LAP: A LINEARIZE AND PROJECT METHOD FOR SOLVING
INVERSE PROBLEMS WITH COUPLED VARIABLES

JAMES HERRING∗, JAMES NAGY∗, AND LARS RUTHOTTO∗

Abstract. Many inverse problems involve two or more sets of variables that represent different physical quantities but are tightly coupled with each other. For example, image super-resolution requires joint estimation of image and motion parameters from noisy measurements. Exploiting this structure is key for efficiently solving large-scale problems to avoid, e.g., ill-conditioned optimization problems.

In this paper, we present a new method called Linearize And Project (LAP) that offers a flexible framework for solving inverse problems with coupled variables. LAP is most promising for cases when the subproblem corresponding to one of the variables is considerably easier to solve than the other. LAP is based on a Gauss-Newton method, and thus after linearizing the residual, it eliminates one block of variables through projection. Due to the linearization, the block can be chosen freely and can represent quadratic as well as nonlinear variables. Further, LAP supports direct, iterative, and hybrid regularization as well as constraints. Therefore LAP is attractive, e.g., for ill-posed imaging problems. These traits differentiate LAP from common alternatives for this type of problems such as variable projection (VarPro) and block coordinate descent (BCD). Our numerical experiments compare the performance of LAP to BCD and VarPro using four coupled problems with one quadratic and one nonlinear set of variables.

Key words. Nonlinear Least Squares, Gauss-Newton Method, Inverse Problems, Regularization, Image Processing, Variable Projection

AMS subject classifications. 65F10, 65F22, 65M32

1. Introduction. We present an efficient Gauss-Newton method called Linearize And Project (LAP) for solving large-scale optimization problems whose variables consist of two or more blocks representing, e.g., different physics. Problems with these characteristics arise, e.g., when jointly reconstructing image and motion parameters from a series of noisy, indirect, and transformed measurements. LAP is motivated by problems in which the blocks of variables are nontrivially coupled with each other but some of the blocks lead to well-conditioned and easy-to-solve subproblems. As two examples of such problems arising in imaging, we consider super-resolution [27, 10, 4] and motion corrected Magnetic Resonance Imaging (MRI) [1, 6].

A general approach to solving coupled optimization problems is to use alternating minimization strategies such as Block Coordinate Descent (BCD) [21, 24]. These straightforward approaches can be applied to most objective functions and constraints and also provide flexibility to include regularization strategies. However, alternating schemes have been shown to converge slowly for problems with tightly coupled blocks [24, 4]. Additionally, as we demonstrate in a numerical experiment, BCD can be very sensitive to initializations when used for solving non-convex problems.

One specific class of coupled optimization problems, which has received much attention, is separable least squares problems; see, e.g., [16, 15, 25]. Here, the variables can be partitioned such that the residual function is linear in one block and nonlinear in the other. One common method for solving separable least-squares problems is Variable Projection (VarPro) [16, 15, 25]. The idea is to derive a nonlinear least squares problem of reduced size by eliminating the linear variables through projections. VarPro is most effective when the projection, which entails solving a linear least-squares problem, can be computed cheaply and accurately. When the number of...
linear variables in the problem is large, iterative methods can be used to compute the projection [11], but iterative methods can become inefficient when the least-squares problem is ill-posed. Further, standard iterative methods do not provide bounds on the optimality, which are needed to ensure well-defined gradients; see our discussion in Sec. 2.2. We note that recently proposed methods such as in [9] can provide that information and are, thus, attractive options in these cases. Another limitation of VarPro is that it is not straightforward to incorporate inequality constraints on the linear variables. Some progress has been made for box-constraints using a pseudo-derivative approach [29] leading to approximate gradients for the nonlinear problem. Finally, VarPro limits the options for adaptive regularization parameter selection strategies for the least-squares problem.

As an alternative to the methods above, we propose the LAP method for solving coupled optimization problems. Our contributions can be summarized as follows:

- We propose an efficient iterative method called LAP that computes the Gauss-Newton step at each iteration by eliminating one block of variables through projection and solving the reduced problem iteratively. Due to the linearization of Gauss-Newton, any block can be eliminated. Further, LAP supports various types of regularization strategies and constraints for all variables.
- We explore different regularization strategies including Tikhonov regularization using the gradient operator with a fixed regularization parameter and a hybrid regularization approach [5], which simultaneously computes the search direction and selects an appropriate regularization parameter at each iteration.
- We use projected Gauss-Newton to implement element-wise lower and upper bound constraints with LAP on the optimization variables. We note adding inequality constraints on the linear variables in VarPro is non-trivial.
- We compare the performance of LAP with block coordinate descent (BCD) and variable projection (VarPro) by analyzing convergence, CPU timings, number of matrix-vector multiplications, and the solution images and motion parameters for several coupled problems with linear imaging models including 2D and 3D super-resolution and motion correction for magnetic resonance imaging.

Our paper is organized as follows: Sect. 2 introduces a general formulation of the motion-corrected imaging problem, which we use to motivate and illustrate LAP, and briefly reviews BCD and VarPro; Sect. 3 explains our proposed scheme, LAP, for the coupled Gauss-Newton iteration along with a discussion of regularization options and implementation of bound constraints using projected Gauss-Newton; and Sect. 4 provides experimental results for several examples using LAP and compares it with BCD and VarPro. We end with a brief summary and some concluding remarks.

2. Motion-Corrected Imaging Problem. In this section we give a general description of coupled optimization problems arising in image reconstruction from motion affected measurements and briefly review Block Coordinate Descent (BCD) and Variable Projection (VarPro).

We follow the guidelines in [23], and consider images as continuously differentiable and compactly supported functions on a domain of interest, \( \Omega \subseteq \mathbb{R}^d \) (typically, \( d = 2 \) or 3). We assume that the image attains values in a field \( \mathbb{F} \) where \( \mathbb{F} = \mathbb{R} \) corresponds to real-valued and \( \mathbb{F} = \mathbb{C} \) to complex-valued images. We denote by \( x \in \mathbb{F}^n \) a discrete image obtained by evaluating a continuous image at the cell-centers of a rectangular grid with \( n \) cells.
The discrete transformation $y \in \mathbb{R}^{d \times n}$ is obtained by evaluating a function $y : \Omega \rightarrow \mathbb{R}^d$ at the cell-centers and can be visualized as a transformed grid. For the rigid, affine transformations of primary interest in this paper, transformations are comprised of shifts and rotations and can be defined by a small set of parameters denoted by the variable $w$, but in general the number of parameters defining a transformation may be large. We observe that under a general transformation $y(w)$, the cell-centers of a transformed grid do not align to the cell-centers of the original grid, so to evaluate a discretized image under a transformation $y$, we must interpolate using the known image coefficients $x$. This interpolation can be represented via a sparse matrix $T(y(w)) \in \mathbb{R}^{n \times n}$ determined by the transformation. For the examples in this paper, we use linear interpolation, but other alternatives are possible; see, e.g., [23] for alternatives. The transformed version of the discrete image $x$ is then expressed as a matrix-vector product $T(y(w))x$.

Using the above definitions of discrete images and their transformations, we then consider the discrete, forward problem for $N$ distinct data observations,

$$d_k = K_k T(y(w_k)) x + \epsilon_k, \quad \text{for all } k = 1, 2, \ldots, N,$$

(2.1)

where $d_k \in \mathbb{F}^{m_k}$ is the measured data, $K_k \in \mathbb{F}^{m_k \times n}$ is a matrix corresponding to the problem-specific image operator, and $\epsilon_k$ is image noise, which we assume to be independently and identically distributed Gaussian noise. In this paper, we focus on the case in which the motion can be modeled by a small number of parameters. In our case, $w_k \in \mathbb{R}^3$ or $\mathbb{R}^6$ models rigid transformations in 2D and 3D, respectively. The total dimension of the motion parameters across all data measurements is then given by $p = 3N$ or $p = 6N$ for $d = 2$ and $d = 3$, respectively. In the application at hand, we note that $p \ll n$. To simplify our notation, we use the column vectors $d \in \mathbb{F}^m$ where $m = m_k \cdot N$ and $w \in \mathbb{R}^p$ to denote the data and motion parameters for all $N$ measurements, respectively.

