Parallel Implementation of a Frozen Flow Based Wavefront Reconstructor

Keith W. Kelly

Mathematics and Computer Science
Emory University
Atlanta, GA 30322, USA
keith@keithi.us

James G. Nagy

Mathematics and Computer Science
Emory University
Atlanta, GA 30322, USA
nagy@mathcs.emory.edu

ABSTRACT

Obtaining high resolution images of space objects from ground based telescopes is challenging, often requiring the use of a multi-frame blind deconvolution (MFBD) algorithm to remove blur caused by atmospheric turbulence. In order for an MFBD algorithm to be effective, it is necessary to obtain a good initial estimate of the point spread function (PSF), or, equivalently, the wavefront phase that defines the PSF. Although wavefront sensors work well in low turbulence situations, they are less effective in high turbulence, such as when imaging in daylight, or when imaging objects that are close to the Earth's horizon. One promising approach, which has been shown to work very well in high turbulence settings, uses a frozen flow assumption on the atmosphere to capture the inherent temporal correlations present in consecutive frames of wavefront data. Exploiting these correlations can lead to more accurate estimation of the wavefront phase, and the associated PSF, which leads to more effective MFBD algorithms. However, the computational problem requires solving very large scale least squares problems, which can be prohibitively expensive in situations when it is necessary to use a large number of frames or when there are a large number of turbulent layers in the atmosphere. In this paper we describe a parallel implementation that overcomes this constraint. The parallel implementation exploits sparse matrix computations, and uses the Trilinos package developed at Sandia National Laboratories.

1. INTRODUCTION

Because atmospheric turbulence distorts light as it travels through the atmosphere, images of space objects obtained using ground based telescopes are typically distorted by blurring. The amount of blurring depends on the severity of turbulence. Mathematically, the observed image, g, is related to the true object, f, through a convolution model

$$g(x,y) = \int_{\mathcal{R}^2} k(x-\xi, y-\eta) f(\xi,\eta) d\xi d\eta + e(x,y)\,, \tag{1}$$

where the *point spread function* (PSF), k, models the blurring operation, and e represents additional (unknown) noise. Using a Fourier optics model for atmospheric turbulence [14], the PSF can be expressed in terms of the wavefront phase error (deviation of the wavefront from planarity), ϕ, of the light that reaches the telescope mirror,

$$k(x,y) = \left| \mathcal{F}^{-1} \left\{ \mathcal{P}(x,y) e^{\iota\phi(x,y)} \right\} \right|^2 , \tag{2}$$

where $\iota = \sqrt{-1}$, $\mathcal{P}$ is a characteristic function that models the shape of the telescope aperture (e.g., a circle or annulus), and $\mathcal{F}^{-1}$ is the 2-dimensional inverse Fourier transform. In perfect seeing conditions, $\phi = 0$, and in good seeing conditions (low turbulence), ϕ is a smooth function.

In low turbulence situations, a wavefront sensor (WFS) can accurately estimate the wavefront phase, and either adaptive optics systems or computational post processing (deconvolution) can then be used to obtain a good estimate of the object, f. Specifically, a WFS measures estimates of wavefront gradients, $\frac{\partial \phi}{\partial x}$ and $\frac{\partial \phi}{\partial y}$, and ϕ is then obtained by solving a simple inverse problem [1, 2, 13]. However, the WFS can only obtain measurements on a coarse grid, and this under sampling can result in very poor estimates when ϕ is highly oscillatory (that is, in high turbulence situations) [4]. In this case, it may be possible to use an initial estimate of the wavefront phase in a *blind deconvolution* algorithm to jointly estimate ϕ (and, hence, the PSF) and the object f. Multiple observations can be used to make the highly underdetermined blind deconvolution problem more tractable; that is, $j = 1, 2, \ldots, m$ frames of observations

$$g_j(x,y) = \int_{\mathcal{R}^2} k_j(x-\xi, y-\eta) f(\xi,\eta) d\xi d\eta + e(x,y)\,, \quad k_j(x,y) = \left| \mathcal{F}^{-1} \left\{ \mathcal{P}(x,y) e^{\iota \phi_j(x,y)} \right\} \right|^2 ,$$

are obtained, and the aim is to compute approximations of ϕ_j and f. This problem is referred to as *multi-frame blind deconvolution* (MFBD).

