously, Hanke[9] and Nemirovskii[15] showed related results for the conjugate gradient method and variants thereof. Let the initial approximate solution be $\tilde{x}_0 = 0$. Then the ℓth approximate solution, $\tilde{x}_\ell$, determined by the GMRES iterative method lives in the Krylov subspace (5).

The subspace dimension ℓ can be thought of as a regularization parameter. Some common Krylov subspace iterative methods and the subspaces in which the computed approximate solutions $\tilde{x}_\ell$ live are listed in Table 1. The initial approximate solution is assumed to be $\tilde{x}_0 = 0$; see, e.g., Saad[16] for further details on Krylov subspace iterative methods. The RRGMRES is discussed by Calvetti et al..[1,6] Applications of several Krylov subspace iterative methods to image restoration have recently been discussed by Calvetti et al.,[3–6] Hanke and Nagy,[10] and Lee et al..[14]

Solution subspace	Iterative methods
$\mathcal{K}_\ell(A^TA, A^T\tilde{b})$	CGNR
$\mathcal{K}_\ell(A, \tilde{b})$	GMRES, BiCG, QMR, FOM
$\mathcal{K}_\ell(A, A\tilde{b})$	RRGMRES

Table 1. Some Krylov subspace iterative methods and their Krylov subspaces

Each iteration with a Krylov subspace method of Table 1 requires the evaluation of either one matrix-vector product, Au, $u \in \mathbf{R}^{m^3}$ (GMRES, RRGMRES, and FOM), or of two matrix-vector products, Au and A^Tw, $u, w \in \mathbf{R}^{m^3}$ (CGNR, BiCG, QMR). Since the matrix A models spatially invariant blur, the matrix-vector products can be evaluated by using the fast Fourier transform (FFT), provided that suitable boundary conditions on the data in equation (1) are imposed; see, e.g., Jain[13] for details. The application of the FFT can lead to significant reduction of the computational work required by a Krylov subspace iterative method, because typically the evaluation of matrix-vector products make up the bulk of the computational work required by these methods.

It is the purpose of the present paper to describe an efficient parallel implementation of an FFT-based method for matrix-vector product evaluation. Computed examples for clusters of 1, 4, and 16 processes with distributed memory illustrate the performance of our scheme. We remark that while easy-to-use software for deconvolution has recently been made available by Lee et al.,[14] we are not aware of public-domain software for efficient parallel

deconvolution of three-dimensional images. The methods discussed in this paper are a first step towards a public-domain library for numerical methods for deconvolution of three-dimensional images by iterative methods on a computational cluster.

The enormous size of the linear systems (4) that arise in three-dimensional image deconvolution problems poses significant computational challenges. In order to simplify the computations, the point-spread function sometimes is assumed to vanish except on a two-dimensional subspace, e.g., the point-spread function may be allowed to be nonvanishing in the xy-plane only. Then blur in the xz- and yz-planes is ignored. This simplification makes it possible to decouple the three-dimensional image deconvolution problem into m independent two-dimensional problems, and the deconvolution method is applied to each one of the m two-dimensional image planes independently. Restoration methods that require the point-spread function to vanish outside a two-dimensional subspace are sometimes referred to as "no neighbor" methods in the image processing literature. We will refer to these methods as *planar methods*. While planar methods typically require significantly less computational work than more general methods for three-dimensional image deconvolution, they can give inferior quality of the restored three-dimensional image. The latter is illustrated in Section 3.

2. IMPLEMENTATION

It is often impossible or impractical to implement deconvolution methods for the linear systems of equations (4) that arise in three-dimensional image restoration on a single-processor workstation because of the large size of the linear system. Therefore, simplifying assumptions are often imposed on the problem in order to be able to reduce the problem to manageable size. However, the simplifications can affect the quality of the computed solution adversely.

The wide availability of inexpensive computing clusters makes implementation of deconvolution methods for three-dimensional images practical. We are developing a library and an Application Programming Interface (API) that implement distributed computation for Krylov subspace iterative methods for deconvolution problems.