Given a set of measurements $\{d_1, d_2, \ldots, d_N\}$, the motion-corrected imaging problem consists of jointly estimating the underlying image parameters and the motion parameters in (2.1). We formulate this as the coupled optimization problem

$$\min_{x \in C_x, w \in C_w} \Phi(x, w) = \frac{1}{2} \|KT(w)x - d\|^2 + \frac{\alpha}{2} \|Lx\|^2,$$

(2.2)

where the matrices $K$ and $T$ have the following block structure

$$K = \begin{bmatrix} K_1 & & \\ & \ddots & \\ & & K_N \end{bmatrix} \quad \text{and} \quad T(w) = \begin{bmatrix} T(y(w_1)) \\ T(y(w_2)) \\ \vdots \\ T(y(w_N)) \end{bmatrix}.$$

We denote by $C_x \subset \mathbb{F}^n$ and $C_w \subset \mathbb{R}^p$ rectangular sets used to impose bound constraints on the image and motion parameters. Lastly, we regularize the problem by adding a chosen regularization operator, $L$ (e.g., a discrete image gradient or the identity matrix) and a regularization parameter, $\alpha > 0$, that balances minimizing the data misfit and the regularity of the reconstructed image. We note that finding a good regularization parameter $\alpha$ is a separate, challenging problem which has been widely researched [5, 7, 18, 30]. One strength of LAP is that, for $L = I$, it allows for regularization methods that automatically select the $\alpha$ parameter [5, 12, 13, 14].
In our numerical experiments, we investigate one such hybrid method for automatic regularization parameter selection [5] as well as direct regularization using a fixed $\alpha$ value.

In order to prepare our discussion of LAP, we now provide a brief review of Block Coordinate Descent and Variable Projection, which are two commonly used methods for solving problems such as (2.2).

2.1. Block Coordinate Descent (BCD). BCD represents a fully decoupled approach to solving coupled optimization problems such as (2.2); see, e.g., [24]. In BCD, the optimization variables are partitioned into a number of blocks. The method then sequentially optimizes over one block of variables while holding all the others fixed. After one cycle in which all subsets of variables have been optimized, one iteration is completed. The process is then iterated until convergence. For this paper, we separate the variables into the two subsets of variables suggested by the structure of the problem, one for the image variables and another for the motion variables. At the $k$th iteration, we fix $w^k$ and obtain the updated image $x^{k+1}$ by solving

$$
x^{k+1} = \arg\min_{x \in C} \Phi(x, w^k).
$$

(2.3)

Afterwards, we fix our new guess for $x^{k+1}$ and optimize over the motion $w$,

$$
w^{k+1} = \arg\min_{w \in C} \Phi(x^k, w).
$$

(2.4)

These two steps constitute a single iteration of the method. We note that BCD is decoupled in the sense that while optimizing over one set of variables, we neglect optimization over the other. This degrades convergence for tightly coupled problems [24]. However, BCD is applicable to a wide range of problems, allows for straightforward implementation of bound constraints, and supports various types of regularization. For our numerical experiments, we solve the BCD imaging problem (2.3) inexactly using a single step of projected Gauss-Newton with bound constraints and various regularizers which we introduce below. The optimization problem in the second step (2.4) is small-dimensional and separable, and we perform a single step of Gauss-Newton with a direct solver to compute the search direction.

2.2. Variable Projection (VarPro). VarPro is frequently used to solve problems with separable least-squares problems such as the one in (2.2); see, e.g., [15, 25]. The key idea in VarPro is to eliminate the linear variables (here, the image) by projecting the problem onto a reduced subspace associated with the nonlinear variables (here, the motion) and then solving the resulting reduced, nonlinear optimization problem. In our problem (2.2), eliminating the image variables requires solving a least-squares problem involving the matrix $T(w)$ that depends on the current motion parameters. We express the projection by

$$
x(w) = \arg\min_{x \in C} \Phi(x, w).
$$

(2.5)

Substituting this expression in for $x$, we then obtain a reduced dimensional problem in terms of the nonlinear variable $w$,

$$
\min_{w \in C} \Phi(x(w), w).
$$

(2.6)
The reduced problem is solved to recover the motion parameters, noting that by solving (2.5) at each iteration, we simultaneously recover iterates for the image. Assuming \( C_w = \mathbb{R}^p \) (unconstrained case), we see that the first-order necessary optimality condition in (2.6) is

\[
0 = \nabla_w \Phi(x(w), w) + \nabla_w x(w) \nabla_x \Phi(x(w), w).
\]  

(2.7)

Note that in the absence of constraints on \( x \) (i.e., \( C_x = \mathbb{F}^n \)) the second term on the right hand vanishes due to the first order optimality condition of (2.6). However, this is not necessarily the case when \( C_x \neq \mathbb{F}^n \) or when (2.5) is solved with low accuracy. In those cases \( \nabla_x \Phi(x(w), w) \) does not equal 0 and computing \( \nabla_w x(w) \), which can be as hard as solving the original optimization problem problem (2.2), is inevitable. In such situations, neglecting the second term in (2.7) may considerably degrade the performance of VarPro.

3. Linearize and Project (LAP). We now introduce the LAP method for solving the optimization problem (2.2). We begin by linearizing problem (2.2) following a standard Gauss-Newton framework. In each iteration, computing the search direction then requires solving a linear system that couples the image and motion parameters. Here is where we project onto one block of variables. This offers flexibility in terms of regularization and can handle bound constraints on both the motion and image variables. We introduce our approach to solving the linear coupled problem for the Gauss-Newton step by breaking the discussion into several subsections. We start with a subsection introducing our strategy of projection onto the image space, followed by a subsection on the various options for image regularization that our projection approach offers. Lastly, we discuss imposing bound constraints with our strategy using projected Gauss-Newton.

3.1. Linearizing the Problem. We use a Gauss-Newton framework to solve the coupled problem (2.2). To solve for the Gauss-Newton step at each iteration, we first reformulate the problem by linearizing the residual \( r(x, w) = KT(w)x - d \) around the current iterate, \((x_0, w_0)\). Denoting this residual as \( r_0 = r(x_0, w_0) \), we can write its first order Taylor approximation as

\[
r(x_0 + \delta x, w_0 + \delta w) \approx r_0 + [J_x \quad J_w] \begin{bmatrix} \delta x \\ \delta w \end{bmatrix},
\]

(3.1)

where \( J_x = \nabla_x r_0^T \) and \( J_w = \nabla_w r_0^T \) are the Jacobian operators with respect to the image and motion parameters, respectively. They can be expressed as

\[
J_x = T^T K^T, \quad \text{and} \quad J_w = \text{diag}(\nabla_w (T(y(w_1))x), \ldots, \nabla_w (T(y(w_N))x))^T K^T,
\]

where each term \( \nabla_w (T(y(w_k))x) \) is the gradient of the transformed image at the transformation \( y(w_k) \); see [4] for a detailed derivation. Both \( J_x \in \mathbb{F}^{m \times n} \) and \( J_w \in \mathbb{F}^{m \times p} \) are sparse and we recall that in the application at hand \( p \ll n \).

After linearizing the residual around the current iterate, we substitute the approximation (3.1) for the residual term in (2.2) to get

\[
\min_{\delta x, \delta w} \hat{\Phi}(\delta x, \delta w) = \frac{1}{2} ||J_x \delta x + J_w \delta w + r_0||^2 + \frac{\alpha}{2} ||L \delta x||^2 
\]

(3.2)

subject to \( x_0 + \delta x \in C_x \) and \( w_0 + \delta w \in C_w \).