It is important to emphasize the challenges faced in high turbulence situations: Higher turbulence requires better initial estimates of ϕ_j in MFBD algorithms. However, because it can only obtain data on a coarse grid, getting good initial estimates from a WFS becomes more difficult as the turbulence increases.

Jefferies and Hart [9] observed that if we can assume the atmosphere satisfies a frozen flow model (FFM), then multiple overlapping frames of WFS data can be effectively interpolated to improve the sampling resolution. It was shown in [4] that the FFM approach can be formulated as a large scale, sparse linear least squares problem, similar to what arises in digital super-resolution applications [5]. Although this previous work demonstrates that the FFM reconstruction algorithm can be solved efficiently in many situations, the large scale nature of the problem puts limitations on the number of atmospheric layers and the number of frames of data that can be incorporated into the model. In this paper, we describe a parallel implementation of the FFM reconstruction algorithm that allows us to overcome these limitations. The rest of this paper is organized as follows. In Section 2 we describe the FFM reconstruction algorithm, and in Section 3 we describe the high performance computing tools and framework used in the project. In Section 4 we provide some experiments on simulated data to illustrate the effectiveness of our approach. Some concluding remarks are given in Section 5.

2. WFS PHASE RECONSTRUCTION AND THE FROZEN FLOW MODEL

As mentioned in the introduction, when attempting to use MFBD algorithms, it is very important to obtain a good initial estimate of the wavefront phase. A WFS can be used to obtain estimates of $\frac{\partial \phi}{\partial x}$ and $\frac{\partial \phi}{\partial y}$, from which the phase, ϕ, can be reconstructed. However, because the WFS can only obtain estimates on a coarse grid, interpolation is needed to fill in missing data; this is illustrated in Figure 1, where the black dots (within the circular aperture) are computed by interpolating the WFS measurements (red dots).

In low turbulence situations, ϕ is smooth, and so interpolation between the coarse WFS measurements can be expected to provide fairly accurate estimates of the missing data. However, these interpolated values are likely to be poor estimates in high turbulence. The most reliable way to improve the wavefront approximation is to obtain better sampling from the WFS. This can be done, as Jefferies and Hart [9] observed, if multiple frames of WFS data can be collected quickly, and if the atmospheric turbulence satisfies a frozen flow model (that is, the turbulence can be modeled by a series of independent static layers, each moving across the telescope aperture with the prevailing wind at the altitude of the layer). While the FFM may not hold over long time scales, a number of studies have shown that it is a reasonable approximation for short periods [8, 12, 17]. Thus, by obtaining multiple frames of WFS data over a short

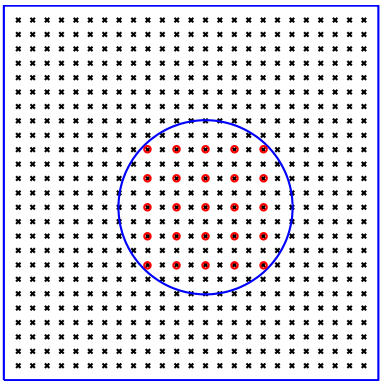

Figure 1: Simple cartoon illustration of a WFS. Here, the red dots denote points where actual estimates might be obtained by a WFS, and the black dots (within the circular aperture) are estimated from the red dots by interpolation.

time period, we can obtain overlapping samples of measured data, and thus improve the accuracy of interpolation. This is illustrated in Figure 2 for a single atmospheric layer. Notice that the improved sampling depends on the wind velocity and on how quickly the WFS data can be collected. The FFM reconstruction problem, then, is to compute wavefront gradient estimates on a fine grid (e.g., represented by black dots within the blue circular regions in Figure 2), given coarse grid measurements from the WFS, as well as information (e.g., wind vectors) about dominant atmospheric layers.