The computation of matrix-vector products Av, or, equivalently, the computation of convolutions of the discretized point-spread function with discretized images v, typically dominates the computational work required by the GMRES method. Convolution (blurring) amounts to replacing each pixel of the discrete image by a weighted average of pixels in a neighborhood. The size of the neighborhood is determined by the support of the point-spread function. The localized action of the convolution on the image pixels suggests that a three-dimensional image be subdivided into cubes or rectangular prisms as shown in Figure 1. Each subdomain can, for example, be associated with a processor, or, more generally, with one or several processes. The neighborhood around pixels near the boundary of each subdomain extends into adjacent subdomains. This is illustrated in Figure 1 by the dashed lines. Values of pixels in these regions must be exchanged between processes associated with adjacent subdomains.

The software library uses message passing to distribute the subdivided three-dimensional image to processes. The processes communicate using message passing to exchange boundary data when required. The computation of the convolution for each subdomain can be carried out efficiently by using the FFT. The library employs the recently developed high-performance implementation of the FFT by Frigo and Johnson,[7] as well as tuned basic linear algebra software from the ATLAS project.[17] These software components are combined to provide an efficient implementation of a collection of distributed Krylov subspace iterative methods for deconvolution. Algorithm 2.1 below outlines the computations required for the GMRES method.

ALGORITHM 2.1. *Distributed GMRES method for deconvolution.*

1. *Subdivide the domain into subdomains and distribute the kernel and subdomains to processes.*
2. *Processes communicate to compute the next Krylov subspace basis vector:*
   (a) *Exchange boundary data.*
   (b) *Compute matrix-vector product.*

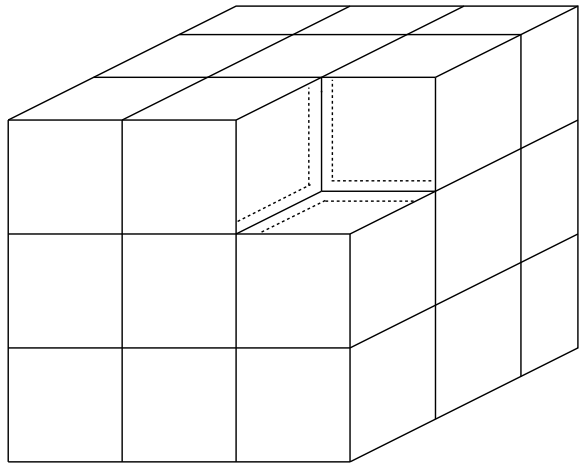

Figure 1. Illustration of image subdivision.

   (c) Orthogonalize.

3. *Project the problem into the Krylov subspace and solve the projected problem.*

4. *Evaluate stopping criterion (using the projected data):*

   (a) If the stopping criterion is not satisfied, then increase subspace dimension and goto step 2.

   (b) If the stopping criterion is satisfied, then return computed approximate solution of (4) and exit.

Other Krylov subspace iterative methods, such as the QMR and BiCG methods, can be implemented in a similar way as outlined by Algorithm 2.1. The implementations of distributed Krylov subspace iterative methods obtained in this manner are well suited for deployment on widely available Beowulf clusters and other parallel distributed memory computers. They can be used to solve deconvolution problems that are too large to be solved practically on single-processor workstations.

Parallel computational methods on distributed memory clusters can be difficult to implement and often lack portability. We strive to overcome these difficulties by writing portable code in C and developing an easy to use API for the numerical library. The API provides a small set of functions that can be called from sequential applications. These functions are high-level interfaces to the library routines and hide the parallel computations from the calling program. The API includes functions that:

- Determine a subdivision and distribute data (initialization).
- Increase the dimension of the Krylov subspace and carry out one step of the iterative method.
- Compute inner products.
- Compute linear combinations of vectors.

Deconvolution methods using the API can be written in a high level sequential programming language, such as C, Matlab using the Mex interface, or Octave.

3. NUMERICAL EXPERIMENTS

We consider the restoration of a noisy and blurred three-dimensional test image. The blur- and noise-free test image has parallel equidistant bars in a z-plane. We refer to these bars as "horizontal," since they are shown as horizontal bars in the left-hand side image of Figure 2, which displays the three-dimensional test image seen from above. The horizontal bars are located above parallel equidistant bars in a lower z-plane. The latter bars are perpendicular to the horizontal bars, and we refer to them as "vertical," since they are the vertical bars in the left-hand side image of Figure 2. The right-hand side image of Figure 2 displays the z-plane with the horizontal bars. For display purposes, each image in this section is scaled so that the image pixel values lie in the interval $[0, 255]$.