By solving this problem we obtain the updates for the image and motion parameters, denoted by \( \delta x \) and \( \delta w \), respectively. As (3.2) is based on a linearization it is common practice to solve it only to a low accuracy; see, e.g., [24].
3.2. Projecting the problem onto the image space. To solve the coupled linear problem in (3.2), we propose projecting the problem onto the image space and solving for $\delta x$. To project, we first note that the first order optimality condition for the linearized problem with respect to $\delta w$ is

$$0 = \nabla_{\delta w} (J_x \delta x + J_w \delta w + r_0)^\top (J_x \delta x + J_w \delta w + r_0)$$

or equivalently

$$0 = 2(J_w^T J_w \delta w + J_w^T J_x \delta x + J_w^T r_0).$$

Solving this condition for $\delta w$, we get

$$\delta w = -\left(J_w^T J_w\right)^{-1} \left(J_w^T J_x \delta x + J_w^T r_0\right).$$ (3.3)

We can then substitute this expression for $\delta w$ into (3.2) and group terms. This projects the problem onto the image space and gives a new problem in terms of $\delta x$, the Jacobians $J_x$ and $J_w$, and the residual $r_0$ given by

$$\min_{\delta x} \frac{1}{2} \left\| \left(I - J_w (J_w^T J_w)^{-1} J_w^T\right) J_x \delta x + \left(I - J_w (J_w^T J_w)^{-1} J_w^T\right) r_0 \right\|^2 + \frac{\alpha}{2} \|L \delta x\|^2$$

$$= \min_{\delta x} \frac{1}{2} \left\| P_{J_w}^\perp (J_x \delta x + r_0) \right\|^2 + \frac{\alpha}{2} \|L \delta x\|^2$$

(3.4)

where $P_{J_w}^\perp = I - J_w (J_w^T J_w)^{-1} J_w^T$ is a projection onto the orthogonal complement of the column space of $J_w$. This least squares problem can be solved using an iterative method with an appropriate right preconditioner [3, 22, 28]. In particular, we observe that

$$P_{J_w}^\perp J_x = J_x - J_w (J_w^T J_w)^{-1} J_w^T J_x$$

is a low rank perturbation of the operator $J_x$ since rank($P_{J_w}^\perp$) $\leq p \ll n$. Hence a good preconditioner for $J_x$ should be a suitable preconditioner for the projected operator. Also, we note that $(J_w^T J_w)^{-1}$ $\in \mathbb{R}^{p \times p}$ is symmetric positive-definite if $J_w$ is full rank. Thus, it can be inverted by computing and storing its Cholesky factors once per Gauss-Newton iteration for computational efficiency. We use this strategy in practice with no ill effects, and it increases the efficiency of iteratively solving (3.4). After solving the preconditioned projected problem for $\delta x$, one obtains the accompanying motion step $\delta w$ via (3.3).

3.3. Regularization. After linearizing and projecting onto the space of the linear, image variable, we solve the regularized least squares problem (3.4). For our applications, this problem is high-dimensional and we use an iterative method to approximately solve it with low accuracy. Also, the problem is ill-posed and thus requires regularization. Here we discuss direct, iterative, and hybrid methods for regularization that can be used in LAP.

To begin with, note that if we denote by $A = P_{J_w}^\perp J_x$ and $b = -P_{J_w}^\perp r_0$, then (3.4) is a standard Tikhonov regularized problem

$$\min_{\delta x} \frac{1}{2} \|A \delta x - b\|^2 + \frac{\alpha}{2} \|L \delta x\|^2.$$
Problems of this form are well studied in the literature; see, e.g., [19, 8, 30, 20]. The quality of a given iterative solution for problems of this type depends on the selection of an appropriate regularization parameter $\alpha$ and a regularization operator $L$. Common choices for the regularizer include the discretized gradient operator $L = \nabla_h$ using forward differences and the identity $L = I$. Methods for regularization parameter selection include the discrepancy principle, L-curve, and generalized cross validation (GCV). These methods have limitations: the discrepancy principle requires a priori knowledge about the noise level, while the L-curve and GCV methods often require a singular value decomposition (SVD) of the matrix $A$ which in practice may prove too expensive for large-scale problems.

Another approach is iterative regularization. Iterative regularization exploits the fact that the early iterations of iterative methods like Landweber and Krylov subspace methods contain information about the solution corresponding to large singular values and low-frequency singular vectors of $A$. Additionally, these early iterations are less affected by noise in the data. It follows that during the early iterations of these methods, the iterates tend to contain more information about the true solution, and the relative error between the true solution and the computed approximation at each iteration decreases. However as the methods continue iterating, information corresponding to small singular values and high-frequency singular vectors begins to enter the solution which amplifies noise in the problem eventually causing the relative error of the solution to increase even though the data fit of the solution may improve. This reduction of the relative error during the early iterations followed by its increase in later iterations is known as semi-convergence. The goal of iterative regularization techniques is stop the iterative method at the iteration where the relative error is minimized. In practice, this is difficult because the relative error at each iteration cannot be computed in practice and the quality of the solution is often sensitive to an accurate choice of the stopping iterate. Thus, the main difficulty in effectively using iterative regularization is the accurate determination of stopping criteria for the iterative method without knowledge of the true solution.

Hybrid regularization methods (see e.g., [5, 12, 13, 14] and the references therein) represent a further alternative by seeking to combine the advantages of direct and iterative regularization. Hybrid methods use Tikhonov regularization with a new regularization parameter $\alpha_k$ at each step of an iterative Krylov subspace method such as LSQR [26]. The direct regularization for the early iterates of a Krylov subspace method is possible due to the small dimensionality of the Krylov subspace. This makes SVD-based parameter selection methods for choosing $\alpha_k$ computationally feasible at each iteration of the method. The variability of $\alpha_k$ at each iteration then helps to stabilize the semi-convergence behavior of the iterative regularization from the Krylov subspace method, making the Krylov method less sensitive to the choice of stopping iteration. Thus, hybrid methods seek to combine the advantages of both direct and iterative regularization while avoiding the cost of SVD-based parameter selection for the full-dimensional problem and alleviating the sensitivity to stopping criteria which complicates iterative regularization.

Each of the methods in this paper necessitates solving a regularized system in the image variable given by (3.4) for LAP, (2.5) for VarPro, and (2.3) for BCD. For these problems, we use both direct and hybrid regularization when possible. We begin by running LAP with both the gradient regularizer, $L = \nabla_h$ and a fixed $\alpha$. This approach is also feasible within the VarPro framework subject to the ability to solve the regularized, linear least squares problem to a sufficient degree of accuracy, and
is straightforward to implement using BCD. To test hybrid regularization we use the software for HyBR \[5\] in LAP and BCD to automatically choose $\alpha$ using a weighted GCV method.

### 3.4. Optimization.

In many imaging problems bounds on the image and/or motion variables are known, in which case imposing such prior knowledge into the inverse problem is desirable. To impose bound constraints using LAP, we use projected Gauss-Newton following the description in [17]. This method represents a compromise between a full Gauss-Newton, which converges quickly when applied to the full problem, and projected gradient descent, which allows for the straightforward implementation of bound constraints. For projected Gauss-Newton, we separate the step updates $\delta \mathbf{x}$ and $\delta \mathbf{w}$ into two sets: the set of variables for which the bound constraints are inactive (the inactive set) and the set of variables for which the bound constraints are active (the active set). We denote these subsets for the image by $\delta \mathbf{x}_{\delta} \subset \mathbf{x}$ and $\delta \mathbf{x}_{\mathcal{A}} \subset \delta \mathbf{x}$ where the subscript $\mathcal{I}$ and $\mathcal{A}$ denote the inactive and active sets, respectively. Identical notation is used for the inactive and active sets for the motion.

On the inactive set, we take the standard Gauss-Newton step at each iteration given by the strategy described in Sect. 3.2. This step is computed by solving (3.2)

\[
\min_{\delta \mathbf{x}_{\delta}} \frac{1}{2} \left\| \left( \mathbf{I} - \hat{\mathbf{J}}_{\mathcal{I}} (\hat{\mathbf{J}}_{\mathcal{I}}^\top \hat{\mathbf{J}}_{\mathcal{I}})^{-1} \hat{\mathbf{J}}_{\mathcal{I}}^\top \right) \mathbf{J}_{\delta} \mathbf{x}_{\delta} + \left( \mathbf{I} - \hat{\mathbf{J}}_{\mathcal{I}} (\hat{\mathbf{J}}_{\mathcal{I}}^\top \hat{\mathbf{J}}_{\mathcal{I}})^{-1} \hat{\mathbf{J}}_{\mathcal{I}}^\top \right) \mathbf{r}_0 \right\|^2 + \frac{\gamma}{2} \| \hat{\mathbf{L}} \delta \mathbf{x}_{\delta} \|^2,
\]

where $\hat{\mathbf{J}}_{\mathcal{I}}$, $\hat{\mathbf{J}}_{\mathcal{A}}$, and $\hat{\mathbf{L}}$ are $\mathbf{J}_{\mathcal{I}}$, $\mathbf{J}_{\mathcal{A}}$, and $\mathbf{L}$ restricted to the inactive set via projection.

