McMaster University

Advanced Optimization Laboratory

Title:

A first-order framework for inverse imaging problems

Author:

Christopher Kumar Anand

AdvOL-Report No. 2007/1

January, 2007
Hamilton, Ontario, Canada

A FIRST-ORDER FRAMEWORK FOR INVERSE IMAGING PROBLEMS

CHRISTOPHER KUMAR ANAND

Abstract. We argue that some inverse problems arising in imaging can be efficiently treated using only single-precision (or other reduced-precision) arithmetic, using a combination of old ideas (first-order methods, polynomial preconditioners), and new ones (bilateral filtering, total variation). Using single precision, and having structures which parallelize in the ways needed to take advantage of low-cost/high-performance multi-core/SIMD architectures, this framework is especially suited to embedded image reconstruction applications like medical imaging. We show with a simulated magnetic resonance imaging problem that this method can be numerically effective. Since the convergence/error analysis is particularly simple for pure quadratic objectives, this approach can also be used in embedded environments with fixed computation budgets, or certification requirements. Simple analysis for the quadratic case also serves as a basis for the analysis of nonlinear problems solved via a sequence of quadratic approximations. We include one example of a nonlinear, nonquadratic penalty function.

1. Introduction

In this paper, we propose a method of solving inverse problems in imaging which are characterized by two challenges: large optimization problems and tight budgets (for cost, power consumption, and solution time). Significant recent progress in mathematical methods such as Total Variation regularization [ROF92] and anisotropic diffusion of image processing demonstrates that image quality can be significantly improved by incorporating novel regularization and iterative strategies into the inverse problem models, but at considerable computational cost. Bilateral filtering [TM98], on the other hand, provides a remarkable level of noise reduction in a single step and offers efficient computation. Our method arose in answer to the question: Can the efficiency of bilateral filtering be brought to inverse imaging problems?

These algorithm developments coincide with a period of rapid change in hardware architecture. For example, the recently-released first-generation Cell BE delivers 25GFlops (double precision) or 200GFlops (single precision) on a single chip. Unfortunately, to reach higher levels of performance, the model of computation has changed in ways which constrain the types of operations that can be performed efficiently: multiple cores, loss of uniform memory access from different cores, single-instruction multiple-data (SIMD), reduction in precision (from double to single on the Cell or lower on graphics processors), and constrained program and working data sizes. Our method was developed to fit into this framework and take advantage of peak performance on these processors.

Date: January 25, 2007.

In this work, we explain that for a large class of inverse problems, including most problems referred to as image estimation or image reconstruction, first-order methods based on Neumann series and inspired by Bilateral Filtering can

(1) stably produce good results for linear and nonlinear models, entirely using single (or lower) precision;
(2) take full advantage of newer hardware (effectively using SIMD and small working-memory footprints);
(3) provide residual error estimates for any finite number of iterations.

As with any first-order method, we do not obtain superlinear convergence, but we do observe good initial convergence and, provided that sufficient model information is available, can guarantee stability for the method, *i.e.* we would never see divergence as is possible with the conjugate gradient method. This is a good tradeoff in cases with high levels of measurement noise, absolute computation time limits, and completely unsupervised computation.

Recently, Dongarra *et al.* [KD06] have shown that on architectures with slower double-precision performance, basic linear algebra can be significantly accelerated by using iterative refinement and mixed single/double precision. Our approach is potentially more general in that it uses only single precision, but more restricted in depending on the specific structure of inverse imaging problems. It would be interesting to compare the present approach with one based on iterative refinement in the future.

This framework applies to any linear inverse problem

(1) in which the model variables are regular samplings of a vector-space-valued function (*e.g.*, a multi-spectral image), and
(2) for which there is an efficiently-computable (but not necessarily sparse) forward problem.

It applies to some nonlinear problems, and we demonstrate the use of a nonlinear penalty, but provide no proofs of convergence in the general nonlinear case. We use a specific numerical test problem from Magnetic Resonance Imaging (MRI), SENSitivity Encoding (SENSE), in a variant which does admit a direct inversion, so that we have a baseline for comparison. Similarly, we will focus our discussion on efficient implementation for the Cell processor in particular, since it is the target of on-going implementation work, and represents a direction we expect processor design to follow.

Any medical imaging problem (imaging using x- and gamma-rays, ultrasound), volumetric radar and sonar, seismic imaging, and remote sensing can be adapted to this framework. We expect the resulting algorithms to be competitive in cases involving large data/model sizes with significant measurement noise.

2. Previous Work

Inverse problems are being solved in many fields of application, sometimes without being labeled as inverse problems. Without developing a survey of the related literature, we will try to put our method into context, making it clear that several other approaches to these problems exist.

In this section, we sketch the most important alternate methods, and the methods most related to this work. We have tried to choose references which are either

closely related to the inverse problems which motivate us, or which synthesize two or more of the related fields.

2.1. **Optimization with and without derivatives.** All methods can be classified by their use of model derivatives (no use, uses gradients, uses Hessians).

No derivatives. Methods not using gradients include filtering methods applied to images or other data with positional metadata (*i.e.* meshes), and simple iterative methods including Projection onto Convex Sets (POCS). These methods are generally more stable, but with the worst asymptotic convergence (when they are used iteratively and do converge). Filtering is very often applied once, which makes it a fast method. Bilateral Filtering, which we will discuss further below is very effective in many domains. POCS can be very effective when the geometry and computational complexity of large systems of linear constraints (which may come from linearizing nonlinear constraints) is well understood. It is an everyday tool (see [MSP86]) and a source of innovation in our MRI test problem (see [SKPJ04]), and an actively-applied method in other areas (for example [AM06]). It is most successful when the linear constraints correspond to enforced sparsity of representation in different bases, typically corresponding to time and frequency domains. The penalty functions we introduce based on a simplex of possible pixel values is related to these methods.

First derivatives. Steepest descent iteratively uses the negative gradient as a search direction for a one-dimensional search. This method is as stable as the gradient (usually very stable), but has poor theoretical and measured complexity on large, interesting problems. One important class of first-derivative methods is based on diffusion equations with spatially-varying (anisotropic) coefficients, proposed by [PM90]. Although such methods may use higher derivatives in constructing the DE, the steps are gradient steps. The only method of improving such methods is to improve the search direction. Our most important penalty function is optimized to produce good search directions.

Second derivatives. Methods which use second derivatives fall into several subcategories, including Newton's method, Neumann series, and Krylov subspace methods (which includes the Conjugate Gradient (CG) method [HS52] also applied previously to our MRI test problem [PWBB01]). Imaging problems are, in general and in the particular cases of interest, too large for the application of Newton's method. For large problems with well-behaved spectra, CG offers rapid convergence (both theoretical and observed) with very low computational complexity. Unfortunately, the calculation of residuals in CG introduces numerical instability which is difficult to predict and is significantly amplified by doing computations at single precision and/or processing data with significant noise components. Another approach to solving large linear systems with fast matrix-vector multiplication, but no fast inversion is to use a (truncated) series,

$$(I+E)^{-1} = I - E + \frac{1}{2}E^2 - \cdots, \tag{1}$$

or another polynomial approximation to the inverse. This approach is used in conjunction with other methods, notably as a method of preconditioning CG to improve convergence (see [JMP83]). We are proposing to use this approach by itself, not as a method of preconditioning another method.

This is not a comprehensive list of relevant optimization methods. We have also not considered constrained optimization, sophisticated uses of function spaces (*e.g.*

basis pursuit in redundant wavelet bases, or spline bases [LP06]), filter search and L-curve analysis.

2.2. **Total Variation.** Total variation as a regularizer for inverse problems (image recovery) was introduced in [ROF92]. The basic idea is to use

$$\int_\Omega \|\nabla f\|, \tag{2}$$

the L^1 norm of the gradient of the model function, as a regularizer. Using the L^1 norm is motivated both by heuristics (penalizing oscillations versus steep slopes) and robust statistics, see [Hub81], which seek to reduce the large effect outliers can have on estimates based on limited samples (image pixels always have limited numbers of neighbors). Total Variation was quickly shown to produce visually and numerically superior results to standard problems in image processing, see [Cha04], and image reconstruction, see [CCT04], including constrained optimization [BCRS03].

The main difficulty with TV was the lack of smoothness in the penalty function, which can be dealt with in various ways, notably by introducing an auxiliary optimization problem and using a custom primal-dual [CGM99] or second-order cone [GY05] approach. These latter approaches share the robustness and efficiency of interior point methods. Unfortunately, they involve solving linear systems which become ill-conditioned as the method converges, even in double precision.

The connection with robust statistics has played a more direct role in the development of at least one algorithm, see [HHK$^+$03].