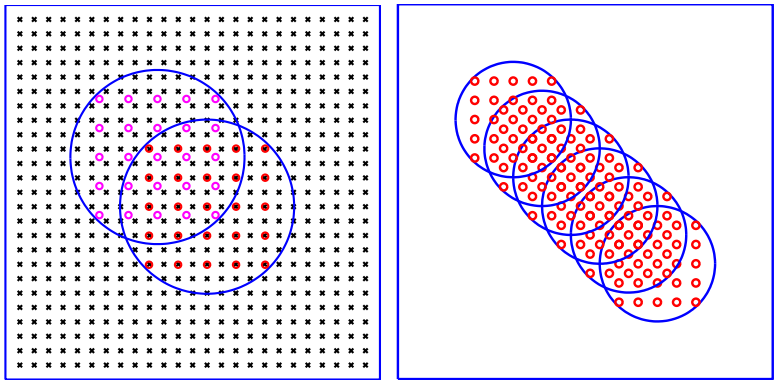

Figure 2: Simple cartoon illustration of overlapping WFS data. The left cartoon shows two overlapping frames, where the red and magenta dots denote data obtained by the WFS, and the black dots (within the circular aperture regions) are points that need to be interpolated. The right cartoon illustrates six overlapping frames of WFS data.

To describe the FFM reconstruction algorithm, first suppose that there is only one dominant layer of atmospheric turbulence. Let $\phi_x^{(j)}$ and $\phi_y^{(j)}$ be vectors that contain the WFS (low resolution) measured phase gradients for frame j, $j = 1, 2, \ldots, m$, where m is the number of frames of data collected by the WFS. Our goal is to reconstruct high resolution gradients within the composite region covered by all of the overlapping frames (that is, within the union of the blue circles in Figure 2); denote these composite high resolution gradients by ϕ_x and ϕ_y. To relate the low resolution measured data ($\phi_x^{(j)}$ and $\phi_y^{(j)}$) to the composite high resolution grid information we want to reconstruct (ϕ_x and ϕ_y), we need the following mathematical operations:

- Move the composite high resolution grid so that the information corresponding to frame j matches with a windowed region corresponding to the pupil aperture. Because the FFM assumes that changes in the gradients between frames are rigid movements, we can represent these by affine transformations (the affine transformation used for each frame can be determined from the wind velocity information). This affine transformation can be represented by a matrix operation, $A^{(j)}\phi_x$ and $A^{(j)}\phi_y$.
- Punch out the composite region corresponding to the pupil aperture window. This can also be represented as a matrix operation, $WA^{(j)}\phi_x$ and $WA^{(j)}\phi_y$.
- Finally, restrict (or down sample) the high resolution grid to the low resolution WFS measured data: $RWA^{(j)}\phi_x$ and $RWA^{(j)}\phi_y$.

That is, for each frame of data, we have

$$\phi_x^{(j)} = RWA^{(j)}\phi_x \quad \text{and} \quad \phi_y^{(j)} = RWA^{(j)}\phi_y$$

If we have m total frames of data, then we can expand this to

$$\phi_x^{(1:m)} = (I \otimes RW)A_1\phi_x \quad \text{and} \quad \phi_y^{(1:m)} = (I \otimes RW)A_1\phi_y$$

where I is an $m \times m$ identity matrix, $\otimes$ represents the Kronecker product and

$$\phi_x^{(1:m)} = \begin{bmatrix} \phi_x^{(1)} \\ \phi_x^{(2)} \\ \vdots \\ \phi_x^{(m)} \end{bmatrix}, \quad \phi_y^{(1:m)} = \begin{bmatrix} \phi_y^{(1)} \\ \phi_y^{(2)} \\ \vdots \\ \phi_y^{(m)} \end{bmatrix}, \quad A_1 = \begin{bmatrix} A^{(1)} \\ A^{(2)} \\ \vdots \\ A^{(m)} \end{bmatrix}.$$

The subscript on A_1 is used to emphasize that, so far, the atmospheric model consists of a single layer of turbulence.