We then obtain the corresponding motion step on the inactive set by

\[
\delta \mathbf{w}_{\delta} = - \left( \hat{\mathbf{J}}_{\mathcal{A}}^\top \hat{\mathbf{J}}_{\mathcal{A}} \right)^{-1} \left( \hat{\mathbf{J}}_{\mathcal{A}}^\top \hat{\mathbf{J}}_{\mathcal{I}} \mathbf{x}_{\delta} + \hat{\mathbf{J}}_{\mathcal{A}} \mathbf{r}_0 \right).
\]

For the active set, we perform a scaled, projected gradient descent step given by

\[
\begin{bmatrix}
\delta \mathbf{x}_{\mathcal{A}} \\
\delta \mathbf{w}_{\mathcal{A}}
\end{bmatrix}
= - \begin{bmatrix}
\hat{\mathbf{J}}_{\mathcal{I}}^\top \mathbf{r}_0 \\
\hat{\mathbf{J}}_{\mathcal{A}}^\top \mathbf{r}_0
\end{bmatrix} - \alpha \begin{bmatrix}
\hat{\mathbf{L}}^\top \hat{\mathbf{L}} \delta \mathbf{x}_{\mathcal{A}} \\
0
\end{bmatrix}.
\]

where again $\hat{\mathbf{J}}_{\mathcal{I}}$, $\hat{\mathbf{J}}_{\mathcal{A}}$, and $\hat{\mathbf{L}}$ represent the projection of $\mathbf{J}_{\mathcal{I}}$, $\mathbf{J}_{\mathcal{A}}$, and $\mathbf{L}$ onto the active set. We note that the regularization parameter $\alpha$ should be consistent for both (3.5) and (3.7). When using direct regularization with a fixed $\alpha$, this is obvious. However, for the hybrid regularization approach discussed in Sect. 3.3, a choice is required. We set $\alpha$ on the active set at each iteration to be the same as the $\alpha$ adaptively chosen by the hybrid regularization on the inactive set at the same iterate.

The full step for a projected Gauss-Newton iteration is then given by a scaled combination of steps on the inactive and active sets

\[
\begin{bmatrix}
\delta \mathbf{x} \\
\delta \mathbf{w}
\end{bmatrix}
= \begin{bmatrix}
\delta \mathbf{x}_{\delta} \\
\delta \mathbf{w}_{\delta}
\end{bmatrix} + \gamma \begin{bmatrix}
\delta \mathbf{x}_{\mathcal{A}} \\
\delta \mathbf{w}_{\mathcal{A}}
\end{bmatrix}.
\]

Here, the parameter $\gamma > 0$ is a weighting parameter to reconcile the difference in scales between the Gauss-Newton and gradient descent steps. To select this parameter, we
follow the recommendation of [17] and use

$$\gamma = \frac{\max (\| \delta x^I \|_{\infty}, \| \delta w^I \|_{\infty})}{\max (\| \delta x^A \|_{\infty}, \| \delta w^A \|_{\infty})}. \quad (3.9)$$

This choice of $\gamma$ ensures that the projected gradient descent step taken on the active set is no larger than the Gauss-Newton step taken on the inactive set, and we have used it with no ill effects in practice.

After combining the steps for both the inactive and active sets using (3.8), we use a projected Armijo line search for the combined step before updating. A standard Armijo line search chooses a step size $0 < \eta \leq 1$ by backtracking from the full Gauss-Newton step ($\eta = 1$) to ensure a reduction of the objective function. We couple this with a projection that prevents the updated image and motion from violating the bound constraints. Variables that would leave the feasible region are projected onto the boundary and join the active set for the next iteration. However, the projection does not prevent variables from leaving the active set and joining the inactive set. Due to this exchange between the sets, it is necessary to update the inactive and active sets after the line search at each next iteration. We also note that for the special, unconstrained case, the line search reverts to the standard Armijo approach with no need for projection. However for this case, the problem reverts to the standard Gauss-Newton framework which should remove the necessity for a line search.

Lastly, we discuss the choice of stopping criteria for the projected Gauss-Newton method, which again depends on the type of regularization used. For a fixed regularization parameter $\alpha$, we monitor the descent of the objective function and the norm of the projected gradient including the regularizer term. The projected gradient is the first order optimality condition of the projected Gauss-Newton method, and can be computed via a projection [2]. The projection is necessary because the bound constraints prevent gradient entries corresponding to variables with minima outside the feasible region from converging to zero. When using hybrid regularization, we must again consider the variability of $\alpha$ at each iteration. Specifically, selecting a different $\alpha$ parameter at each iteration changes the weight of the regularization term in the objective function (2.2) and the corresponding term in the projected gradient at each iteration. This makes the full objective function value and projected gradient for these methods unreliable stopping criteria. Instead, we omit the regularization term and monitor only the residual term of the objective function and its projected gradient when using hybrid regularization. This allows us to monitor the data fit without the fluctuations associated with the varying $\alpha$.

### 3.5. Summary of the Method

We summarize the discussion of LAP by presenting the entire projected Gauss-Newton algorithm using the LAP framework. This provides a framework to efficiently solve the coupled imaging problems of interest while including the flexible options for regularization and bound constraints which helped motivate our approach. The complete algorithm is given below in Algorithm 1.

### 4. Numerical Experiments

We now test LAP for several coupled problems. To begin with, we test for a low-dimensional exponential curve fitting problem provided by O’Leary and Rust along with their variable projection code [25]. After testing the method for this small example, we then compare it for three imaging problems. These are a two-dimensional super resolution problem, a three-dimensional super resolution problem, and a linear MRI motion correction problem. For all four examples, we compare the results using LAP to those of VarPro and BCD. For the exponential curve fitting problem, we look at the sensitivity of the methods to their starting
Algorithm 1 Linearize and Project (LAP)

1: Given $x_0$, $w_0$ compute the initial active and inactive sets, $A$ and $I$, the initial objective function value $\Phi(x_0, w_0)$, the initial residual $r_0$, and the initial Jacobians $J_x^{(0)}$ and $J_w^{(0)}$
2: for $k = 1, 2, 3, \ldots$ do
3: Use LAP for the Gauss-Newton step on the inactive set solving (3.5) and (3.6) with appropriate regularization and choice of parameter $\alpha$
4: Compute the projected gradient descent step on the active set (3.7)
5: Combine the steps on the active and inactive sets using (3.8) and (3.9)
6: Perform the projected Armijo line search
7: Update active and inactive sets $A$ and $I$, objective function value $\Phi(x_k, w_k)$, residual $r_k$, and Jacobians $J_x^{(k)}$ and $J_w^{(k)}$
8: if Stopping criteria satisfied then
9: return $x_k, w_k$
10: end if
11: end for

guesses. To compare the methods on the imaging problems, we will look at quality of the resultant image, ability of the methods to correctly determine the motion parameters across all data frames, the number of iterations required during optimization, the cost of optimization in terms matrix-vector multiplications, and the CPU time to reach a solution. To compare the quality of the resultant image and motion, we use relative errors with respect to the known, true solutions. The matrix-vector multiplications of interest are those by the Jacobian operator associated with the linear, imaging variable $J_x$. Matrix-vector multiplications by this operator are required in the linear least squares systems for all three methods and are the most computationally expensive operation within the optimization.

4.1. Low-dimensional Exponential Curve Fitting. The first example is a low-dimension exponential curve fitting problem provided as part of O’Leary and Rust’s VarPro implementation [25]. The problem is as follows: given a vector with 10 observations of a function $y(t) = \begin{bmatrix} 7.0 & 5.2 & 2.9 & 1.4 & -0.24 & -0.52 & -1.0 & -1.0 & -0.91 & -0.68 \end{bmatrix}$, evaluated at times $t = \begin{bmatrix} 0.0 & 0.10 & 0.22 & 0.31 & 0.46 & 0.50 & 0.63 & 0.78 & 0.85 & 0.97 \end{bmatrix}$, determine linear parameters $c \in \mathbb{R}^2$ and nonlinear parameters $\mu \in \mathbb{R}^3$ to fit the model curve

$$
\eta(t) = c_1 \exp(-\mu_2 t) \cos(\mu_3 t) + c_2 \exp(-\mu_1 t) \cos(\mu_2 t) = c_1 \Psi_1(\mu, t) + c_2 \Psi_2(\mu, t)
$$

to the data, subject to some weights

$$
m = \begin{bmatrix} 1.0 & 1.0 & 1.0 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 \end{bmatrix}.
$$