The blur- and noise-free test image consists of a $100 \times 100 \times 100$ regular mesh of grayscale pixel values in the range $[0, 255]$. Thus $m = 100^3$ in (2) and (4). For computational simplicity, we use the mesh points (i, j, k), with $1 \leq i, j, k \leq 100$, for the discretized three-dimensional image. This amounts to a rescaling of Ω. The discrete convolution kernel is defined by

$$h_{ijk} := \begin{cases} c \exp\left(-\dfrac{(i-50)^2 + (j-50)^2 + (k-50)^2}{2}\right), & 25 \leq i, j, k \leq 75, \\ 0, & \text{otherwise}, \end{cases}$$

where the scaling factor c is chosen so that $\sum_{i,j,k} h_{ijk} = 1$. The resulting discretized point-spread function is symmetric. The horizontal bars are in the plane $k = 52$ and the vertical bars in the plane $k = 48$. Thus, the planes of the horizontal and vertical bars are four pixels apart.

The noise vector d in (3) is constructed in the following manner. Let the entries of the vector x contain the lexicographically ordered pixel values of the blur- and noise-free test image, and let A be the blurring matrix. Then the vector $b := Ax$ represents the blurred, but noise-free, image associated with the test image represented by x. The noise vector $d \in \mathbf{R}^{m^3}$ has normally distributed random entries with zero mean, scaled so that $\|d\| = 0.01\|Ax\|$, where $\|\cdot\|$ denotes the Euclidean vector norm. The right-hand side vector of the linear system (4) is given by (3). The image plane corresponding to the horizontal bars for the blurred and noisy image is displayed by the left-hand side image of Figure 3.

All computations were carried out using Octave 2.1.36 on a Rocketcalc cluster of 16 Intel Xeon processors connected by gigabit ethernet. Double precision arithmetic with about 16 significant decimal digits was used.

The quality of the restored two-dimensional image plane containing the horizontal bars serves as a benchmark for comparing different deconvolution methods. We compare the GMRES method applied to the linear system (4) to a planar method using the GMRES method on the image plane containing the horizontal bars. The initial approximate solution $\tilde{x}_0 = 0$ is used for the GMRES method.

The image blurring in this example is computed by embedding the matrix A in a block-circulant matrix and applying the FFT. The inverse of this block-circulant matrix provides an approximation of the inverse convolution. This approximation can be effective in low-noise applications, and it is also sometimes used to construct a preconditioner. However, attempting to simply apply the approximate inverse to the right-hand side $\tilde{b}$ of (4) to determine an approximation of the desired noise- and blur-free image x is ill-advised due to the discrete ill-posed nature of the problem and the presence of noise in $\tilde{b}$. The right-hand side image of Figure 3 displays a naive "restoration" obtained by this approach.

The left-hand side image of Figure 4 shows the restored image determined by 7 steps of the GMRES method applied to the linear system (4). The right-hand side image of Figure 4 displays the restored image produced by the planar deconvolution method with 7 steps of the GMRES method applied to the image plane containing the horizontal bars. The planar method is seen to yield a restored image of inferior quality.

The test image is small enough that the restoration can be computed using one processor in the cluster. We also ran the example on four and 16 processors to evaluate parallel scalability. Figure 5 displays the ratio of wall-clock computation time for n processors to wall-clock computation time for 1 processor, for $n = 1, 4, 16$. Maximum speed-up is shown for reference (i.e., four times faster on four processors, 16 times faster on 16 processors). This example is referred to as the "small problem" in the figure.

The speed-up results for a larger problem constructed in the same way as the above example on an $m \times m \times m$ mesh with $m = 250$ are also displayed in Figure 5. Speed-up values are given relative to four processors, since this larger problem was too big to be run on one processor. We remark that parallel scalability depends on many problem- and algorithm-specific factors. Figure 5 shows the scalability of this example to be quite good.