2.3. **Markov Processes.** Another way of obtaining the similar models is to explicitly model the pixel values as random fields, using Markov processes to capture the dependence of pixels on their neighbors. See [NMDI94], [ZBS01] for examples close to our problem domain. In this context, it is natural to introduce distributions from Robust statistics.

2.4. **Bilateral Filtering.** Although anisotropic diffusion (AD) was quite successful, it is inherently expensive (depending as it does on gradient descent). Bilateral filtering is a one-step noise-reduction method which achieves similar results. For an introduction which makes explicit the connection with AD, see [Bar02]. The idea is to combine spatial convolution filtering, with range filtering.

Let $\Omega \subset \mathbb{Z}^N$ be a multi-dimensional lattice of points, V a real vector space and $f : \Omega -> V$ a (noisy) image. Convolution with a kernel $c : \mathbb{Z}^N -> \mathbb{R}$,

$$\hat{f}(x) = \sum_{y \in \mathbb{R} \setminus \{x\}} c(y-x) f(y), \tag{3}$$

is the simplest method of filtering noise. It works when the noise and true image have different spectra (usually the noise is white, with uniform high- and low-frequency components, but the image has very small high-frequency components).

Bilateral filtering introduces a range kernel function $s : V \to V$,

$$\hat{f}(x) = \sum_{y \in \mathbb{R} \setminus \{x\}} c(y-x) s(f(y) - f(x)), \tag{4}$$

which modulates the weighting of neighboring pixel values. In the original and later application papers, this is seen to reduce edge blurring in images with sharp edges, because it reduces the influence of neighboring pixels with very different values on

each other. Both c and s are usually taken to be Gaussian kernels, with the c being truncated at some finite width to reduce computation, and s chosen to match the expected noise statistics in f.

Although Bilateral Filtering works well as a single-step filtering method, the concept can be used in an iterative derivative-free method, for example in AD [Bar05]. The cycle of relationships is closed in the analysis, [BSMH97], of the link between robust statistics and AD.

2.5. **Exploiting Commodity/Lower-Precision Hardware in Scientific Computation.** In another direction, the community of researchers exploring the limits of computation possible on graphics processors has made significant progress in translating scientific algorithms to the limited precision and restricted computational model of current graphics processors. This work offers an alternative to our proposed method and significant progress has been made, including recent work implementing interior point linear programming using OpenGL [JO06]. For algorithms which do not perform well in single precision, it is possible to use single-precision calculations and iterative refinement to produce higher-precision results, as was done for very high-precision linear algebra in [GZ03] and for accelerating Cell computations in [KD06].

Note: For numerical analysts who may not be familiar with the level and kind of parallelism available in current commodity architectures, we examine the case of the first-generation Cell [KDH$^+$05]. This processor contains 8 general-purpose processor cores (Synergistic Processor Units, SPUs), each capable of dispatching a single SIMD floating-point operation per cycle. Each SIMD operation operates on 128 bit operands, which can be operated on as a short vector of four single-precision floating-point numbers. Although it is possible to reorder the constituents of operands and synthesize conditional execution, peak floating-point performance is only available when the four floating-point elements in each operand are operated on in the same manner. For example, adding two vectors can easily be arranged to function this way, while Cholesky decomposition of a single tridiagonal block will operate at near one quarter the peak rate. This can be remedied if four blocks of the same size are processed in parallel, with the data stored in memory in an interleaved fashion. Again, using the Cholesky decomposition example, data dependencies between instructions mean that four decompositions executed in parallel will suffer from pipeline stalls equal to the six cycles of latency of a floating-point operation. To achieve peak performance, 6×4 decompositions must be executed in parallel, using software-pipelined loops. To use all eight SPUs, we need 8×6 decompositions to execute in parallel. Since the SPUs can only efficiently address 256MB of local storage, some method of buffering data in and out of the local storage is required. Double buffering entire blocks is the simplest scheme, requiring an additional factor of two. In total, 384 decompositions must be in-flight in parallel to achieve peak performance. The exact amount of parallelism required to achieve peak performance depends on both the high-level and low-level data dependencies. For example, dense Cholesky decompositions can be parallelized internally, using SIMD features at near peak performance, and long vectors may be broken into blocks and pipelined at the high level. But hundred-way parallelism will still be required for a single Cell, and thousand-way for a small cluster.

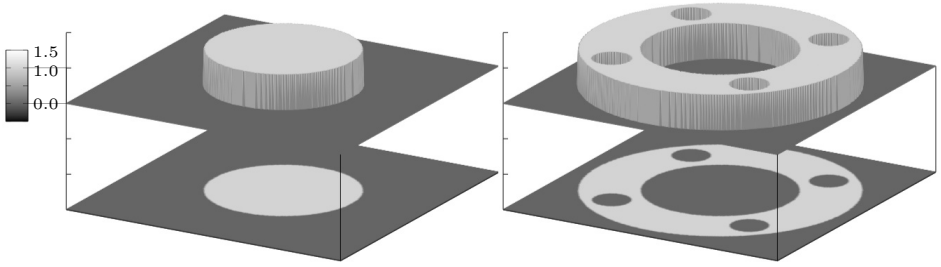

FIGURE 1. Real *(left)* and imaginary *(right)* parts of the ideal density. The same vertical and gray scales are used in all comparable plots.

2.6. **Novelty.** Our contribution to the solution of inverse imaging problems consists of

- the decomposition of the Hessian into block diagonal and other blocks for use in Neumann series or polynomial approximations,
- penalty functions designed to mimic bilateral filters,
- the optimization of spatial kernels to produce desired gradient directions from penalty functions.

Although we could not find the repeated application of Neumann series, single applications have been used in other ways in image reconstruction problems, see [WNC01], and repetition of such application was probably considered previously.

3. Example Problem

We will use a computationally simple inverse problem for the purposes of exposition and numerical evaluation: MR SENSE (Sensitivity Encoding) with regular two times undersampling of signals in the frequency domain. In this problem, we want to determine a complex tissue density function

$$\rho : \Omega \to \mathbb{C} \tag{5}$$

from measurements

$$\mu_1, \mu_2, \mu_3, \mu_4 : \Omega/2 \to \mathbb{C}, \tag{6}$$

in which we take $\Omega = \{0, 1, \ldots, 255\} \times \{0, 1, \ldots, 255\}$ and $\Omega/2 = \{0, 1, \ldots, 127\} \times \{0, 1, \ldots, 255\}$. The real and imaginary parts of our numerical test object ρ are given in Fig. 1. The measurements are modeled by

$$\mu_{m;i,j} = S_{m;i,j}\rho_{i,j} + S_{m;i+128,j}\rho_{i,j} + \epsilon_{m;i,j}, \tag{7}$$

where the $\epsilon_{m;i,j}$ are identically normally distributed, zero mean, independent measurement errors. Physically, the measurements are reconstructed MR images which are undersampled in one direction so that the reconstructed images alias, with the top/bottom of the image also appearing in the middle. MR measurements are collected using antennae which modulate the tissue density by their spatial sensitivity, $S_m : \Omega \to \mathbb{C}$. SENSE works well when the antennae are designed to produce linearly-independent sensitivities.

This is a particular example of SENSE, in which a regular undersampling pattern in frequency space produces (after reordering) a block diagonal forward problem

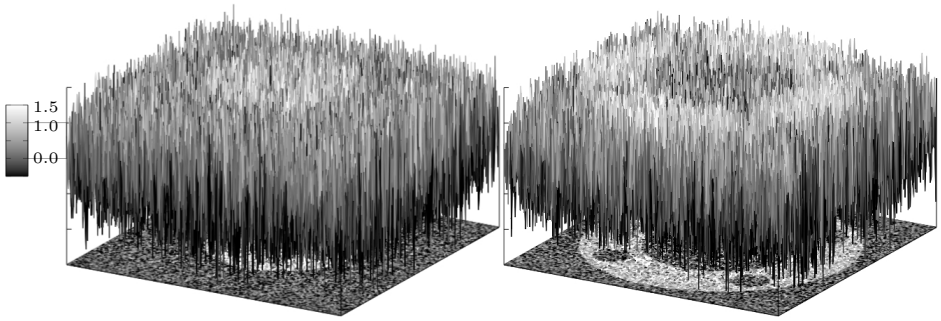

FIGURE 2. Solution of the direct problem. The fixed gray and vertical scale reduces the visibility of the noise.

with 4×8 (real) dense blocks. We take advantage of this structure to solve the inverse problem directly as a baseline solution, see Fig. 2.