This can be expressed within our optimization framework as

$$
\min_{c, \mu} \frac{1}{2} \| M(\Psi(\mu, t)c - y(t)) \|^2_2,
$$

(4.1)
where $M = \text{diag}(m)$ and $\Psi(\mu, t) = [\Psi_1(\mu, t) \ \Psi_2(\mu, t)]$. Note that for $\Psi_1(\mu, t)$ and $\Psi_2(\mu, t)$, the exponential and cosine functions should be evaluated and multiplied element-wise for each entry of $t$ resulting in vectors in $\mathbb{R}^{10}$. The Jacobians $J_c$ and $J_\alpha$ are then given by

$$J_c = M [\Psi_1(\alpha, t) \ \Psi_2(\alpha, t)] \quad \text{and} \quad J_\alpha = M [\nabla_{\mu_1} \Psi \ \nabla_{\mu_2} \Psi \ \nabla_{\mu_3} \Psi] c,$$

where

$$\nabla_{\mu_1} \Psi = -t \odot \exp(-\mu_1 t) \odot \cos(\mu_2 t)$$
$$\nabla_{\mu_2} \Psi = -t \odot \exp(-\mu_2 t) \odot \cos(\mu_3 t) - t \odot \exp(-\mu_1 t) \odot \sin(\mu_2 t)$$
$$\nabla_{\mu_3} \Psi = -t \odot \exp(-\mu_3 t) \odot \sin(\mu_3 t).$$

Here again, the exponential and trigonometric functions are evaluated element-wise for each entry of $t$ and $\odot$ indicates the element-wise Hadamard product.

We chose this example to test the convergence of LAP, VarPro, and BCD for a low dimensional problem with a known solution for many different initial guesses. To this end, we solve the problem with all three methods over a three dimensional grid of initial guesses for the nonlinear parameter $\mu$. We generate the data using the true parameters and no noise is added to ensure a unique global minimum at the true solution,

$$c_{\text{true}} = \begin{bmatrix} 6.0 \\ 1.0 \end{bmatrix} \quad \text{and} \quad \mu_{\text{true}} = \begin{bmatrix} 1.0 \\ 2.5 \\ 4.0 \end{bmatrix}.$$

We then solve the optimization problem with no regularization for any of the methods. A method is deemed convergent if the relative error of the computed solution with respect to the true solution is less than $10^{-4}$. The initial guesses are of the form $\mu_{\text{true}} + p$ where $\mu_{\text{true}}$ is the true solution for the nonlinear parameters and $p \in [-2, 2]^3$ is a perturbation (see below for more details on the set of values used). For VarPro, such an initial guess for the nonlinear parameters suffices. For LAP and BCD, initial guesses for the linear parameters are also necessary, so we use the computed linear parameters from the first iteration of VarPro. For all methods, Cholesky factorization on the normal equations is used in the solver for all linear systems required in the optimization due to the low dimensionality of the problem.

We track which method converges over a $20 \times 20 \times 20$ grid of initial guesses, with the perturbation $p$ chosen in region described above, and plot two-dimensional contour cross-sections of the objective function (4.1) in Fig. 4.1. To create the two-dimensional cross-sections, one of $\mu_1$, $\mu_2$, or $\mu_3$ is set equal to the known, true nonlinear parameter and the other two are the varied initial guesses. From the results displayed, we make a number of observations. Firstly, the contours are non-convex with many local minima as expected for a highly nonlinear problem. For this grid of starting guesses, BCD converges for only one initial guess close to the true solution, which corroborates previous observations that the method is unsuitable for tightly coupled problems (finer grids nearer the solution yield more points of convergence). Both LAP and VarPro converge for a considerably larger number of initial guesses in the grid with VarPro converging for a slightly higher number. Within the grid of initial guesses, VarPro converges for 3,558 of the 8,000 starting guesses while LAP converges for 2,813. However, the set of points where LAP converges is not a subset of the points where VarPro converges. Thus while LAP is somewhat more sensitive to the
The contours show two-dimensional cross-sections of the objective function (4.1) created by fixing one of $\mu_1$, $\mu_2$, or $\mu_3$ on the true parameter and varying the other two. Note that the objective function is highly non-convex with many local minima. The true solution is indicated by a solid, red dot at the center of the plot. On top of the contours, we plot the initial guesses for which LAP, VarPro, and BCD converge. BCD converges for only a very few initial guesses near the true solution. VarPro and LAP are comparable with VarPro converging for the highest number of initial guesses.

-choice of starting guess for this problem, these experiments clearly illustrate that LAP represents a distinct alternative to VarPro with similar performance.

### 4.2. Two-Dimensional Super Resolution

Next, we run numerical experiments using a relatively small two-dimensional super resolution problem. To construct a super resolution problem with known ground truth image and motion parameters, we use the 2D MRI dataset provided in FAIR [23] (original resolution $128 \times 128$) to generate 32 frames of low-resolution test data (resolution $32 \times 32$) after applying 2D rigid body transformations with randomly chosen parameters and adding Gaussian white noise. The resulting total number of parameters in the optimization is 16,528 corresponding to 16,384 for the image and 144 for the motion. In our optimization framework (recalling the notation from Section 2), the imaging operator $K$ is then a block diagonal matrix composed of 32 identical down-sampling matrices $K_k$ along the diagonal, which relate the high resolution image to the lower resolution one via block averaging.

As mentioned in [4], the choice of initial guess is crucial in super resolution problems, so we perform rigid registration of the low-resolution data onto the first volume (resulting in 31 rigid registration problems) to obtain a starting guess for the motion parameters. The resulting relative error of the parameters is around 2%. Using these parameters, we solve the linear image reconstruction problem to obtain a starting guess for the image with relative error is around 5%.

We then solve the problem using LAP, VarPro, and BCD for noise levels of 1%, 2%, and 3%. For all three approaches, we compare two regularization strategies. All three methods are run using the gradient operator, $L = D$ with a fixed regularization parameter $\alpha = 0.01$. LAP and BCD are then run with the Golub-Kahan hybrid regularization approach detailed in 3.3 (denoted as HyBR in tables and figures). As this approach is unfeasible for VarPro, we instead use a fixed $\alpha = 0.01$ and the identity as our second regularizer, $L = I$. For LAP and BCD, we add bound constraints on the image space in the range 0 to 1 for both choices of regularizer, with both bounds active in practice. The number of active bounds varies for different noise levels and realizations of the problem, but can include as many as 30 – 35% of the image variables for both LAP and BCD. VarPro is run without bound constraints. Neither constraints...
nor regularization are imposed on the motion parameters for the three methods.

All three methods require solving two different types of linear systems, one associated with the image variables and another with the motion variables. LAP solves these systems to determine the Gauss-Newton step. We use LSQR [26] with a stopping tolerance of $10^{-2}$ to solve (3.4) for both regularization approaches. The motion step (3.3) is computed using the Cholesky factors of $(J_\omega^T J_\omega)^{-1}$ due to the reduced dimension of the problem. VarPro requires solving the linear system (2.5) within each function call. For both choices of regularization, we solved this system using LSQR with a fixed dimension of 20 for the Krylov subspace. Recall that this system must be solved to a higher accuracy than the similarly sized systems in LAP and BCD to maintain the required accuracy in the gradient; see Sec. 2.2. Gauss-Newton on the reduced dimensional VarPro function (2.6) then requires solving the reduced linear system in the motion parameters, which we solve using Cholesky factorization on the normal equations. For BCD, coordinate descent requires alternating solutions for the linear system in the image, (2.3), and a nonlinear system in the motion parameters, (2.4). For both of these, we take a single projected Gauss-Newton step. For the image, this is solved using LSQR with a stopping tolerance of $10^{-2}$ using MATLAB’s \texttt{lsqr} function for direct regularization and \texttt{HyBR}’s LSQR for hybrid regularization. For the motion, we use Cholesky factorization on the normal equations. During these solves, we track computational costs for the number of matrix-vector multiplications by the Jacobian operator associated with the image, $J_\omega$. For LAP and BCD, these multiplications are required when solving the system for the Gauss-Newton step for the image step, while for VarPro, they are necessary for the least squares solve within the objective function.

We compare results for the methods using the relative errors of the resultant image and motion parameters. We separate relative errors for the image and motion. Plots of the these errors against iteration can be seen in Fig. 4.3 for the problem with 2% added noise. The corresponding resulting images for the 2% error case can be seen in Fig. 4.2. Lastly, a table of relevant values including the average number of iterations, minimum errors for image and motion, matrix-vector multiplications, and CPU timings for the methods taken over 10 different realizations of the problem for all three noise levels is in Table 4.1.