A more realistic atmospheric model is to use a finite number of discrete layers, where each layer is characterized by a turbulence strength that is approximately constant within the layer. The effect of light propagation through the entire turbulence region, assuming the index of refractive fluctuations for each layer are statistically independent, is the sum of the effects through each layer [14]. Using this multilayer atmospheric model, the FFM reconstruction problem can be written as

$$\begin{cases} \phi_x^{(1:m)} = \begin{bmatrix} (I \otimes RW)A_1 & (I \otimes RW)A_2 & \cdots & (I \otimes RW)A_L \end{bmatrix} \begin{bmatrix} \phi_{x,1} \\ \phi_{x,2} \\ \vdots \\ \phi_{x,L} \end{bmatrix} \\ \\ \phi_y^{(1:m)} = \begin{bmatrix} (I \otimes RW)A_1 & (I \otimes RW)A_2 & \cdots & (I \otimes RW)A_L \end{bmatrix} \begin{bmatrix} \phi_{y,1} \\ \phi_{y,2} \\ \vdots \\ \phi_{y,L} \end{bmatrix} \end{cases} \tag{3}$$

where $\phi_{x,\ell}$ and $\phi_{y,\ell}$ are the composite, high resolution gradients at layer ℓ, and A_ℓ is motion matrix at layer ℓ with $\ell = 1, 2, \ldots, L$. If we define $\widehat{R} = I \otimes R$, $\widehat{W} = I \otimes W$, $A = [A_1 \ A_2 \ \cdots \ A_L]$, and

$$\phi_{x,(1:L)} = \begin{bmatrix} \phi_{x,1} \\ \phi_{x,2} \\ \vdots \\ \phi_{x,L} \end{bmatrix}, \quad \phi_{y,(1:L)} = \begin{bmatrix} \phi_{y,1} \\ \phi_{y,2} \\ \vdots \\ \phi_{y,L} \end{bmatrix}, \quad A_\ell = \begin{bmatrix} A_\ell^{(1)} \\ A_\ell^{(2)} \\ \vdots \\ A_\ell^{(m)} \end{bmatrix},$$

then we can write the multilayer, multiframe model much more compactly as

$$\begin{cases} \phi_x^{(1:m)} = \widehat{R}\,\widehat{W}A\phi_{x,(1:L)} \\ \phi_y^{(1:m)} = \widehat{R}\,\widehat{W}A\phi_{y,(1:L)} \end{cases}$$

The FFM reconstruction, then, requires computing the high resolution composite gradients $\phi_{x,(1:L)}$ and $\phi_{y,(1:L)}$, given the low resolution WFS gradients $\phi_x^{(1:m)}$ and $\phi_y^{(1:m)}$ and matrices $\widehat{R}$, $\widehat{W}$, and A. Because the problem is underdetermined, we incorporate (as is done in similar digital resolution algorithms) Tikhonov regularization, and solve the damped least squares problems

$$\begin{cases} \min\limits_{\phi_{x,(1:L)}} \left\| \phi_x^{(1:m)} - \widehat{R}\,\widehat{W}A\phi_{x,(1:L)} \right\|_2^2 + \lambda^2 \|\phi_{x,(1:L)}\|_2^2 \\ \min\limits_{\phi_{y,(1:L)}} \left\| \phi_y^{(1:m)} - \widehat{R}\,\widehat{W}A\phi_{y,(1:L)} \right\|_2^2 + \lambda^2 \|\phi_{y,(1:L)}\|_2^2 \end{cases} \tag{4}$$

A more detailed description of this least squares formulation of the FFM reconstruction problem can be found in [4]. The matrices in the least squares problems (4) are very large, and, in general, they do not have any obvious structure that can be exploited in the computations; for example, there is no Toeplitz structure that can take advantage of fast Fourier transforms to improve computational efficiency. However, the matrices are sparse, and this can be exploited in iterative algorithms, such as the conjugate gradient method for least squares problems, LSQR [11, 15].