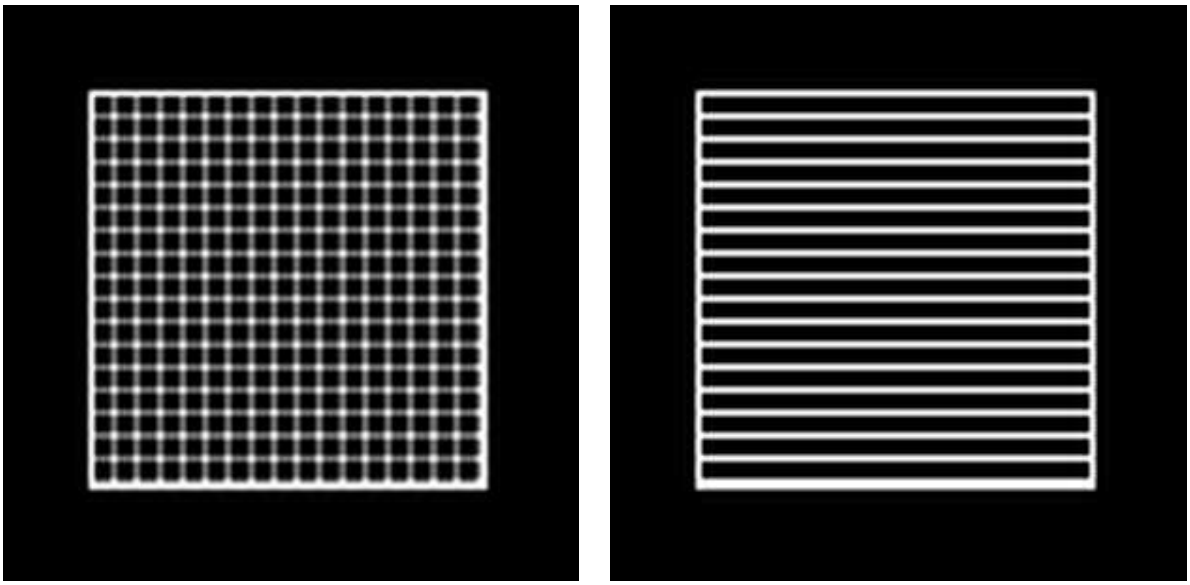

Figure 2. Test image seen from above (left). Plane of interest only (right).

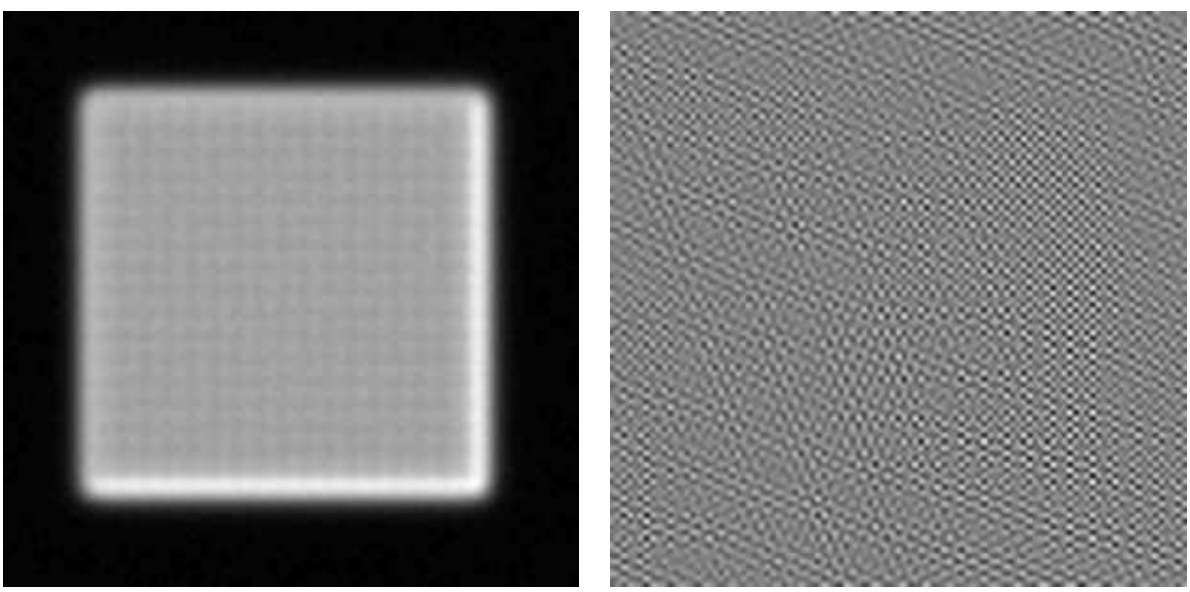

Figure 3. Image plane of interest of the available blurred and noisy image (left). Naive "restoration" obtained by a block-circulant approximation of the inverse convolution (right).