The usual measure for noise in MRI (Signal to Noise Ratio, SNR) is calculated as the ratio of the largest expected pixel value to the standard deviation of the measured error. In our case, the true image has pixel values 1, i or 0, so the SNR is the inverse of the standard deviation of the error, which we have chosen to be 2, when measured over the pixels with true value of 1, giving an SNR of 1/2. (More noise than signal.)

The conditioning of the forward problem depends on the sensitivities, and is referred to as the G-factor in the MR literature. For typical medical imaging, all of the antennae are constrained to lie outside the patient, and the sensitivities will all be weak and flat in the middle of the patient. These pixels will be associated with the blocks with the highest condition numbers.

This example problem is apt because (i) in many imaging problems, increased signal quality comes with increased collection time (and cost); imaging is never fast enough, so producing images from noisier data is welcome; and (ii) irregular sampling patterns (which arise naturally in many efficient imaging situations, *e.g.* [PWBB01, ACK06]) lead to problems with no direct inverses.

Using a regularly-sampled representative for this class of problem simplifies the exposition of the problem and implementation of the algorithm. It also makes available a direct solution (without regularization) to illustrate the relative magnitude of the signal and noise, which would be harder to gauge in the general case.

4. ALGORITHM

The principle is to formulate one or a series of objective functions ϕ_i, including a fit-to-data term and a penalty term(s), and to apply a polynomial approximation to a Newton step, solving

$$(\mathcal{H}(\phi_i) + \alpha I)\,\Delta x_{i+1} = -\nabla\phi_i, \tag{8}$$

where αI is a positive multiple of the identity. The polynomial is an approximation of $1/(1+x)$ on an interval containing the spectrum of BA^{-1}, where

$$\mathcal{H}(\phi_i) + \alpha I = A + B, \tag{9}$$

is a decomposition into A, block diagonal with banded blocks, and B general. The decomposition is chosen to minimize the spectral radius of B, *e.g.* by making B

zero on the nonzero bands of A. The blocks correspond to rows or columns in the image, depending on the ordering (row- or column-major order). We will assume row-major organization, unless another organization is specifically mentioned. We make A block diagonal with banded blocks because in the Hessian, this corresponds to objective/penalty functions which depend on the relationships between close neighbors in the row direction. An objective which depends on close neighbors in all directions will have a block banded Hessian with banded blocks, in which most blocks are zero and nonzero blocks are only nonzero on their central sub/super diagonals.

We choose this sparse structure for A because each of the blocks (which correspond to rows) can be factored in parallel. The Hessian, and hence B, may be dense, as long as matrix-vector-products can be efficiently computed. All of the Hessians for penalty functions introduced in this paper can be effectively parallelized. The problem of fast computation of the fit-to-data term (gradient and Hessian) is not unique to this method, and for most important inverse problems fast implementations already exist.

In an N^2 image, if A has $m \ll N$ nonzero bands, the Cholesky decomposition can be done in parallel for each image row/A block in $\mathcal{O}(mN \otimes N)$ operations using $O(mN)$ working space, to form

$$A = LL^T, \tag{10}$$

where we use the notation $\mathcal{O}(x \otimes y)$ to mean $\mathcal{O}(x)$ operations per parallel computation, with y such computations for the whole image. In the case of an N^3 image, $\mathcal{O}(mN \otimes N^2)$ operations would be required.

The same block diagonal structure applies to L. Formally,

$$\begin{aligned}(A+B)^{-1} =& (LL^T + B)^{-1} \\ =& L^{T^{-1}}(\mathbb{I} + L^{-1}BL^{T^{-1}})^{-1}L^{-1}\end{aligned}$$

(replace with Taylor series)

$$\begin{aligned}=& (L^{T^{-1}}L^{-1}) - (L^{T^{-1}}L^{-1}BL^{T^{-1}}L^{-1}) \\ &+ (L^{T^{-1}}L^{-1}BL^{T^{-1}}L^{-1}BL^{T^{-1}}L^{-1}) - ...\end{aligned}$$

A truncated Taylor series provides a fast method of finding approximate solutions to $(A+B)^{-1}(\Delta x) = -\nabla\phi_i$, using the following steps:

(1) calculate $L^{-1}(-\nabla\phi_i)$ by back-solving in $\mathcal{O}(nN)$ operations
(2) calculate $L^{T^{-1}}L^{-1}(-\nabla\phi_i)$ by back-solving in $\mathcal{O}(nN)$ more operations
(3) save result
(4) calculate $BL^{T^{-1}}L^{-1}(-\nabla\phi_i)$ using the fast computation for B
(5) calculate $L^{-1}BL^{T^{-1}}L^{-1}(-\nabla\phi_i)$ by back-solving in $\mathcal{O}(nN)$ more operations
(6) calculate $L^{T^{-1}}L^{-1}BL^{T^{-1}}L^{-1}(-\nabla\phi_i)$ by back-solving in $\mathcal{O}(nN)$ more operations
(7) subtract from result
... continue to the order of truncation.

This process converges if the spectrum of $L^{-1}BL^{T^{-1}}$ is in an interval $[-b, b]$ with $b < 1$. If a bounding interval $[-b, b']$ is known, then for any order, we can find

a minimax polynomial $p(x)$ such that

$$\max_{x\in[-b,b']} \|1/(1+x) - p(x)\| = \epsilon \tag{11}$$

is small. It follows that a linear combination,

$$L^{T^{-1}} p(L^{-1} B L^{T^{-1}}) L^{-1}(-\nabla\phi_i) \tag{12}$$

using the coefficients of p, of the terms calculated in the procedure above, provides a better approximation of Δx than the truncated Taylor series of the same order. (See [JMP83] for an explanation and application to preconditioned conjugate gradient.)

Using the minimax polynomial requires little extra machine computation, especially on machines with fused multiply-add instructions. Note that even if the Taylor series diverges, we can still find a minimax polynomial as long as the spectrum is bounded, although we wouldn't expect a good rate of convergence to result.

The optimization algorithm contains the above procedure as an inner iteration. The full procedure is

(1) compute gradient at current iterate (x_i);
(2) compute Hessian and $L^T L$ (if the objective is convex quadratic);
(3) solve for the approximate Newton step using the procedure above;
(4) repeat until the step size or decrease in objective value fail to cross a threshold.

We allow for non-stationary objective functions to accommodate non-convex objective functions which need to be solved in stages to prevent convergence to undesirable minima.

To ensure stability of the factorization step or to improve convergence, it may be necessary to add a multiple of the identity to the Hessian. Let α be this multiple and let $\mathcal{H}$ be the Hessian. The quadratic approximation to the objective can be written

$$\phi_{\text{approx}} = \frac{1}{2}(x - x_0)^T \mathcal{H}(x - x_0) + \phi_0, \tag{13}$$

for some constant ϕ_0. The gradient is $\mathcal{H}(x - x_0)$. The difference between the next iterate and the minimum point is

$$\begin{aligned} \|x + \Delta x - x_0\| &= \left\|x + p(A^{-1}B)A^{-1}(-\mathcal{H}(x - x_0)) - x_0\right\| \\ &\le \left\|p(A^{-1}B)A^{-1}(-\mathcal{H}(x - x_0)) - (\mathcal{H} + \alpha I)^{-1}(-\mathcal{H}(x - x_0))\right\| \\ &\quad + \left\|(\mathcal{H} + \alpha I)^{-1}(-\mathcal{H}(x - x_0)) - (x - x_0)\right\| \\ &\le \left(\epsilon \|\mathcal{H}\| + \left\|(\mathcal{H} + \alpha I)^{-1}\right\| \alpha\right) \|x - x_0\|. \end{aligned} \tag{14}$$

So if the spectrum of the Hessian is bounded away from zero, we can find a polynomial approximation of sufficient degree to make ϵ small enough to ensure that approximate Newton step is a contraction mapping, and the iteration converges. The rate of convergence depends on the conditioning of the Hessian and the order of the polynomial approximation.

We have shown that this framework converges in infinite precsion. It works even with low-precision arithmetic well past a level of convergence meaningful in imaging problems because errors do not accumulate from outer iteration to outer iteration, and errors in the inner iteration amount to tens of ulps which is below the error of the approximation.

For a polynomial $p(x) = \sum_i p_i x^i$, rounding errors during the inner iteration will be on the order of m ulps plus $\sum_i |p_i|/p_{\min}$ ulps multiplied errors in multiplication by B, where $p_{\min}$ is the minimum magnitude coefficient of the polynomial. In reasonable situations this result in a ratio under two, and in our test-case we have always been able to keep it under one. We use the fact, see [GVL96], that Cholesky decomposition of the diagonally-dominated blocks of A introduces at most a few ulps of error.