For direct regularization using the discrete gradient operator, the solutions for all three methods are comparable in terms of the relative error for both the image and the motion. Furthermore, these solutions are superior to those using hybrid regularization or the identity operator, suggesting that this is a more appropriate regularizer for this problem. LAP succeeds in accurately recovering the correct motion parameters better than BCD and comparably with VarPro. This is observable for in the relative error plots for the 2% added noise case in Fig. 4.3, and for all three noise levels in Table 4.1. However, the iterations for LAP cost significantly less in terms of time and matrix-vector multiplications than those of VarPro, meaning that each iteration of LAP is significantly faster in terms of CPU time. BCD is also relatively cheap in terms of function calls and CPU time, but it requires more iterations to reach the same level of accuracy. Overall, the LAP approach compares favorably to VarPro and BCD for this example in terms of both the resulting solutions and cost, particularly in recovering the motion parameters for the problem.

4.3. Three-Dimensional Super Resolution. The next problem is a larger three-dimensional super resolution problem. Again, we use a 3D MRI dataset provided in FAIR [23] to construct a super resolution problem with a known ground truth image
and motion parameters. The ground truth image (resolution $160 \times 96 \times 144$) is used to generate 128 frames of low-resolution test data (resolution $40 \times 24 \times 32$). Each frame of data is shifted and rotated by a random 3D rigid body transformation, after which it is downsampled using block averaging. Lastly, Gaussian white noise is added. Again, we run the problem for 1%, 2%, and 3% added noise per data frame. The resulting optimization problem has $2^{121} \times 608$ unknowns, 2,211,840 for the image and 768 for the motion parameters. The data has dimension $5 \times 898 \times 240$. The formulation of the problem is identical to that of the two-dimensional super resolution problem with appropriate corrections for the change in dimension. The imaging operator $K$ in the three-dimensional example is block diagonal with 128 identical down-sampling matrices $K$ which relate the high resolution image to the low resolution data by block averaging.

The initial guess for the three-dimensional problem is generated using the same strategy as in the two-dimensional case. To generate a guess for the motion parameters, we register all frames onto the first frame (thus solving 127 rigid registration problems.) This gives an initial guess for the motion with approximately 3% relative error. Using this initial guess, we then solve a linear least squares problem to obtain an initial guess for the image with a relative error around 13%. We note that this initial guess is not as close to the true solution as the one obtained for the two-dimensional super resolution example. This may impact the quality of the solution obtainable for examples with large amounts of noise.
Table 4.1  

| Noise Level | Method          | Iter. | Rel. Err. x | Rel. Err. w | MatVecs | Time(s) |
|-------------|-----------------|-------|-------------|-------------|---------|---------|
| 1% Noise    | LAP + HyBR      | 22.0  | 1.20e-1     | 1.96e-1     | 204.0   | 18.4    |
|             | LAP + $\nabla_h$| 16.1  | **2.72e-2** | **3.31e-3** | **65.4** | 5.7     |
|             | VarPro + $I$    | 23.1  | 8.06e-2     | 2.29e-2     | 502.0   | 18.9    |
|             | VarPro + $\nabla_h$ | 17.7  | 3.35e-2     | 2.24e-2     | 394.0   | 14.3    |
|             | BCD + HyBR      | **10.0** | 7.67e-2   | 1.05e-2     | 98.2    | 12.1    |
|             | BCD + $\nabla_h$| 28.6  | 3.78e-2     | 2.28e-2     | 117.2   | 24.8    |
| 2% Noise    | LAP + HyBR      | 9.7   | 6.79e-2     | 2.57e-2     | 59.4    | **4.8** |
|             | LAP + $\nabla_h$| 22.7  | 6.55e-2     | 2.63e-2     | 85.3    | 8.9     |
|             | VarPro + $I$    | 26.3  | 1.19e-1     | 2.52e-2     | 566.0   | 20.9    |
|             | VarPro + $\nabla_h$ | 21.1  | **2.50e-2** | **2.50e-2** | 462.0   | 17.8    |
|             | BCD + HyBR      | **8.5** | 8.13e-2   | 2.98e-2     | **40.9** | 6.7     |
|             | BCD + $\nabla_h$| 30.9  | 7.17e-2     | 2.64e-2     | 88.2    | 21.6    |
| 3% Noise    | LAP + HyBR      | 8.9   | 9.63e-2     | 4.98e-2     | 52.3    | 8.9     |
|             | LAP + $\nabla_h$| 14.8  | 9.22e-2     | 4.84e-2     | 82.6    | 9.6     |
|             | VarPro + $I$    | 33.0  | 1.69e-1     | 4.85e-2     | 602.0   | 42.8    |
|             | VarPro + $\nabla_h$ | 23.3  | **8.32e-2** | **4.76e-2** | 430.0   | 34.8    |
|             | BCD + HyBR      | **3.0** | 1.06e-1   | 5.43e-2     | **20.5** | **3.5** |
|             | BCD + $\nabla_h$| 15.7  | 1.09e-1     | 5.43e-2     | 69.5    | 12.3    |

This table shows data for the 2D super resolution for multiple values of added Gaussian noise. The columns from left to right give the stopping iteration, relative error of the solution image, relative error of the solution motion, number of function evaluations during optimization, and time in seconds using \texttt{tic} and \texttt{toc} in MATLAB. All values are averages taken from 10 instances with different motion parameters, initial guesses, and noise realizations. The best results for each column is bold-faced.

For this example, we test LAP, VarPro, and BCD using only the gradient regularizer with a fixed $\alpha = 0.01$ due to its better performance in the 2D example. For LAP and BCD, we implement bound constraints on the image variables restricting the solution element-wise to the range $[0, 1]$. These constraints are not applied to VarPro. Also identical to the two dimensional case, all three methods require solving two linear systems, one larger corresponding to the dimension of the image and one smaller in the dimension of the motion. For LAP and BCD, the larger system is solved using LSQR with a tolerance of $1e-2$ while for VarPro we use LSQR with a Krylov subspace of fixed dimension of 50 in order to achieve the required accuracy. For all three methods, the reduced system in the motion is solved using Cholesky factorization on the normal equations.

The resulting images and relative error plots for the images and motion parameters can be found in Figs. 4.4 and 4.5, respectively. Like the 2D super resolution example, LAP converges faster to the motion parameters in the early iterations than BCD and VarPro, although VarPro does eventually reach lower minima for all three noise levels. However, this difference is not noticeable in the resultant images, and VarPro is far more expensive in terms of matrix-vector multiplications and CPU time as seen in Table 4.2. BCD performs similarly with LAP in terms of the the reconstructed image and the number of function calls required, but it does less well at recovering the motion parameters and requires more CPU time due to the higher number of iterations needed for convergence.

4.4. MRI Motion Correction. The final test problem is a two-dimensional MRI motion correction problem. The goal in this MRI application is to reconstruct a complex-valued MRI image from its Fourier coefficients that are acquired block-wise in
Fig. 4.4. This figure shows two dimensional cross-sections of the reconstructed images and image errors for the 3D super resolution problem with 2% noise. From left to right, the blocks show the initial guess $x_{\text{initial}}$, the absolute value of the error for the initial guess, the solution $x_{\text{LAP}}$ for LAP $+ \nabla h$, and its absolute error. Each cross section was taken by fixing one dimension along its midpoint of the domain with the dotted red lines indicating the cross-sections’ relations to the others.

Fig. 4.5. This figure plots the relative errors for both the reconstructed image and the motion parameters for the 3D super resolution for 2% added noise. LAP succeeds in capturing the correct motion parameters in fewer iterations than VarPro and BCD and in recovering images of comparable quality.

a sequence of measurements. Since the measurement process typically requires several seconds or minutes, in some cases the object being imaged moves substantially. Motion renders the Fourier samples inconsistent and – without correction – results in artifacts and blurring in the reconstructed MRI image. To correct for this, one can instead view the collected MRI data as a set of distinct, complex-valued Fourier samplings, each measuring some portion of the Fourier domain and subject to some unknown motion parameters. The resulting problem of recovering the unknown motion parameters for each Fourier sampling and combining them to obtain a single motion-corrected MRI image fits into the coupled imaging framework presented in this paper.