3. COMPUTATIONAL APPROACH

Implementation of the FFM reconstruction problem can be computationally very expensive, especially for large numbers of data frames and atmospheric layers. Therefore, in this section, we describe a parallel implementation. To maximize our effort to transition the methods to practical use, we are interested in producing software that is easy to use, stable, well-documented, and actively supported. In addition, the software must be portable to a variety of computer platforms and have acceptable licensing agreements. To achieve these goals we are making use of the *Trilinos* project [7] developed at Sandia National Laboratories. Trilinos provides a variety of core mathematical software (such as BLAS and LAPACK) for parallel architectures that have been designed using high quality software engineering practices. The term "Trilinos" can be roughly translated from Greek to mean "a string of pearls", referring to the fact that each individual package within Trilinos is a gem in its own right, but when combined, they create a mathematical software library that is worth more than the sum of its parts [10]. Trilinos is free software, which can be redistributed under the terms of the GNU Lesser General Public License as published by the Free Software Foundation. In addition, Trilinos provides a streamlined process and set of tools for development of new algorithmic implementations, and promotes interoperability of independently developed software. Using Trilinos has facilitated implementation of the algorithms developed in this work, for both large distributed memory parallel computers, as well as for less expensive modern multi-core commodity machines. Although Trilinos contains packages that can be used to solve very large scale linear and nonlinear problems, there is nothing currently available specifically for imaging applications. However, it is an ideal environment for our purposes; building off of the Trilinos native matrix and vector objects, as well as some of the core Trilinos packages, we have been able to develop an efficient, parallel method for the FFM reconstruction.

The foundation for Trilinos is the Petra object model, which describes objects such as matrices and vectors, and was designed with the ability to do parallel data redistribution efficiently and easily. Although the Petra object model is abstract, there are concrete implementations, such as Epetra. One particular object model in Epetra that is very beneficial to our application is the `Epetra_CrsMatrix`, which constructs a sparse matrix in compressed row storage format [6]. In addition, the `EpetraExt` package provides tools, such as sparse matrix-matrix multiplication, which are nontrivial to implement efficiently. In our application, each of the matrices $\widehat{R}$, $\widehat{W}$ and A can be constructed as an `Epetra_CrsMatrix`, and,

in addition, we can form the explicit product $\widehat{R}\,\widehat{W}A$. Note that in the FFM reconstruction, all we need is the product $\widehat{R}\,\widehat{W}A$, and this observation is important because the product is a much more sparse than A. For example, on a test problem with 25 frames of data and 3 atmospheric layers, $\widehat{R}$ has 409,600 nonzeros, $\widehat{W}$ has 301,450 nonzeros, and A has 17,249,503 nonzeros. However, $\widehat{R}\,\widehat{W}A$ has only 1,412,598 nonzeros, just over 8% of the nonzeros than are in A. Note that the most expensive computation at each LSQR iteration, which we use to solve the least squares problems (4), is the computation of a matrix-vector and a matrix transpose-vector product with $\widehat{R}\,\widehat{W}A$. Therefore, this reduction in storage can have a significant impact on the overall speed of the FFM reconstruction. However, it should be noted that if our FFM reconstruction is combined with a phase reconstructor and an MFBD algorithm, then it is necessary to also work with the individual matrices $\widehat{R}$, $\widehat{W}$ and A.

The next important Trilinos package we use is Teuchos, which is a "toolkit" that provides wrappers to BLAS and LAPACK, smart pointers, parameter lists, XML parsers, etc. Of particular importance is the `Teuchos:RCP` smart pointer, which combines the concepts of smart pointers and reference counting to build a low-overhead but effective tool for simplifying dynamic memory management in C++ [3].

Finally, the Belos package in Trilinos provides a powerful framework for iterative linear solvers, such as LSQR. Within Belos, it is possible to define a `Belos::LinearProblem` object class, which is composed of an `Epetra_CrsMatrix` defining $\widehat{R}\,\widehat{W}A$ and an `Epetra_MultiVector` representing $\phi_x^{(1:m)}$ and $\phi_x^{(1:m)}$. With the linear problem defined, then a call is made to the LSQR solution manager, `Belos::LSQRSolMgr`, which takes as input the linear problem and any solver parameters, such as the residual stopping tolerance. We remark that the LSQR solver provided in Belos has Tikhonov regularization built into the algorithm; we simply need to specify the regularization parameter λ in the solver parameter list that is passed to the LSQR solution manager.