The dense operators B occurring in these types of problems (*i.e.* Fourier and Radon transforms) are stable in practice. Conventional image reconstructions would not work if this were not the case.

Note: In many embedded and large-data-throughput applications, numerical convergence criteria are not used, because the computation budget is fixed. In these cases, asymptotic convergence is not important in itself, but the implied numerical stability is important, since many such computations run unsupervised.

5. Penalties

Convergence of our method depends on the structure of the objective function. Because inverse problems come with a variety of structures, we would like to be able to ensure rapid convergence with the penalty terms. Two properties can ensure this: dominance of the block diagonal component of the Hessian and a good gradient direction.

5.1. **Bilateral Regularization.**

Modeling our first family of penalty functions on bilateral filtering, we replace (4) with a penalty

$$\phi_{\text{bi}}(f) = \sum_{y \neq x} c(y-x)s(f(y)-f(x)), \tag{15}$$

where c and s can be any kernel functions, including ones used in bilateral filtering. The choice of c and s is guided by

(1) previous use in statistics or filtering,
(2) descent direction, $-\nabla\phi_{\text{bi}}$,
(3) sparsity and conditioning of the Hessian.

We want the negative gradient to point in the direction of a more likely image than the current estimate. For problems where pixel values represent component properties, *e.g.* water content or radio opacity, images are expected to be piecewise constant to a first approximation, with most of the signal in the low-frequency components, while the noise is distributed evenly across frequencies. This leads to the design goal

$$-\nabla\phi_{\text{bi}}(f_{\text{high}}) = -f_{\text{high}}, \text{ while}$$
$$-\nabla\phi_{\text{bi}}(f_{\text{zero}}) = 0.$$

For one of the penalties will define, we can formulate this design problem using constrained linear optimization involving the coefficients of c. The optimization model is a multi-dimensional analogue of FIR filter design. Since the penalties form a family, it makes sense to use such an optimal c for the whole family.

For general c and s, the gradient and Hessian are

$$\frac{\partial}{\partial f_i(x)}\phi_{\text{bi}} = \sum_{\{y \mid y \neq x\}} \frac{\partial s(f(x)-f(y))}{\partial f_i(x)} c(x-y), \tag{16}$$

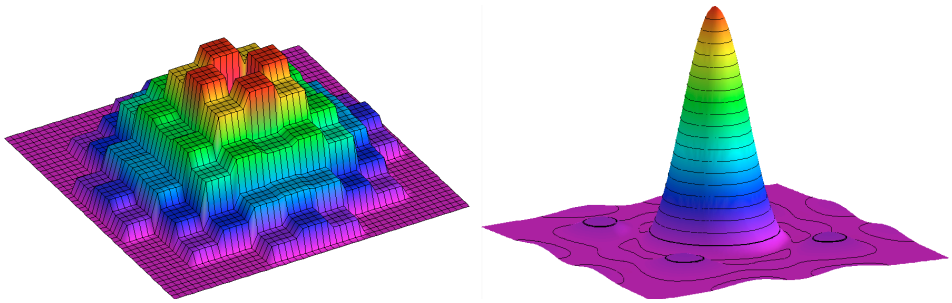

FIGURE 3. Optimal $\sqrt{18}$ spatial kernel and its transfer function, with contours at .05 intervals from -1.05 to $-.05$, with a double contour at -1.

and

$$\mathcal{H}_{f_i(x),f_j(y)}\phi_{\text{bi}} = \quad = \sum_{\{y \mid y \neq x\}} \frac{\partial^2 s(f(x)-f(y))}{\partial f_j(y)\,\partial f_i(x)} c(x-y). \tag{17}$$

The sparsity of the Hessian depends on the support of c and the derivatives of s. The derivatives of s are important when f is vector-valued. If s depends functionally on $\|f\|$ alone, then its Hessian is a multiple of the identity, and the full Hessian decomposes into identical, block banded with banded block blocks. In our implementation, we use this fact to reduce storage for the Hessian and the number of Cholesky decompositions.

Consider $s(t) = \|t\|^2$ (L^2 norm). To simplify the analysis, we assume $\sum_{y\neq 0} c(y) = 1$, and $c(y) \geq 0$. In this case we define

$$\phi_{\text{bi2}} = \sum_{y\neq x} c(y-x)\,\|f(y)-f(x)\|^2\,, \tag{18}$$

and compute

$$\frac{\partial}{\partial f_i(x)}\phi_{\text{bi2}} = 2\sum_{y\neq x}(f_i(x)-f_i(y))c(x-y) \quad = 2\left(f_i(x) - \sum_{x\neq y} f_i(x)c(x)\right), \tag{19}$$

and

$$\begin{aligned} &\mathcal{H}_{f_i(x),f_j(y)}\phi_{\text{bi2}} = 0, && \forall (i\neq j \text{ or } x\neq y)\\ &\mathcal{H}_{f_i(x),f_i(x)}\phi_{\text{bi2}} = 2\sum_{y\neq 0} c(y) = 2, \\ &\mathcal{H}_{f_i(x),f_i(y)}\phi_{\text{bi2}} = -2c(x-y). \end{aligned} \tag{20}$$

We observe that $\nabla\phi_{\text{bi2}}(f)$ is the convolution of f with the kernel formed by 1 at zero and $-c(x)$ for other values of x. This is why the design can be formulated as linear programming. In Fig. 3 we show the optimal two-dimensional discrete function c with support in a disc of radius $\sqrt{18}$ which we use in the numerical examples in this paper.

Other functions s to consider are motivated by use in statistics and filtering:

$$\phi_{\text{biTV}} = \sum_{y\neq x} c(y-x)\,\|f(y)-f(x)\|\,, \tag{21}$$

$\|x\|$	$c(x)$
1	0.04071725
2	0.03499660
4	0.02368359
5	0.02522255
8	0.02024067
9	0.01407202
10	0.01345276
13	0.00850939
16	0.00812839
17	0.00491274
18	0.00396661

TABLE 1. Optimized definition of c used in numerical examples.

is equal to the Total Variation, when $c(x-y)$ is zero except for immediate neighbors, and hence has all the problems associated with the discontinuity at the origin and zero Hessian;

$$\phi_{\text{biHuber}} = \sum_{y \neq x} c(y-x) s_{\text{Huber}}(f(y) - f(x)), \tag{22}$$

in which s_{Huber} is the Huber function used in robust statistics. It is the C^1 function which equals the absolute value outside a neighborhood of zero and a parabola inside. This penalty has a continuous gradient but discontinuous Hessian.

$$\phi_{\text{biNormal}} = -\sum_{y \neq x} c(y-x) e^{\|f(y)-f(x)\|^2}. \tag{23}$$

corresponds to the original kernel used in bilateral filtering, but reduces sparsity and conditioning in the Hessian (since it may be nonconvex); while

$$\phi_{\text{biTv}} = \sum_{y \neq x} \frac{\epsilon c(y-x)}{\sqrt{\left\| \tilde{f}(y) - \tilde{f}(x) \right\|^2 + \epsilon}} \|f(y) - f(x)\|^2, \tag{24}$$

(25)

which is a pixel-scaled version of ϕ_{bi2} in which the scaling depends on the previous estimate ($\tilde{f}$), and hence has the same sparsity (and symmetry among components). It improves conditioning differently as a function of the current estimate, which may improve actual convergence, while weakening bounds on worst-case convergence. We designed this to be a better-behaved and easier-to-implement version of Total Variation. Fig. 4 shows how the one approximates the other. For real-valued images, we could reformulate this penalty using linearly-constrained optimization, but this is out of the scope of this paper.

In the numerical results section, we report on the behaviour of ϕ_{bi2} and ϕ_{biTv}, leaving the other kernels for a future work.