The forward model for this problem was presented by Batchelor et al. [1]. In their formulation, our imaging operator $K$ is again block diagonal with diagonal blocks $K_k$ for $k = 1, 2, \ldots, N$ given by

$$K_k = A_k F S.$$ 

Here, $S$ is a complex-valued block rectangular matrix containing the given coil sensitivities of the MRI machine, $F$ is block diagonal with each block a two-dimensional Fourier transform (2D FFT), and $A_k$ is a block diagonal matrix with rectangular blocks containing selected rows of the identity corresponding to the Fourier sampling
| Noise (%) | Iter. | Rel. Err. x | Rel. Err. w | MatVecs. | Time(s) |
|-----------|-------|-------------|-------------|----------|---------|
| 1%        | LAP $+ \nabla_h$ | 5.4 | 5.45e-2 | 1.69e-3 | 44.0 | 1.10e3 |
|           | VarPro $+ \nabla_h$ | 17.2 | 7.21e-2 | $9.86e-4$ | 1.87e3 | 1.10e4 |
|           | BCD $+ \nabla_h$ | 14.7 | $4.10e-2$ | 3.20e-3 | 97.7 | 4.41e3 |
| 2%        | LAP $+ \nabla_h$ | 5.2 | 6.67e-2 | 2.06e-3 | 38.0 | 1.47e3 |
|           | VarPro $+ \nabla_h$ | 17.2 | 7.55e-2 | $1.01e-3$ | 1.87e3 | 1.33e4 |
|           | BCD $+ \nabla_h$ | 11.5 | $6.42e-2$ | 4.28e-3 | 72.1 | 5.43e3 |
| 3%        | LAP $+ \nabla_h$ | 5.3 | 8.19e-2 | 2.56e-3 | 36.0 | 1.81e3 |
|           | VarPro $+ \nabla_h$ | 17.4 | $8.10e-2$ | $8.03e-4$ | 1.89e3 | 1.42e4 |
|           | BCD $+ \nabla_h$ | 10.0 | 8.63e-2 | 5.35e-3 | 60.8 | 4.64e3 |

This table presents data for the solution of the 3D super resolution for multiple values of added Gaussian noise. The columns from left to right give the stopping iteration, relative error of the solution image, relative error of the solution motion, number of matrix-vector multiplies during optimization, and time in seconds using `tic` and `toc` in MATLAB. All values are averages taken from 10 separate problems with different motion parameters, initial guesses, and noise realizations.

for the $k$th data observation. As with the other examples, the imaging operator $K$ is multiplied on the right by the block rectangular matrix $T$ with $T(y(w_k))$ blocks modeling the motion parameters of each Fourier sampling. We note that the cost of matrix-vector multiplications by this imaging operator is dominated by the 2D FFTs in block $F$, and we note that for 32 receiver coils, the cost of a single matrix-vector multiplication will require $32N^2$ 2D FFTs where $N$ is the number of Fourier samplings in the data set. Additionally, we note that the presence of these FFT matrices prevents us from explicitly storing the matrix and necessitates passing it as a function call for all of the methods. This also applies to the Jacobian with respect to the image, $J_x$. However, $J_w$ is still computed and stored explicitly.

We use the data set provided in the `alignedSENSE` package [6] to set up an MRI motion correction with a known true image and known motion parameters. To this end, we generate noisy data by using the forward problem (2.1). The ground truth image with resolution $128 \times 128$ is rotated and shifted by a random 2D rigid body transformation. The motion affected image is then observed on 32 sensors with known sensitivities. Each of these 32 observations is then sampled in Fourier space. For our problem, each sampling corresponds to $1/16$ of the Fourier domain, meaning that $N = 16$ samplings (each with unknown motion parameters) are needed to have a full sampling of the whole space. We sample using a Cartesian parallel two-dimensional sampling pattern [6]. The resulting data has dimension $128 \times 128 \times 16 = 524,288$, and the resulting optimization problem has 16,432 unknowns corresponding to 16,384 for the image and 48 for the image. Gaussian noise is then added to each data sampling. We run the problem for 5%, 10%, and 15% added noise.

As with previous examples, we compare the LAP method with VarPro and BCD. The MRI motion correction problem differs from the super resolution examples in that the image data and resulting image are complex-valued while the motion parameters are real-valued. This requires consideration during the optimization when taking gradients or projecting between the space of the real and complex-valued variables. For regularization, we test all three methods using the gradient regularizer for a fixed $\alpha = 0.01$. For LAP and BCD, we test using hybrid regularization with HyBR, and we run VarPro with the identity for a regularizer and $\alpha = 0.01$. As with the super resolution problems, the least squares problem in LAP and the least squares...
problem for the imaging step for BCD are solved using LSQR with a tolerance of $10^{-2}$ for all choices of regularization. For VarPro, we use LSQR with a tolerance of $10^{-8}$ or a maximum of 100 iterations to maintain accuracy in the gradient. As with previous examples, Cholesky factorization on the normal equations is used for the lower-dimensional solves with $J_w$. No bound constraints were applied to any of the methods for this example because element-wise bound constraints on the real or imaginary parts of the complex-valued image variables will affect the angle (or phase) of the solution, which is undesirable.

For an initial guess for the motion parameters, we start with $w = 0$ (corresponding to a relative error of 100%). Using this initialization, we solve a linear least squares problem to get an initial guess for the image. For 10% added Gaussian noise in the data, the initial guess for the image has a relative error of around 35%. We show the initial guess in Fig. 4.6.

LAP, VarPro, and BCD all manage to recover fairly accurate reconstructions of both the image and motion parameters for quite large values of noise using either HyBR or the identity as a regularize; see Figs. 4.6 and 4.7. This is likely due to the fact that the problem is well-posed and highly over determined (32 sensor readings for each point in Fourier space.) For this example, the hybrid regularization approach for LAP and BCD produces the best results, with LAP requiring considerably fewer iterations. We remark that the best regularization from this problem differs from the super resolution problems, and shows the importance of the flexibility which LAP offers for regularizing the image. The comparative speed of LAP is observable for the relative error plots for the problem with 10% noise and further evidenced in Table 4.3 for all noise levels over 10 separate realizations of the problem. For the gradient based regularizer, all three methods do not recover the motion parameters accurately. We also note that the number of iterations and their cost is an important consideration for this problem. Because of the distance of the initial guess from the solution, this problem requires more iterations than the super resolution examples. Additionally, the high number of 2D FFTs required for a single matrix-vector multiplication makes multiplications by the Jacobian $J_x$ and $J_w$ expensive. Table 4.3 shows that LAP outperforms VarPro and BCD for both choices of regularizer by requiring fewer, cheaper iterations in terms of both time and matrix-vector multiplications. The difference in cost is most dramatic when compared with VarPro again due to the large number of FFTs required for a single matrix-vector multiplication and the large number of such multiplications required within each VarPro function call. For BCD and LAP, the number of matrix-vector multiplications is similar, but BCD requires more iterations for convergence. Overall, we see that LAP is a better choice for this problem, and that it provides better reconstructions of both the image and motion in fewer, cheaper iterations.

5. Summary and Conclusion. We introduce a new method, called Linearize And Project (LAP), for solving large-scale inverse problems with coupled blocks of variables in a projected Gauss-Newton framework. Problems with these characteristics arise frequently in applications, and we exemplify and motivate LAP using joint reconstruction problems in imaging that aim at estimating image and motion parameters from a number of noisy, indirect, and motion-affected measurements. By design, LAP is most attractive when the optimization problem with respect to one block of variables is comparably easy to solve. LAP is very flexible in the sense that it supports different regularization strategies and simplifies imposing equality and inequality constraints on both blocks of variables. In our numerical experiments using
four separable least-squares problems, we showed that LAP is competitive and often times superior to VarPro and BCD with respect to accuracy and efficiency.

LAP is as general as alternating minimization methods such as Block Coordinate Descent. While BCD ignores the coupling between the variable blocks when computing updates, LAP takes it into consideration. Thus, in our experiments LAP requires a considerable smaller number of iterations, matrix-vector multiplications, and CPU time than BCD to achieve a similar accuracy.