4. NUMERICAL EXPERIMENTS

Previous publications have shown that the FFM reconstructions can produce substantially better phase reconstructions than current approaches; see, for example, [4, 9]. We do not repeat those results here. Instead, we report on the performance of our parallel implementation on a variety of test problems. Specifically, we used a realistic model of atmospheric turbulence, described in [16], to generate a global wavefront phase, assuming that the diameter of the telescope is 3.7 m, the light wavelength is 0.744×10^{-6}m, and with a propagation distance of 25 km. We varied the number of atmospheric layers and the number of frames used in the FFM reconstruction. In each case, wind velocities were generated randomly.

The computations for the experiments reported in this section were performed on a shared memory Linux machine with 8 Dual-Core AMD Opteron 8220 processors for a total of 16 processor cores and 64GB of RAM. Figure 3 shows the number of iterations of the LSQR algorithm required to reconstruct the horizontal and vertical gradients in our test problems.

Figure 3: Iterations for Gradient Reconstruction

		Number of Frames			
		10	25	50	75
Number of Layers	1	845	885	737	862
	2	563	803	730	760
	3	372	798	613	520

(a) Number of iterations required for the horizontal gradient reconstruction in our test problems.

		Number of Frames			
		10	25	50	75
Number of Layers	1	858	884	740	852
	2	624	806	730	756
	3	375	803	618	522

(b) Number of iterations required for the vertical gradient reconstruction in our test problems.

Figure 4 shows how our FFM reconstruction algorithm scales for a fixed number of processes with increasing problem size. Figure 5 shows scaling results for a fixed problem size with increasing number of processes. These results show the good performance of our parallel implementation, but we remark that although these results are promising, it is necessary to do further testing on a variety of machines and parallel architectures.

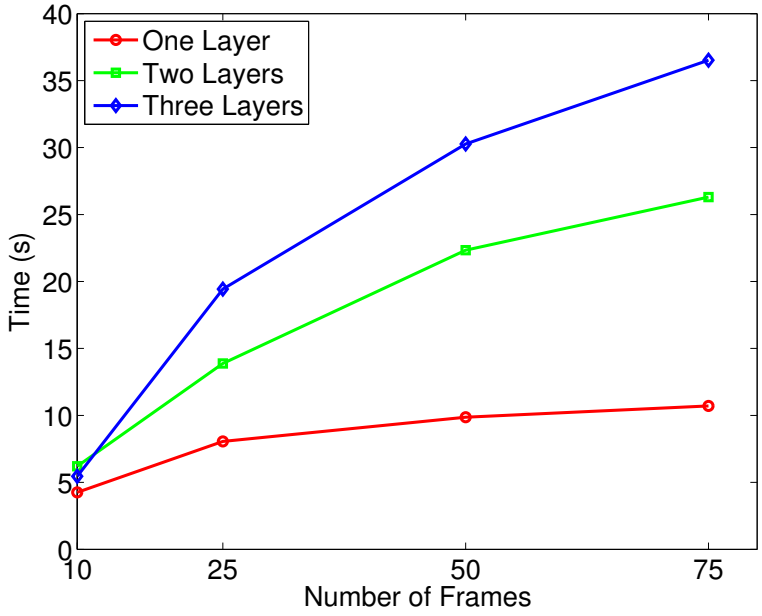

Figure 4: This figure shows timings of the parallel FFM reconstruction algorithm as the number of frames increases, using 8 MPI processes on a shared memory machine.