4. ACKNOWLEDGEMENT

This work was supported in part by NSF grant DMS-0107858.

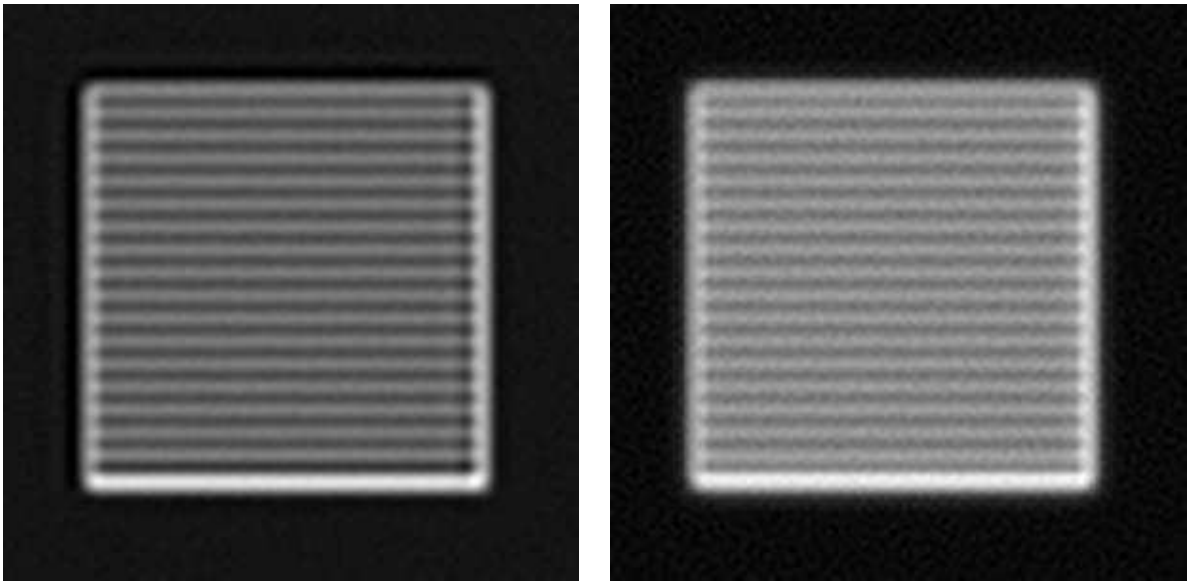

Figure 4. Restored images obtained by three-dimensional restoration (left) and by planar restoration (right). Both restored images were determined by 7 steps of the GMRES method.

REFERENCES

1. D. Calvetti, B. Lewis, and L. Reichel, GMRES-type methods for inconsistent systems, Linear Algebra Appl., 316 (2000), pp. 157–169.
2. D. Calvetti, B. Lewis, and L. Reichel, On the regularizing properties of the GMRES method, Numer. Math., 91 (2002), pp. 605–625.
3. D. Calvetti, B. Lewis, and L. Reichel, GMRES, L-curves and discrete ill-posed problems, BIT, 42 (2002), pp. 44–65.
4. D. Calvetti, B. Lewis, and L. Reichel, Restoration of images with spatially variant blur by the GMRES method, in Advanced Signal Processing Algorithms, Architectures, and Implementations X, ed. F. T. Luk, Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE), vol. 4116, The International Society for Optical Engineering, Bellingham, WA, 2000, pp. 364–374.
5. D. Calvetti, B. Lewis, and L. Reichel, Krylov subspace iterative methods for nonsymmetric discrete ill-posed problems in image restoration, in Advanced Signal Processing Algorithms, Architectures, and Implementations XI, ed. F. T. Luk, Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE), vol. 4474, The International Society for Optical Engineering, Bellingham, WA, 2001, pp. 224–233.
6. D. Calvetti, B. Lewis, and L. Reichel, On the choice of subspace for iterative methods for linear discrete ill-posed problems, Int. J. Appl. Math. Comput. Sci., 11 (2001), pp. 1069–1092.
7. M. Frigo and S. G. Johnson, FFTW: An adaptive software architecture for the FFT, IEEE Proc. ICASSP 1998, vol. 3, pp. 1381–1384.
8. C. W. Groetsch, The Theory of Tikhonov Regularization for Fredholm Equations of the First Kind, Pitman, Boston, 1984.
9. M. Hanke, Conjugate Gradient Type Methods for Ill-Posed Problems, Longman, Essex, 1995.
10. M. Hanke and J. G. Nagy, Restoration of atmospherically blurred images by symmetric indefinite conjugate gradient techniques, Inverse Problems, 12 (1996), pp. 157–173.
11. P. C. Hansen, Rank-Deficient and Discrete Ill-Posed Problems, SIAM, Philadelphia, 1998.
12. P. C. Hansen, Deconvolution and regularization with Toeplitz matrices, Numer. Algorithms, 29 (2002), pp. 323–378.
13. A. K. Jain, Fundamentals of Digital Image Processing, Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