LAP is not limited to separable least-squares problems and thus more broadly applicable than VarPro. Since LAP projects after linearization, it provides the opportunity to freely choose which block of variables gets eliminated. For example, in our numerical examples in Sec. 4.2–4.4, LAP eliminates the parameters associated with the motion (which are of comparably small dimension) when computing the search direction in the projected Gauss-Newton scheme. Due to the robustness of Gauss-Newton methods it suffices to solve the imaging problem iteratively to a low accuracy. By contrast, VarPro eliminates the image variables that enter the residual in a linear way. While this leads to a small-dimensional nonlinear optimization problem for the motion each iteration requires solving the imaging problem to a relatively high accuracy to obtain reliable gradient information; see also Sec 2.2. This can be problematic for large and ill-posed imaging problems LAP can in some cases reduce runtimes by an order of magnitude; see Tables 4.2 and 4.3.
| Noise Level | Method   | Iter. | Rel. Err. x | Rel. Err. w | MatVecs. | Time(s) |
|------------|----------|-------|-------------|-------------|----------|---------|
| 5% Noise   | LAP + HyBR | 76.6  | 3.56e-3     | 1.59e-4     | 3.18e2   | 5.86e2  |
|            | LAP + V_h | 76.2  | 3.74e-3     | 1.67e-4     | 3.62e2   | 4.32e2  |
|            | VarPro + I | 116.0 | 7.93e-2     | 4.32e-2     | 2.19e4   | 1.20e4  |
|            | BCD + HyBR | 128.6 | 4.35e-2     | 2.27e-2     | 4.03e2   | 1.01e3  |
|            | BCD + V_h | 116.6 | 7.83e-2     | 4.20e-2     | 4.27e2   | 8.59e2  |
| 10% Noise  | LAP + HyBR | 76.8  | 6.56e-3     | 3.25e-4     | 3.11e2   | 6.52e2  |
|            | LAP + V_h | 76.1  | 6.88e-3     | 4.15e-4     | 3.55e2   | 5.19e2  |
|            | VarPro + I | 116.0 | 3.08e-2     | 4.33e-2     | 2.19e4   | 1.24e4  |
|            | BCD + HyBR | 128.2 | 4.53e-2     | 2.21e-2     | 3.89e2   | 1.24e3  |
|            | BCD + V_h | 116.0 | 8.02e-2     | 4.21e-2     | 4.12e2   | 1.05e3  |
| 15% Noise  | LAP + HyBR | 77.3  | 9.49e-3     | 4.69e-2     | 3.07e2   | 5.02e2  |
|            | LAP + V_h | 75.0  | 1.61e-2     | 2.92e-2     | 3.40e2   | 3.61e2  |
|            | VarPro + I | 115.8 | 8.29e-2     | 4.36e-2     | 2.18e4   | 9.98e3  |
|            | VarPro + V_h | 115.7 | 8.28e-2     | 4.35e-2     | 2.18e4   | 8.27e4  |
|            | BCD + HyBR | 127.0 | 4.80e-2     | 2.21e-2     | 3.75e2   | 1.06e3  |
|            | BCD + V_h | 112.9 | 8.20e-2     | 4.17e-2     | 3.99e2   | 8.17e2  |

This table shows the results of LAP, VarPro, and BCD for solving the MRI motion correction example for multiple regularizers and varying levels of added noise. Averaged over 10 realizations of the problem, the columns are stopping iteration, relative error of the solution image, relative error of the solution motion, number of matrix-vector multiplies during optimization, and time in seconds using tic and toc in MATLAB. LAP outperforms the other methods in terms of solution quality, computational cost, and CPU time.

It is worth noting that the key step of LAP, which is the projection onto one variable block when solving the approximated Newton system, is a block elimination, and the reduced system corresponds to the Schur-complement.

To allow for comparison with VarPro we focussed on separable least-squares problems. In future work, we will study the performance of LAP to solve general nonlinear optimization problems.

6. Acknowledgements. This work is supported by Emory’s University Research Committee and National Science Foundation (NSF) awards DMS 1522760 and DMS 1522599.

REFERENCES

[1] P. Batchelor, D. Atkinson, P. Irarrazaval, D. Hill, J. Hajnal, and D. Larkman. Matrix Description of General Motion Correction Applied to Multishot Images. *Magnetic resonance in medicine*, 54(5):1273–1280, 2005.

[2] A. Beck. *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB*. SIAM, 2014.

[3] Å. Björck. *Numerical methods for least squares problems*. SIAM, 1996.

[4] J. Chung, E. Haber, and J. Nagy. Numerical methods for coupled super-resolution. *Inverse Problems*, 22(4):1261–1272, 2006.

[5] J. Chung, J. Nagy, and D. O’Leary. A weighted GCV method for Lanczos hybrid regularization. *Electronic Transactions on Numerical Analysis*, 2008.

[6] L. Cordero-Grande, R. Teixeira, E. Hughes, J. Hutter, A. Price, and J. Hajnal. Sensitivity encoding for aligned multishot magnetic resonance reconstruction. *IEEE Transactions on Computational Imaging*, 2(3):266–280, 2016.

[7] E. de Sturler and M. Kilmer. A Regularized Gauss–Newton Trust Region Approach to Imaging
in Diffuse Optical Tomography. *SIAM Journal on Scientific Computing*, 33(5):3057–3086, 2011.

[8] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Kluwer Academic Publishers, Dordrecht, 2000.

[9] R. Estrin, D. Orban, and M. Saunders. LSLQ: an Iterative Method for Linear Least-Squares with an Error Minimization Property. *Technical Report*, 2017.

[10] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar. Advances and challenges in super-resolution. *International Journal of Imaging Systems and Technology*, 14(2):47–57, 2004.

[11] J. Fohring, E. Haber, and L. Ruthotto. Geophysical Imaging of Fluid Flow in Porous Media. *SIAM Journal on Scientific Computing*, 36(5):S218–S236, 2014.

[12] S. Gazzola and J. G. Nagy. Generalized Arnoldi–Tikhonov method for sparse reconstruction. *SIAM J. Sci. Comput.*, 36(2):B225–B247, 2014.

[13] S. Gazzola and P. Novati. Automatic parameter setting for Arnoldi–Tikhonov methods. *J. Comp. Appl. Math.*, 256(15):180–195, 2014.

[14] S. Gazzola, P. Novati, and M. R. Russo. Embedded techniques for choosing the parameter in Tikhonov regularization. *Num. Lin. Alg. Appl.*, 21(6):796–812, 2014.

[15] G. Golub and V. Pereyra. Separable nonlinear least squares: the variable projection method and its applications. *Inverse Problems*, 19(2):R1, 2003.

[16] G. H. Golub and V. Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM Journal on Numerical Analysis*, 10(2):413–432, 1973.

[17] E. Haber. *Computational Methods in Geophysical Electromagnetics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2014.

[18] E. Haber and D. Oldenburg. A GCV based method for nonlinear ill-posed problems. *Computational Geosciences*, 4:41–63, 2000.

[19] P. C. Hansen. *Rank-deficient and discrete ill-posed problems*. SIAM Monographs on Mathematical Modeling and Computation. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1998.

[20] P. C. Hansen. *Discrete Inverse Problems: Insight and Algorithms*. SIAM, Philadelphia, PA, 2010.

[21] R. Hardie, K. Barnard, and E. Armstrong. Joint map registration and high-resolution image estimation using a sequence of undersampled images. *IEEE Transactions on Image Processing*, 6(12):1621–1633, 1997.

[22] M. Hestenes and E. Stiefel. *Methods of conjugate gradients for solving linear systems*, volume 49. NBS, 1952.

[23] J. Modersitzki. *FAIR: flexible algorithms for image registration*, volume 6 of *Fundamentals of Algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2009.

[24] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science, 35:67–68, 1999.

[25] D. O’Leary and B. Rust. Variable projection for nonlinear least squares problems. *Computational Optimization and Applications. An International Journal*, 54(3):579–593, 2013.

[26] C. C. Paige and M. A. Saunders. LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares. *ACM Transactions on Mathematical Software (TOMS)*, 8(1):43–71, 1982.

[27] S. C. Park, M. K. Park, and M. G. Kang. Super-resolution image reconstruction: a technical overview. *IEEE signal processing magazine*, 20(3):21–36, 2003.

[28] Y. Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[29] D. Sima and S. V. Huffel. Separable nonlinear least squares fitting with linear bound constraints and its application in magnetic resonance spectroscopy data quantification. *Journal of computational and applied mathematics*, 203(1):264–278, 2007.

[30] C. Vogel. *Computational methods for inverse problems*, volume 23. SIAM, 2002.