Every nonzero value of c corresponds to nonzero (sub/super)diagonals in $\mathcal{H}\phi_{\text{bi2}}$. In most applications, this means that the memory requirements associated with the approximate Newton step will be $\mathcal{O}(|c| \times (\text{image size}))$, where $|c|$ is the size of the

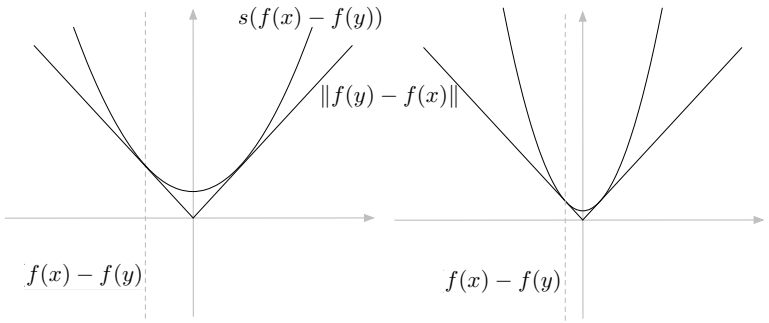

FIGURE 4. Two examples of the biTv kernel approximating the absolute value. The approximation depends on the current value of the difference, as shown on the left. On the right we see that the difference with the absolute value is quite large when the going from a small difference to a large one. This has the effect of limiting step length for steps which would increase differences.

support of c. Functions c with support in a disk (ball for higher dimensions), give the best trade-off in terms of memory/performance. The function in Fig. 3 shows a degree of rotational symmetry, and its transfer function (its Fourier transform) shows that pure spatial frequencies with periods smaller than six pixels, which represent 91 percent of all non-aliasing frequencies, would be reduced in magnitude by 95 percent by a gradient step (if the penalty were the entire objective function). Choice of the support for c and the frequencies to try to eliminate warrant further investigation, and in a future paper we will give the details of the optimization problem, profile a range of choices, and do the same for the spatial kernel of the original bilateral filter.

Note that the penalty ϕ_{bi} is a bound on the difference between a pixel and the weighted average of its neighbors:

$$\sum_{y \neq x} c(y-x) \, \|f(y) - f(x)\|^2 \geq \left\| f(x) - \sum_{y \neq x} c(y-x) f(y) \right\|, \tag{26}$$

but that the inequality is strict generically. The latter would result in roughly twice the number of nonzeros in the Hessian, reducing computational efficiency.

5.2. **Masking.** In slice/volume reconstuctions, the imaged object is often surrounded by air, which in a perfect reconstruction appears as pure noise, but in imperfect reconstructions may contain incorrectly-attributed signal. If the object/air interface can be determined from the device design, or from a low-quality reconstruction, a penalty

$$\phi_{\text{mask}} = \sum_{\{x \,|\, x \text{ is air}\}} \|f(x)\|^2 \tag{27}$$

can be used to push the pixel values to zero (or whatever the appropriate signal for the surrounding air or other substance is). The contribution to the gradient is negative the image value, and the Hessian is diagonal, constant two outside the

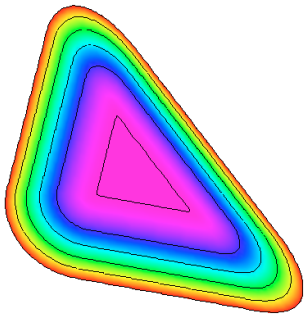

FIGURE 5. Surface rendering of the two-dimensional version of ϕ_{simplex} showing contours at 0,.2,.4,.6,.8.

object and zero inside. If set measurement is uncertain, the penalty can weight the contributions from different pixels according to the certainty of not being inside the object, or we could simply consider questionable pixels to be part of the object.

In limited experience, this works better than constraining those pixels to be zero and then eliminating those variables from the problem. Perhaps this is true in high-noise regimes where assigning external noise pixels to internal pixels can actually make noise problems worse. It is certainly easier to implement boundary conditions and do parallelization when the number of pixels in the object remains constant.

5.3. **Partial Volume.** In problems where the pixel values are modeled as linear combinations of the signals corresponding to different object components (*e.g.* muscle, fat, *etc.* for medical imaging, or forest, field, concrete, *etc.* for remote sensing), the reconstructed vector pixel values should form a simplex (if the pure signals are linearly-independent and the reconstructed values are expressed in a basis formed by extension) or another convex polytope.

In either case, we can add penalty functions to penalize pixels outside the polytope.

We see two ways of implementing such penalties: simplex basin and edge attractor. Let

$$\psi(t) = \begin{cases} t^2 & t < 0 \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

Define

$$\phi_{\text{simplex}}(f) = \sum_i \psi(f_i) + \psi(1 - \sum_i f_i). \tag{29}$$

For two-vectors, the corresponding basin of convergence is visualized in Fig. 5. This penalty is convex and easy to compute. Its Hessian is discontinuous, however, but this could be fixed by switching to the fourth power for the penalty components. If the measurements can be converted to the natural basis of the simplex (extended in the case of more measurements than components), the Hessian is diagonal. Unlike the Hessians of other penalty functions, however, it is not constant on the diagonal elements corresponding to one pixel. In fact, it could be one on any subset of the diagonal and zero on the rest corresponding to components. (It is always one on the remaining directions).

An alternative is

(30)
$$\phi_{\text{magnet}}(f) = \begin{cases} (f - f_0)^2 & f \text{ in the exterior and projects to } f_0 \text{ on the boundary,} \\ 0 & f \text{ in the interior.} \end{cases}$$

This penalty has discontinuous Hessian, but the Hessian is diagonal in any basis for the pixel values, because it is either two or zero on all diagonal elements corresponding to one pixel. If all other penalties have this property, then storage for A, B, and L can be reduced, along with the number of Cholesky decompositions. This comes at the expense of penalizing movement along the boundary of the simplex for pixels outside the simplex.

An alternative to both methods would be to use barriers. Since they are undefined outside the simplex, this would require additional computation to ensure that we never leave the simplex.

5.4. **Segmentation.** In many applications, images are used to make quantitative determinations of component areas/volumes. For example, grey and white matter volume is important in tracking development in children and degenerative diseases in the elderly. Segmentation into components is also required before surface rendering.

In such cases, where images are reconstructed from source data, the most likely segmentations can be determined by incorporating a probability distribution on pixel values into the reconstruction via a penalty function:

(31)
$$\phi_{\text{seg}}(f) = \sum_{\{g \,|\, \text{mean values of components}\}} e^{-\|f-g\|^2/\sigma^2}.$$

This function is not convex for small values of σ, so we have to consider multiple local minima. When it is convex, however, its unique minimum is in the interior of the simplex and not assignable to any component. A reasonable way of getting to a better local minimum is to increase the weighting of this penalty and decrease the size of σ as a function of iteration count. As the weighting of this penalty function gets higher and σ gets smaller, pixels will be forced to move to one of the component means. If the weighting is too high initially, however, some pixels will be trapped in the wrong basin of convergence.

Another consequence of the nonconvexity is that the Hessian may have negative values on the diagonal. This happens for pixel values far away from the simplex, or when σ is small enough for pixels in the interior of the simplex. This will reduce the diagonal of A and even without making A singular and the Cholesky decomposition ill-defined, it tends to increase the eigenvalues of $L^{-1}BL^{T-1}$ which reduces the rate of convergence of the Neumann series and increases the error in the polynomial approximation of $(A+B)^{-1}$. We can compensate by adding a multiple of the identity to the combined Hessian, which means taking a smaller step which is more gradient step and less Newton step, both of which slow convergence. Because of these factors, it is best use this penalty after other penalties have stopped rapidly converging, and only use it in combination with convex penalties.

5.5. **Fit to Data.** The preceding penalties are some of the penalties which encode a priori information about the expected solution, which may be modified and introduced at different iterations to improve convergence. Every problem must also contain a fit-to-data term which is dictated by the problem.

Our assumption is that the problem has a readily computable forward problem: a way of going from model to expected data. If the forward problem is linear $T : P \to M$, where D is the vector space of pixel data and M is the vector space of measurement data, and the measurements are contaminated by white noise, the fit-to-data term is

$$\phi_{\text{data}}(f) = \|Tf - m\|^2, \tag{32}$$

where m are the measurements. The gradient is given by

$$\nabla\phi_{\text{data}} = 2T^T m, \tag{33}$$

and the Hessian by

$$\mathcal{H}\phi_{\text{data}} = 2T^T T. \tag{34}$$

Where the adjoint T^T can always be computed by similar fast methods. The most common case is when T is a close relative to the Radon transform or nonuniform Fourier transform, in which case the adjoint can be accelerated using the same methods.

Modifications for unequal or correlated noise are straightforward. Modifications for nonlinear forward problems require another level of analysis. If the nonlinear problem is convex, and the difference between the nonlinear model and its linearization can be bounded in a trust region, then our framework provides the tools required to solve the nonlinear problem. Simply substitute the linearization of the forward problem at each iteration, and add a multiple of the identity to the overall Hessian to ensure the next step stays within the region on which the linearization is a good approximation. As the solution approaches a minimum for the general nonlinear problem, the step size will reduce in size and converge linearly to the local optimum. We will explore this issue in detail for specific nonlinear forward problems in a future paper.

Unlike the previous penalties, $\mathcal{H}\phi_{\text{data}}$ will in general be dense, and the exact values of a split $A_{\text{data}} + B_{\text{data}} = \mathcal{H}\phi_{\text{data}}$ which matches the sparsity pattern for A may be too expensive to determine exactly. In this case, the requirement on the split is for A to have the same sparsity pattern as the diagonal blocks of the other Hessian components, and B to come with the smallest possible spectral radius.