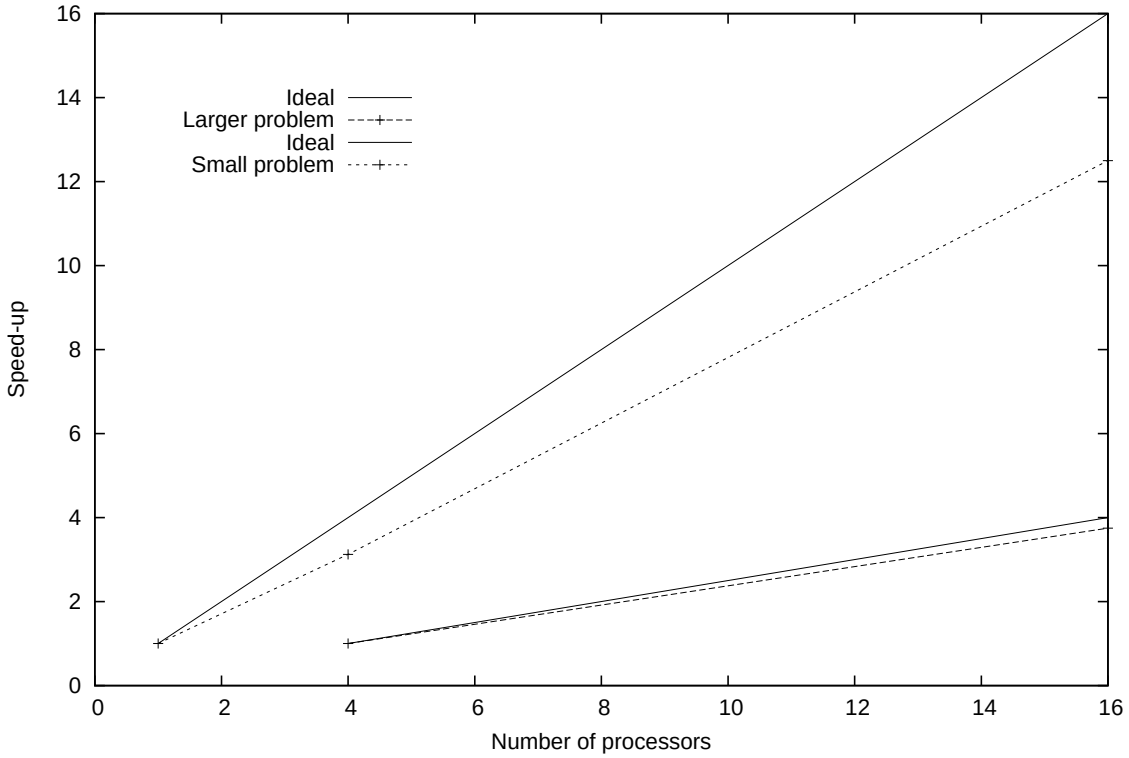

Figure 5. Parallel performance.

14. K. P. Lee, J. G. Nagy, and L. Perrone, Iterative methods for image restoration: A Matlab object oriented approach, submitted for publication, 2002.
15. A. S. Nemirovskii, The regularization properties of the adjoint gradient method in ill-posed problems, USSR Comput. Math. Math. Phys., 26(2) (1986), pp. 7–16.
16. Y. Saad, Iterative Methods for Sparse Linear Systems, 2nd ed., SIAM, Philadelphia, 2003.
17. R. C. Whaley, A. Petitet, and J. J. Dongarra, Automated empirical optimization of software and the ATLAS project, Parallel Computing, 27 (2001), pp 3–25. See also http://math-atlas.sourceforge.net