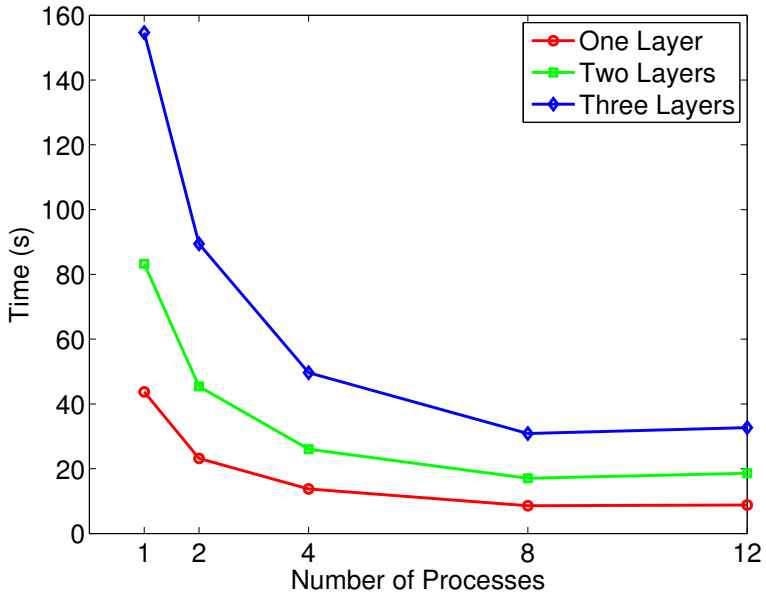

Figure 5: This figure shows timings for the parallel FFM reconstruction algorithm using a fixed problem size of 50 frames, but varying the number of processes for the computation.

5. CONCLUDING REMARKS

The frozen flow method allows for accurate estimate of the wavefront phase for use in multiframe blind deconvolution. However, with large numbers of frames and atmospheric layers, the FFM reconstruction algorithm can be very computationally expensive. In this paper, we have shown that by exploiting parallel architectures and using parallel libraries like Trilinos we can greatly reduce the amount of time necessary for the reconstruction. In addition, the work described here can be incorporated into a larger MFBD software package; this work is currently ongoing.

Acknowledgments

This work has been supported by the Air Force Research Laboratory under contract FA9451-12-C-0004 to HartSCI LLC. The work of J. Nagy is also supported by the Air Force Office of Scientific Research grant FA9550-12-1-0084.

REFERENCES

1. J. M. Bardsley. Wavefront reconstruction methods for adaptive optics systems on ground-based telescopes. *SIAM J. Matrix Anal. Appl.*, 30:67–83, 2008.
2. J. M. Bardsley, S. M. Knepper, and J. G. Nagy. Structured linear algebra problems in adaptive optics. *Advances in Comp. Math.*, pages DOI:10.1007/s10444–011–9172–9, 2011.
3. R. A. Bartlett. Teuchos::RCP Beginner's Guide. Sandia Report SAND2004-3268, 2010.
4. Q. Chu, S. Jefferies, and J. G. Nagy. Iterative wavefront reconstruction for astronomical imaging. *SIAM J. Sci. Comput..*, page to appear, 2013.
5. J. M. Chung, E. Haber, and J. G. Nagy. Numerical methods for coupled super-resolution. *Inverse Problems*, 22:1261–1272, 2006.
6. T. A. Davis. *Direct Methods for Sparse Linear Systems.* SIAM, Philadelphia, PA, 2006.
7. M. Heroux et al. The Trilinos Project. http://trilinos.sandia.gov/index.html.
8. E. Gendron and P. Léna. Single layer atmospheric turbulence demonstrated by adaptive optics observations. *Astrophysics and Space Science*, 239:221–228, 1996.
9. S. M. Jefferies and M. Hart. Deconvolution from wave front sensing using the frozen flow hypothesis. *Opt. Express*, 19:1975–1984, 2011.
10. S. M. Knepper. *Large Scale Inverse Problems in Imaging: Two Case Studeis.* PhD thesis, Emory University, 2011.
11. C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Soft.*, 8:43–71, 1982.
12. L. A. Poyneer, M. van Dam, and J. P. Véran. Experimental verification of the frozen flow atmospheric turbulence assumption with use of astronomical adaptive optics telemtry. *J. Opt. Soc. Am. A*, 26:833–846, 2009.
13. H. Ren, R. Dekany, and M. Britton. Large-scale wave-front reconstruction for adaptive optics systems by use of a recursive filtering algorithm. *Applied Optics*, 44(13):2626–2637, 2005.
14. M. C. Roggemann and B. Welsh. *Imaging Through Turbulence.* CRC Press, Boca Raton, FL, 1996.
15. Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS Publishing Company, Boston, 1996.
16. J. D. Schmidt. *Numerical Simulation of Optical Wave Propagation with Examples in MATLAB.* SPIE Press, Bellingham, WA, US, 2010.
17. M. Schöck and E. J. Spillar. Method for a quantitative investigation of the frozen flow hypothesis. *J. Opt. Soc. Am. A*, 17:1650–1658, 2000.