For example, sparsely sampled radial sampling in MRI and sparse spiral samplings in transmission tomography (including CT), result in $T^T T$ being approximately shift-invariant, given as convolution with a point spread function (psf). The psf approximates a delta function at the origin, and for different sparse samplings above includes low-intensity star patterns centred at the origin. The first term, A_{data} is formed by taking the $2m + 1$ adjacent pixels centred at the origin in the row direction. Fig. 6 shows an actual psf (see [ACK06]), with a yellow outline to show the pixels which contribute to A_{data}.

In many cases, other information can be used to improve the estimate for A_{data}, by understanding the underlying physical model. For example, in SENSE imaging, the columns of A_{data}, taken from the psf can be modulated by the sum of the squared magnitudes of the coil sensitivities ($S_{m;i,j}$).

Even in cases where complete information about T may be unavailable, it may be possible to find a good choice for A_{data}. For example, if T corresponds to partial data loss for samples of the Fourier transform of the image, then the eigenvalues are all one or zero (with eigenvalues coming from the Fourier basis), and $A_{\text{data}} = 1/2$

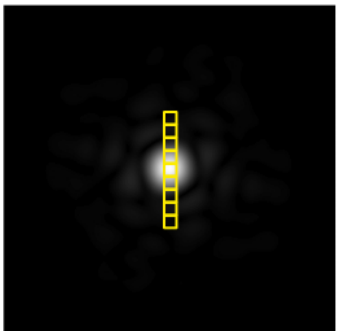

FIGURE 6. Point spread function for MRI sparse sampling trajectory, showing the pixels which correspond to the values of A_{data}.

may be the best estimate available (which limits the absolute value of eigenvalues of $B_{\text{data}} = T^T T - A_{\text{data}}$ to $\pm 1/2$.).

For the specific problem we are using for benchmarking, the images are complex. The psf is real, of the form $\delta_{0,0} + \delta_{0,128}$ (for 256×256 images), which captures the sparsity pattern of the full Hessian. The diagonal of the Hessian is

$$(A_{\text{SENSE2}})_{(i,j),(i,j)} = \sum_m \|S_{m;i,j}\|^2 , \tag{35}$$

by which we mean the diagonal elements of the Hessian corresponding to pixel (i, j). The value is the same for the real and imaginary parts of the image, and the off-diagonal part of the Hessian corresponding to mixed real/imaginary and imaginary/real parts of a single pixel are zero. This allows us to reduce the storage of A and the number of Cholesky decompositions by 1/2.

On the other hand, the contribution to the off-diagonal blocks is not real, so there are twice as many nonzero elements (not counting symmetry in the Hessian):

$$(B_{\text{SENSE2}})_{(i,j),(i,j+128)} = \sum_m S_{m;i,j} S^*_{m;i,j+128}, \tag{36}$$

by which we mean that the four nonzero elements corresponding to the interaction of pixels (i, j) and $(i, j+128)$ correspond to multiplication by this complex number. Since these values are applied directly in the form of a matrix-vector product, and not involved in a Cholesky decomposition, there is no effect on space.

Since the energy of $\mathcal{H}\phi_{\text{data}}$ is divided equally between the diagonal and off-diagonal blocks, this problem can be expected to be more difficult to invert by this method than typical target applications, for which the psf is not sparse, but is concentrated at the origin. Further investigation is required to see how this fact balances out the fact that we don't have a fast exact decomposition into block diagonal and off-diagonal parts.

The total memory footprint of this algorithm depends strongly on the fast algorithm used to calculate T and T^T, but the additional footprint for the regularization will be $(|\text{supp}(c)|/2 + \text{width}(c)) \times (\text{image size})$. For most target problems, the computational cost associated with c will be small relative to the fit-to-data term, so the space requirements are what needs to be traded-off with the improvement per iteration.

6. Numerical Results

Numerically, we verify linear convergence, with all computations done in single precision. (All variables were declared to be single precision, and we verified that single-precision instructions were used in some of the assembly code.) We verify convergence with a variety of different penalty functions and in two regimes: a small number of iterations, to simulate real-time imaging, and a larger number to ensure convergence past visibly-detectable differences. For the simulated real-time situations, we chose 15 iterations, because we wanted to add some additional penalties and they worked best after the initial 10 iterations, but we wanted to keep the number of iterations down. For 15 iterations, we compare the optimized kernel c with the simplest possible kernel, the so-called five-point stencil with obviously unacceptable results for the smaller kernel. We also compare the penalties ϕ_{bi2} and ϕ_{biTv} for visual differences in edge sharpness.

The implementation is in ANSI C, with minimal space-optimizations, taking advantage of the banded structure to allow large problems to execute in RAM. No optimization was done for speed, but even at 100 iterations, the processing speed is comparable the time `gnuplot` required to plot the results.

6.1. **Convergence.** We deliberately chose to test in a high-noise regime, so we expected many iterations to be necessary. In Fig. 7, we see that the linear convergence continues to the 100th iteration, both with the ϕ_{bi2} and ϕ_{biTv} penalties, plus additional penalties after 10 iterations. For the first 10 iterations, see Fig. 8, we used equal weighting for the penalty and fit-to-data term,

$$\min \lambda\phi_{\text{data}} + \lambda_{\text{bi2}}\phi_{\text{bi2}} + \lambda_{\text{biTv}}\phi_{\text{biTv}} + \lambda_{\text{mask}}\phi_{\text{mask}} + \lambda_{\text{magnet}}\phi_{\text{magnet}} + \lambda_{\text{seg}}\phi_{\text{seg}} \tag{37}$$

$\lambda = 1 = \lambda_{\text{bi2}}$ and use a very small regularizer $\alpha = .0125$. In both cases, we perform a simple mask calculation at this point (shrinking a circle until the average pixel value outside the circle makes a sharp increase). We then add the mask penalty $\lambda_{\text{mask}} = 1$ for the remaining even iterations and the magnet penalty $\lambda_{\text{magnet}} = 1$ for iterations 40...59. At iteration 20 we introduced ϕ_{seg}. We decreased the standard deviation of the normal distribution according to $\sigma = .4 + .03/(1+\text{iteration} - 20)$, with the aim of introducing a convex penalty and gradually switching to a more non-convex penalty. (To be safe, we increased α to ensure that the Hessian stays sufficiently positive definite.) After iterations 50 and 70 we decrease the weighting of the fit-to-data term, to .1 and .07 respectively, increase the weight of segmentation to 1.1 and 1.35, and increase the regularizer α to 1.5 and 3.

For comparison, see Fig. 9, we made the same choices with ϕ_{bi2} replaced by ϕ_{biTv} after iteration 10, and ϵ, which controls the maximum thinness of the parabola approximating the absolute value to .06 and then .03 after iteration 75. The convergence plot shows good behavior with all combinations of penalties. Further emphasizing the penalties at iteration 75 makes little difference over the change at iteration 50. This is consistent with expectations based on L-curves for other inverse problems, but does not verify such behaviour in this case.

If you look carefully at Fig. 9, you can see enhancement of some edges using ϕ_{biTv} versus ϕ_{bi2}, but not enough to recommend one over the other.

Of course, we are very interested in embedded real-time applications. In these cases, we cannot wait for convergence, we must do the best we can in a given time budget. To test the algorithm in this case, we repeat the same initial iterations, but follow them with iterations with more aggressive parameters including ϕ_{seg}

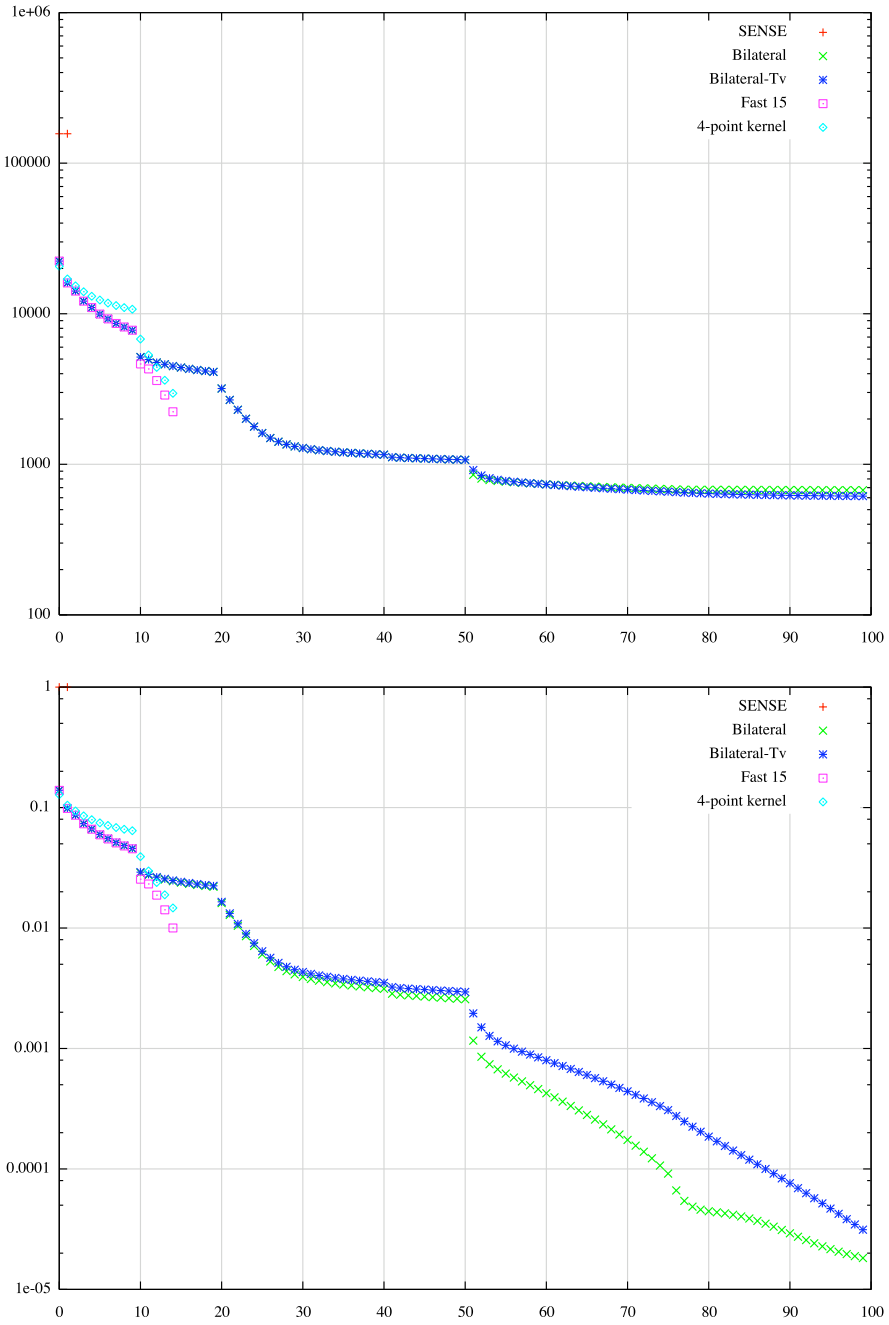

FIGURE 7. Convergence using by stages the simplest to the most complex penalty terms. Normalized to make the error in the direct inverse 1. On the top we plot the total L^2 error (versus the true image). On the bottom we plot the difference between the error of the current iterate and the error in the limit, to show that the linear convergence continues up to the 100th iteration.

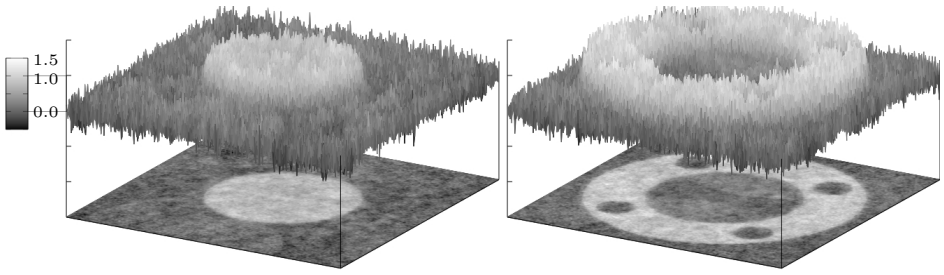

FIGURE 8. Initial 10 iterations, using ϕ_{bi2}.

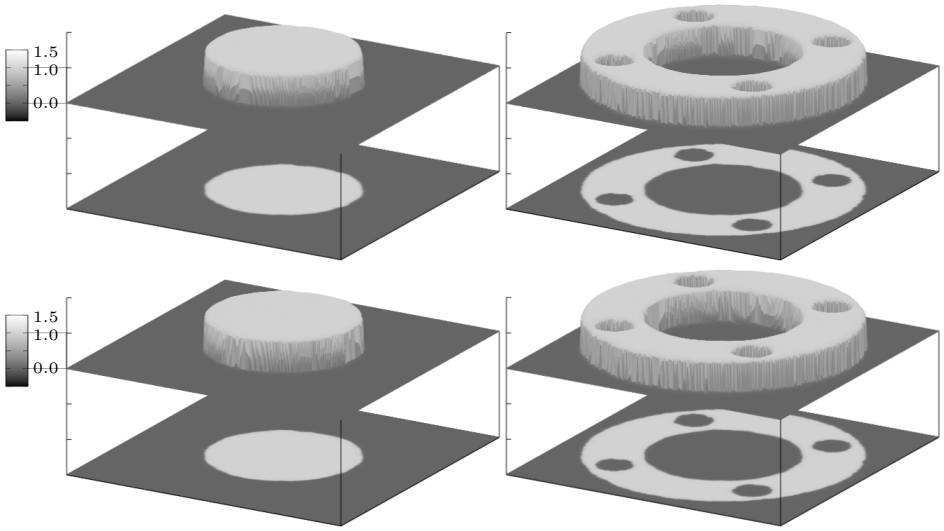

FIGURE 9. After a total 100 iterations, introducing more penalty terms as iteration count increases, comparing ϕ_{bi2} *(top)* and ϕ_{biTv} *(bottom)*.

from iterations 11 instead of from iteration 20 in the other case. As the graph shows, being more aggressive does produce faster convergence, and looking at the resulting images, Fig. 10, shows the images to be faithful representations of the test object. For comparison, to show the value of optimizing c to produce gradient directions which reduce high-frequency components, we did the same experiment with a four-point kernel c (.25 weighting to all of the differences between horizontal and vertical neighbors). As the images in the bottom of the figure show, we do not get a faithful representation of the test object with this kernel. This demonstrates both the superior performance of the optimized c, and the danger in introducing non-convex penalties too early and dropping into the "wrong" basin of convergence. In the embedded case, the convergence properties of the inverse problem may be fixed at design time (when the data collection hardware and parameters are set). So, although the data depends on individual experiments, the penalty weights and parameters can be optimized in advance using expensive methods, *i.e.* pattern search or more sophisticated methods.

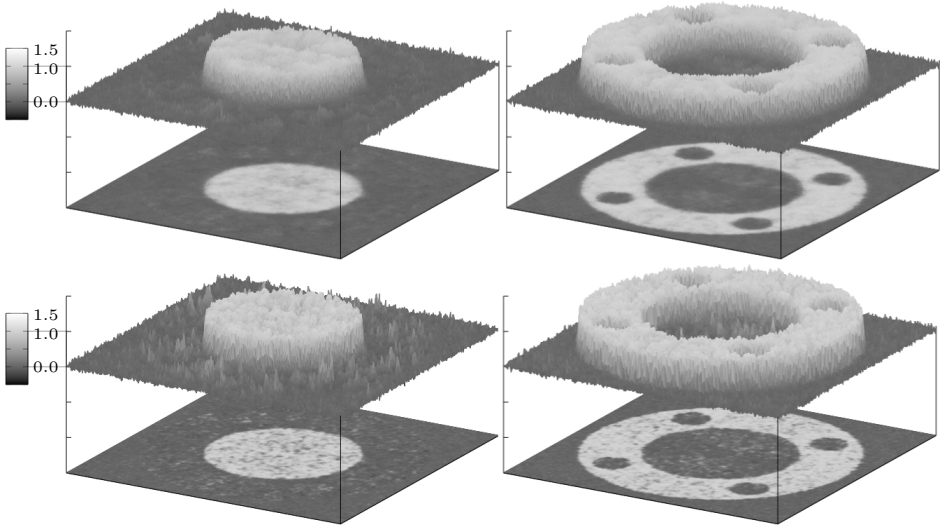

FIGURE 10. Fast 15 iterations, using width $\sqrt{18}$ stencil *(top)*, and 4-point (width one) stencil *(bottom)*.

7. Future Work

Implementation work is underway to show that we can achieve peak performance on the target architecture (Cell). Operating system support for this platform is still not mature, so there are challenging technical issues to overcome.

We are working on a technical report with a full set of optimized kernel functions c, similar kernels for bilateral filtering, and an analysis of their performance relative.

We have shown our method is robust with respect to input noise and rounding errors associated with single precision, and that performance is good for a test problem. It would be interesting to test it on standard imaging problems, *e.g.* deblurring and inpainting. In the standard problems, propagating information across the image requires many iterations, or the solving of large linear systems. This is motivation for recent work on domain decomposition and multi-grid methods, see [GY05], [CC06]. Our method provides another potential solution to the signal propagation problem, at the expense of some extra data shuffling. At each iteration, we are making an approximate Newton step. We can bound the error in the approximation in the L^2 norm, but in imaging problems, such measures can be misleading. In our case, the Newton step is anisotropic, in the sense that for a problem symmetric with respect to column translation, the Cholesky decomposition and back substitution are exact for the regularized problem. Signals propagate along the entire row in each step. Problems symmetric in the row direction, however, do not behave in the same way, since signals cannot propagate farther than the width of c multiplied by the order of the polynomial approximation. So our approach could be described as domain decomposition into rows. But at the expense of a transpose, we can alternate rows and columns (or as many dimensions as exist for a particular problem), without incurring any additional penalty.

L-curves methods should be applied to the larger parameter selection problem introduced here. Can the work on L-curves help with the related problem of steering

nonlinear optimization problems? The segmentation penalty is a simple nonlinear term, but of definite value to the many applications which ultimately require segmented volumes and not images. In the four-point kernel example, we have seen what looks like convergence to unacceptable local minima. In this case, we could ensure convergence to a good optimum by using ϕ_{bi2} with the optimized kernel c, and using sufficiently many initial iterations (based on a numerical termination criteria), but it would be nice to be able to know the earliest point at which it is safe to use the nonlinear penalty, since it significantly improves convergence.

Given our multiple penalties, we could apply a filter algorithm.

A problem which is not evident from the numerical tests, perhaps because of the effectiveness of ϕ_{seg}, is that all of the penalty terms lead to biased estimation problems, in which we expect the estimates to lie inside the simplex/polytope, even if all the true values are on the boundary. Can the estimates be made unbiased in a natural way? Is there a link to path-following methods in constrained optimization?

Although we conceived of this work as an alternative to Krylov space methods, everything we have done to accelerate the Newton steps could be applied either to precondition conjugate gradient iteration or to modify the objective of the CG iteration.

8. Conclusion

We have presented a first-order framework for solving linear and nonlinear inverse problems whose model variables are arranged in grids, *i.e.* images, discrete volumes, *etc.*This method is robust in the face of both reduced-precision computation and high levels of measurement error. We have introduced a number of penalty functions, some of which we have tested numerically. Contrary to our own expectations, the simplest bilateral penalty to implement, ϕ_{bi2}, performs as well as the more complicated ϕ_{biTv} penalty.

This method is, by design, well-suited to all the types of parallelism we need to exploit to benefit from current commodity architectures. The simple nature of its error analysis recommends it to environments where algorithms need to be certified at design time, and provides a method for analyzing nonlinear problems.

References

[ACK06] Christopher Kumar Anand, Andrew Thomas Curtis, and Rakshit Kumar. Durga: A heuristically-optimized data collection strategy for volumetric magnetic resonance imaging. http://www.cas.mcmaster.ca/ anand/papers/preprints.html, 2006.

[AM06] Marcia L. S. Aguena and Nelson D. A. Mascarenhas. Multispectral image data fusion using pocs and super-resolution. *Comput. Vis. Image Underst.*, 102(2):178–187, 2006.

[Bar02] Danny Barash. A fundamental relationship between bilateral filtering, adaptive smoothing, and the nonlinear diffusion equation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(6):844–847, 2002.

[Bar05] Danny Barash. Nonlinear diffusion filtering on extended neighborhood. *Appl. Numer. Math.*, 52(1):1–11, 2005.

[BCRS03] M. Bertalmio, V. Caselles, B. Rougé, and A. Solé. TV based image restoration with local constraints. *J. Sci. Comput.*, 19(1-3):95–122, 2003.

[BSMH97] Michael J. Black, Guillermo Sapiro, David H. Marimont, and David Heeger. Robust anisotropic diffusion: Connections between robust statistics, line processing, and anisotropic diffusion. In *SCALE-SPACE '97: Proceedings of the First International Conference on Scale-Space Theory in Computer Vision*, pages 323–326, London, UK, 1997. Springer-Verlag.

[CC06] Tony F. Chan and Ke Chen. An optimization-based multilevel algorithm for total variation image denoising. *Multiscale Model. Simul.*, 5(2):615–645, 2006.

[CCT04] Eric T. Chung, Tony F. Chan, and Xue-Cheng Taib. Electrical impedance tomography using level set representation and total variational regularization. *Journal of Computational Physics*, 2004.

[CGM99] Tony F. Chan, Gene H. Golub, and Pep Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM J. Sci. Comput.*, 20(6):1964–1977, 1999.

[Cha04] Antonin Chambolle. An algorithm for total variation minimization and applications. *J. Math. Imaging Vis.*, 20(1–2):89–97, 2004.

[GVL96] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[GY05] Donald Goldfarb and Wotao Yin. Second-order cone programming methods for total variation-based image restoration. *SIAM J. Sci. Comput.*, 27(2):622–645, 2005.

[GZ03] Keith O. Geddes and Wei Wei Zheng. Exploiting fast hardware floating point in high precision computation. In *ISSAC '03: Proceedings of the 2003 international symposium on Symbolic and algebraic computation*, pages 111–118, New York, NY, USA, 2003. ACM Press.

[HHK+03] Walter Hinterberger, Michael Hintermüller, Karl Kunisch, Markus Von Oehsen, and Otmar Scherzer. Tube methods for bv regularization. *J. Math. Imaging Vis.*, 19(3):219–235, 2003.

[HS52] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand.*, 49:409–436, 1952.

[Hub81] P. J. Huber. *Robust Statistics*. John Wiley, New York, 1981.

[JMP83] O. G. Johnson, C. A. Micchelli, and G. Paul. Polynomial preconditioners for conjugate gradient calculations. *SIAM J. Numer. Anal.*, 20, 1983.

[JO06] Jin Hyuk Jung and Dianne P. O'Leary. Cholesky decomposition and linear programming on a gpu. Technical report, University of Maryland, 2006.

[KD06] J. Kurzak and J. Dongarra. Implementation of the mixed-precision high performance linpack benchmark on the cell processor. Computer Science Tech Report UT-CS-06-580, LAPACK Working Note #177, University of Tennessee, September 2006.

[KDH+05] J. A. Kahle, N. N. Day, H. P. Hofstee, C. R. Johns, T. R. Maeurer, and D. Shippy. Introduction to the cell multiprocessor. *IBM Journal of Research and Development*, 49(4/5):589–604, July/September 2005.

[LP06] Germana Landi and Elena Loli Piccolomini. Representation of high resolution images from low sampled fourier data: Applications to dynamic mri. *J Math Imaging Vis*, 26:27–40, 2006.

[MSP86] P. Margosian, F. Schmitt, and D. E. Purdy. Faster mr imaging: Imaging with half the data. *Health Care Instr.*, 1:194, 1986.

[NMDI94] M. Nikolova, Ali Mohammad-Djafari, and J. Idier. Inversion of large-support illconditioned linear operators using a Markov model with a line process, 1994.

[PM90] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12(7):629–639, July 1990.

[PWBB01] Klaas P Pruessmann, Markus Weiger, Peter Börnert, and Peter Boesiger. Advances in sensitivity encoding with arbitrary k-space trajectories. *Magn Reson Med*, 46:638–651, 2001.

[ROF92] Leonid I. Rudin, Stanley J. Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[SKPJ04] Alexei A. Samsonov, Eugene G. Kholmovski, Dennis L. Parker, and Chris R. Johnson. Pocsense: Pocs-based reconstruction for sensitivity encoded magnetic resonance imaging. *Magnetic Resonance in Medicine*, 52(6):1397–1406, 2004.

[TM98] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *ICCV*, pages 839–846, 1998.

[WNC01] Jean-Marc Wagner, Frédéric Noo, and Rolf Clackdoyle. 3-d image reconstruction from exponential x-ray projections using neumann series. In *ICASSP*, 2001.

[ZBS01] Yongyue Zhang, Michael Brady, and Stephen Smith. Segmentation of brain mr images through a hidden markov random field model and the expectation-maximization algorithm. *IEEE Transactions on Medical Imaging*, 20(1):45–57, 2001.

Department of Computing and Software, McMaster University, Hamilton, Ontario, Canada L8S 4K1